Departamento de Lenguajes y Sistemas Informáticos e Ingeniería de
Software

Escuela Técnica Superior de Ingenieros Informáticos

Universidad Politécnica de Madrid

POLITÉCNICA

Propuesta de un Modelo de Comportamiento Colectivo de Estudiantes para un Sistema Inteligente de Tutoría dirigido al Entrenamiento Procedimental

Diego Riofrío Luzcando

Dirigida por:

Jaime Ramírez Rodríguez

Tesis que presenta para optar al título de

Doctor en Software y Sistemas

Junio 2017

TRIBUNAL DE TESIS

D^a Angélica de Antonio Jiménez
Profesora titular de la Universidad Politécnica de Madrid
(Presidente)

D^a Ernestina Menasalvas Ruiz
Profesora titular de la Universidad Politécnica de Madrid
(Secretario)

D^a Julia María Clemente Párraga
Profesora titular de la Universidad de Alcalá
(Vocal)

D Gonzalo Méndez Pozo
Profesor contratado doctor de la Universidad Complutense de Madrid
(Vocal)

D Salvador Sanchez Alonso
Profesor titular de la Universidad de Alcalá
(Vocal)

“Tell me and I forget. Teach me and I remember. Involve me and I learn.”

Benjamin Franklin

AGRADECIMIENTOS

Esta página simboliza la satisfacción y el alivio de estar acabando este arduo trabajo, pero a su vez significa la nostalgia de una etapa que se acaba.

Allá por el 2010 comenzaba este proyecto con mucha expectativa, pero también melancolía por dejar a mi gente. Sin embargo, en estos años, he conocido muchas buenas personas que han llegado a ser grandes amigos y a una ciudad que me ha hecho sentir como uno más de los suyos.

Entre esas personas quisiera agradecer en primer lugar a mi director Jaime Ramírez, pues esta tesis es tanto suya como mía, por el tiempo que me ha dedicado en tantas largas reuniones, por haber estado disponible muchas veces cuando he necesitado sin pedirle permiso y por trabajar hasta la madrugada en los artículos.

Otra persona a la que le debo mi gratitud es a Angélica de Antonio, quien fue la primera en creer en mi sin ni siquiera conocerme y abrirme las puertas del laboratorio para colaborar en el proyecto de laboratorios virtuales, que fue el punto de partida de este trabajo. Además, quisiera agradecer a Marta Berrocal y a Mariano Rico, de los cuales he aprendido mucho especialmente en los días que duró dicho proyecto.

Pero principalmente, quisiera agradecer a Valeria, que sin su compañía, apoyo, paciencia y motivación nunca hubiera podido llevar a cabo este proyecto que nos marcamos hace varios años. A mis padres Víctor y María, que son los que más han sufrido por mi ausencia, por haberme enseñado el valor de la educación y el esfuerzo. A mis hermanos Javier y Diana, a los cuales quisiera además pedir disculpas, pues me he perdido varios momentos importantes de sus vidas.

A mi gente de allá, de acá, y a los que están regados por el mundo, entre esos Daniel, Miguel, Rafael, Marcelo y Fernando, que a pesar de haber estado cada uno en una punta del mundo hemos estado juntos siempre. A Cristian y Pamela, pues hemos sufrido juntos las penas de este proceso, pero también nos hemos alegrado con cada logro.

Finalmente, agradezco a la Secretaría de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) y al Instituto de Fomento y Talento Humano, ambas instituciones del Estado ecuatoriano, por la financiación otorgada para el desarrollo de esta tesis.

RESUMEN

La educación ha sacado provecho de los sistemas computacionales desde los años cuarenta. De ahí nacen los entornos de aprendizaje por computador, que luego con el paso del tiempo y la evolución tecnológica, han ido mejorando inspirados en las teorías constructivistas del aprendizaje.

A finales de los años ochenta, surgen los llamados Sistemas Inteligentes de Tutoría (SIT), los cuales son aplicaciones instruccionales que utilizan técnicas de inteligencia artificial para decidir lo que se debe enseñar en cada momento a un alumno y/o las estrategias más adecuadas para ayudarlo a asimilar mejor los conocimientos. Para ello, es muy importante conocer al alumno para poder adaptar la enseñanza a sus necesidades. Por esta razón se han desarrollado diferentes modelos del estudiante, los cuales representan el conocimiento que tiene el SIT acerca de un alumno. Estos sistemas contienen arquitecturas e implementaciones complejas, las cuales normalmente necesitan poseer información a priori del estudiante. Ahora bien, si el SIT no tiene suficiente información acerca de un estudiante, por ejemplo, en el caso de alumnos nuevos, un modelo de este tipo no podrá otorgar predicciones fiables acerca del futuro rendimiento de dicho alumno.

Por otro lado, la minería de datos es conocida por su capacidad para predecir el comportamiento de los usuarios. Y, gracias a la popularidad de los entornos de aprendizaje por computador, la comunidad científica ha tenido acceso a una gran cantidad de datos que detallan el comportamiento de los estudiantes en esos sistemas. De la aplicación de la minería de datos a los datos registrados en estos entornos de aprendizaje, nace la disciplina conocida como *Educational Data Mining*.

Sin embargo, existen pocos trabajos en la literatura que exploren el potencial del *Educational Data Mining* a la hora de predecir el comportamiento de los estudiantes en entornos de entrenamiento procedimental. Es precisamente para proponer una solución a este problema, en donde esta tesis realiza su principal aportación, un modelo de comportamiento colectivo de estudiantes. El principal objetivo de este modelo es predecir las acciones que podría realizar un estudiante en un entorno de entrenamiento procedimental, para así poder generar una tutoría basada en esta predicción. Para ello, este modelo se construye a partir de los registros de actividad de estudiantes (*logs*) que utilizaron

dicho entorno de entrenamiento con el mismo fin. Estos *logs* son divididos inicialmente en grupos mediante una técnica de clusterización. Posteriormente, para cada clúster se crea un autómata extendido a partir de las secuencias de eventos encontradas en los registros de actividad de los estudiantes en dicho clúster, siguiendo una técnica exhaustiva de minería de secuencias.

Esta tesis también presenta el diseño de un visualizador interactivo del modelo de comportamiento colectivo de estudiantes. Este visualizador ha sido diseñado siguiendo las recomendaciones de expertos en usabilidad de sistemas de presentación de información, con el fin de que la interacción con el autómata sean lo más intuitivas y efectivas posibles.

Además, se presenta una estrategia de tutoría, que, mediante la técnica pedagógica del andamiaje, proporciona una ayuda adaptada a las necesidades de cada estudiante para que sea capaz de superar su zona de desarrollo próximo a lo largo del proceso de entrenamiento. Esta estrategia recurre, en primer lugar, al modelo del estudiante propuesto en otra tesis doctoral, y si este modelo no puede proporcionar predicciones fiables, recurre en segunda instancia al modelo propuesto en esta tesis.

La capacidad de predicción de este modelo y la utilidad del visualizador para mejorar la retroalimentación otorgada por un SIT o la docencia de un profesor, han sido validadas utilizando los *logs* registrados en un laboratorio virtual 3D de biotecnología.

Como resultado de estas validaciones se concluyó que el modelo puede proveer predicciones lo suficientemente fiables como para que un SIT ofrezca un soporte adaptado a cada tipo de alumno. Además, se comprobó que el visualizador permitió realizar mejoras en la retroalimentación del tutor automático integrado en un entorno de aprendizaje procedimental, en la medida de que se redujeron los errores de los alumnos en las partes de la práctica cuya tutoría fue mejorada.

ABSTRACT

Education has taken advantage of computer systems since the forties. In this way computer based learning environments were born, which then with the passage of time and technological evolution, have gone improving inspired by constructivist learning theories.

In the late 1980s, the so-called Intelligent Tutoring Systems (ITSs) emerged, which are computer based instructional systems that use artificial intelligence techniques to define what should be taught to a student in each moment and/or the strategies to help him/her to better assimilate knowledge. For this, it is important to know the student's features to adapt the instruction to his/her needs. For this reason, different student models have been developed to represent the knowledge that the ITS owns about a student. These models contain complex architectures and implementations that typically need prior student's information. Nevertheless, this kind of models cannot give reliable predictions on the future performance of a student, if the ITS does not have enough information about this student, for example, in the case of new students.

Additionally, data mining is well known for its capacity for predicting user behavior. And, thanks to the popularity of computer based learning environments, the scientific community has had access to a large amount of data that detail the behavior of students in those systems. From the application of data mining to the data logged in this kind of learning environments, a discipline known as Educational Data Mining was born.

However, there are few papers in the literature that explore the potential of Educational Data Mining to predict students' behavior in procedural training environments. It is precisely to propose a solution to this problem, where this thesis makes its main contribution, a behavior collective student model. The main objective of this model is to predict actions of new students in a procedural training environment, in order to enable a tutoring feedback adapted to this prediction. For that, this model is built based on activity logs from students that used the training environment with the same purpose. These logs are initially divided into groups using a clustering technique. Subsequently, an extended automaton is created for each cluster from the sequence of events found in the logs of that cluster by following an exhaustive sequence mining technique.

This thesis also presents the design of an interactive visualizer of the behavior collective student model. This visualizer has been designed according to the usability experts' recommendations on information presentation systems, in order to make the interaction with the automaton as intuitive and effective as possible.

In addition, this thesis outlines a tutoring strategy, which relying on scaffolding technique, gives a tutoring feedback adapted to the needs of each student so that he/she can overcome its zone of proximal development throughout the training process. This process first uses the results obtained from the student model proposed in another doctoral dissertation. But if this model cannot provide reliable predictions, this strategy resorts to the model proposed in this work.

The predictive capacity of this model and the utility of the visualizer for improving the ITS feedback or the teaching of a teacher have been validated using the activity logs collected in a 3D biotechnology virtual laboratory.

As a result of these validations we can conclude that the model can provide reasonably good predictions as for a ITS to offer adapted support to each type of student. In addition, it was found that the visualizer supported to make improvements to the automatic tutor feedback integrated in a procedural learning environment. These improvements reduced students' errors in the parts of the practice where tutoring was enhanced.

ÍNDICE

ÍNDICE DE FIGURAS	XIX
ÍNDICE DE TABLAS	XXIII
ÍNDICE DE ALGORITMOS	XXV
1. INTRODUCCIÓN	1
1.1. Contexto y Motivación	1
1.2. Objetivos Generales	4
1.3. Resumen de la Solución Adoptada	4
1.4. Estructura del Documento	6
I PLANTEAMIENTO DEL PROBLEMA	9
2. FUNDAMENTOS TEÓRICOS	11
2.1. Fundamentos Pedagógicos	11
2.1.1. Teoría del desarrollo cognitivo	13
2.1.2. Teoría del aprendizaje sociocultural	14
2.1.3. Zona del desarrollo próximo	15
2.1.4. Teoría de la actividad	17
2.1.5. Tipos de problemas	18
2.1.6. Andamiaje	18
2.1.7. Aprender haciendo	21
2.1.8. Aprendizaje autorregulado	22
2.2. Entornos de Aprendizaje por Computador	22
2.2.1. Entornos virtuales 3D	23
2.2.2. Sistemas inteligentes de tutoría	27
2.3. Minería de Datos	30

2.3.1.	Descubrimiento de conocimiento en base de datos	31
2.3.2.	Fase de minería de datos en proceso de descubrimiento de conoci- miento en base de datos	32
2.3.3.	Análisis de agrupamiento o clustering	35
2.3.4.	Minería de datos secuenciales	37
2.3.5.	Herramientas para la minería de datos	42
3.	ANTECEDENTES	47
3.1.	MAEVIF	47
3.1.1.	Arquitectura	48
3.2.	Modelo del Estudiante de MAEVIF	50
3.2.1.	Ontología de modelado del estudiante	51
3.2.2.	Método de diagnóstico pedagógico	52
3.3.	Laboratorio Virtual de Biotecnología	53
3.3.1.	Protocolo del laboratorio	55
3.3.2.	Tutor automático	56
3.3.3.	Comunicación entre objetos	60
3.4.	Discusión	60
4.	ESTADO DEL ARTE	63
4.1.	Minería de Datos en la Educación	63
4.1.1.	Descubrimiento de conocimiento en base de datos de entornos educativos	65
4.1.2.	Trabajos relacionados	66
4.2.	Visualización de información de datos de educación	68
4.3.	Sistemas inteligentes de tutoría para entrenamiento procedimental inte- grados con minería de datos	69
4.3.1.	Metodología de búsqueda	69
4.3.2.	Trabajos relacionados	72
4.4.	Discusión	73
5.	DEFINICIÓN DEL PROBLEMA	75
5.1.	Objetivos de la Tesis	76
5.1.1.	Propuesta de un modelo de comportamiento colectivo de estudian- tes	76
5.1.2.	Diseño de una visualización del modelo de comportamiento colecti- vo de estudiantes	77

5.1.3. Diseño de una estrategia de tutoría basada en el modelo de comportamiento colectivo de estudiantes	78
5.2. Supuestos de la Tesis	78

II SOLUCIÓN ADOPTADA 81

6. MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES 83

6.1. Integración del Modelo propuesta en un Sistema Inteligente de Tutoría	84
6.1.1. Integración del modelo de comportamiento colectivo de estudiantes en MAEVIF	87
6.2. Descripción del Modelo de Comportamiento Colectivo de Estudiantes	87
6.3. Definición del Autómata Extendido	89
6.3.1. Zona de flujo correcto	91
6.3.2. Zona de errores irrelevantes	92
6.3.3. Zona de errores relevantes	93
6.3.4. Definición formal del autómata	95
6.4. Ejemplos de aplicabilidad del autómata extendido	96
6.4.1. Autómata para el mantenimiento de bujías	97
6.4.2. Autómata para el laboratorio virtual de biotecnología	101
6.5. Proceso de construcción del autómata	104
6.6. Proceso de actualización del modelo	106
6.7. Proceso de reclasificación de un estudiante	106
6.8. Funciones de Agrupamiento	108

7. VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES 111

7.1. Posicionamiento y Representación de los Estados	114
7.2. Representación de las Aristas	118
7.3. Dimensionamiento del Grafo	119
7.4. Interacción con el Grafo	119
7.4.1. Acercamiento	119
7.4.2. Filtrado	122
7.4.3. Detalles bajo demanda	122
7.4.4. Relación	124
7.5. Ejemplos de Aplicaciones del Visualizador	125

7.5.1. Calificar a los alumnos	126
7.5.2. Comparar el comportamiento de los alumnos de diferentes clústeres	129
7.5.3. Encontrar errores comunes	131
8. ESTRATEGIA DE TUTORÍA	137
8.1. Propuesta de Estrategia de Tutoría	138
8.2. Selección de Tutoría para el Estudiante	140
8.2.1. Tipos de tutoría	141
8.2.2. Tipos y niveles de pistas	142
8.3. Proceso de Selección de Tutoría	147
8.4. Tutoría basada en el Modelo del Estudiante	151
8.4.1. Tutoría para actividad	151
8.4.2. Tutoría para la zona de desarrollo próximo del estudiante	154
8.5. Tutoría basada en el Modelo de Comportamiento Colectivo de Estudiantes	156
8.5.1. Tutoría para la zona de desarrollo próximo del estudiante	156
8.5.2. Tutoría para la prevención de errores	160
III RESULTADOS Y APORTACIONES	165
9. VALIDACIÓN	167
9.1. Validación de la Precisión de las Predicciones del Modelo de Comportamiento Colectivo de Estudiantes	169
9.1.1. Cálculo del error promedio del modelo de comportamiento colectivo de estudiantes	169
9.1.2. Resultados	173
9.1.3. Discusión	176
9.2. Validación de la Clasificación de los Alumnos	178
9.2.1. Resultados	178
9.2.2. Discusión	180
9.3. Validación de la Utilidad del Visualizador	181
9.3.1. Resultados	184
9.3.2. Discusión	190
10. CONCLUSIONES Y TRABAJOS FUTUROS	193
10.1. Conclusiones	193
10.2. Trabajos Futuros	195

APÉNDICE A. PROTOCOLO Y CONFIGURACIÓN DE TUTORÍA DEL LABORATORIO VIRTUAL DE BIOTECNOLOGÍA	211
--	-----

ÍNDICE DE FIGURAS

2.1. Modelo de mediación del aprendizaje de Vygotsky	15
2.2. Tríada de sujeto, objeto y artefactos mediadores.	15
2.3. Zona del Desarrollo Próximo.	16
2.4. Ejemplo de una ZDP.	17
2.5. Descomposición de la actividad	17
2.6. Cono de la experiencia de Dale	21
2.7. Arquitectura clásica de un SIT.	28
2.8. Descubrimiento de conocimiento en base de datos (KDD).	31
2.9. Ejemplo de un grafo.	40
2.10. Ejemplo de un digrafo.	41
2.11. Ejemplos de episodios	42
2.12. Seleccionador de interfaz de WEKA.	43
2.13. <i>Explorer</i> de WEKA.	44
2.14. RapidMiner.	45
2.15. Visualización de datos en RapidMiner.	46
3.1. Arquitectura de MAEVIF.	48
3.2. Principales relaciones ad-hoc entre las ontologías modulares	51
3.3. Esquema del método de diagnóstico pedagógico	53
3.4. Laboratorio virtual de biotecnología.	54
3.5. Esquema de protocolo para el laboratorio virtual de biotecnología.	55
3.6. Protocolo para la preparación de medio de enraizamiento en el laboratorio virtual de biotecnología.	56
3.7. Ejemplos de mensajes de tutoría en el laboratorio virtual. Los globos grandes son imágenes ampliadas de las capturas de pantalla.	57
3.8. Ejemplo de acciones y sus interdependencias.	58

ÍNDICE DE FIGURAS

3.9. Ejemplo de la secuencia de mensajes intercambiados entre el avatar y algunos de los objetos involucrados en la práctica, siguiendo las acciones y dependencias mostradas en la figura 3.8.	61
4.1. Principales disciplinas relacionadas con el EDM	64
4.2. Aplicación de la minería de datos en el diseño de sistemas educativos .	64
4.3. KDD en EDM	65
4.4. Número de trabajos relacionados por concepto afín a esta tesis.	71
6.1. Arquitectura del SIT extendido con el MCCE.	85
6.2. Arquitectura de MAEVIF con Agente de Comportamiento Colectivo de Estudiantes.	86
6.3. Modelo de Comportamiento Colectivo de Estudiantes.	88
6.4. Ejemplo de un autómatas extendido del MCCE.	89
6.5. Ejemplo de proceso de limpieza y cambio de bujías.	97
6.6. Ejemplo de autómatas del proceso de la figura 6.5.	98
6.7. Ejemplo del proceso del laboratorio virtual de biotecnología.	101
6.8. Ejemplo de autómatas del proceso de la figura 6.7.	102
7.1. Captura de pantalla del visualizador del MCCE.	112
7.2. Visualización de un estado. flt1 representa al código de la acción de obtener un vaso de precipitados	115
7.3. Disposición de los estados de errores relevantes desde un estado correcto.	117
7.4. Estado de acciones ya realizadas.	117
7.5. Estado de acciones no encontradas.	117
7.6. Visualización de un evento (arista).	118
7.7. Visualización del peso de un evento.	118
7.8. Ejemplos de disposiciones del autómatas.	120
7.9. Ejemplos de filtros de frecuencias en el autómatas.	121
7.10. Vista por estudiante del visualizador del MCCE.	123
7.11. Ejemplos de relaciones (colores) de eventos por estado.	125
7.12. Secuencia de estados correctos del estudiante 22 del laboratorio de biotecnología.	127
7.13. Secuencia de estados del estudiante 13 del laboratorio de biotecnología. .	127
7.14. Secuencias de estados del estudiante 27 del laboratorio de biotecnología. .	128
7.15. Visualización global del autómatas de cada clúster.	129
7.16. Primer error en el autómatas del clúster 0.	130
7.17. Primera secuencia de errores en el autómatas del clúster 0.	131

7.18. Comparación de errores al finalizar la mezcla de la fase 2 en los autómatas de los dos grupos de estudiantes.	132
7.19. Secuencia final de estados de los autómatas de los dos grupos de estudiantes.	133
7.20. Error irrelevante frecuente poco importante para el profesor.	133
7.21. Error irrelevante frecuente importante para el profesor.	134
7.22. Errores relevantes frecuentes.	134
8.1. Estrategia de tutoría.	139
8.2. Componentes del módulo de tutoría.	141
8.3. Actividades, acciones y objetivos de aprendizaje.	142
8.4. Proceso general de tutoría.	148
8.5. Definición del cálculo del ZDP para el ME.	154
8.6. Definición del cálculo del ZDP para el MCCE.	157
8.7. Ejemplo de ZDP para autómata de mantenimiento de bujías.	158
8.8. Ejemplo de ZDP para autómata del laboratorio de biotecnología.	158
9.1. Media de error por método y función de agrupamiento cuando <i>soporte</i> = 0.	177
9.2. Pruebas de normalidad de las calificaciones de alumnos entre 2013 y 2015.	185
9.3. Pruebas de normalidad de las calificaciones de alumnos del 2016.	185
9.4. Pruebas de normalidad de las diferencias de frecuencias de error para los dos períodos.	187

ÍNDICE DE TABLAS

4.1. Palabras clave por concepto para búsqueda de trabajos relacionados. . . .	70
4.2. Número de artículos por temática no relevante de los 21 encontrados. . .	71
9.1. Estados por zona del MCCE de validación sin clusterizar	169
9.2. Gama de colores para valores inferiores de medias de error de predicción.	173
9.3. Media del error por método para agrupamiento por errores.	173
9.4. Media del error por método para agrupamiento por errores y tiempo. . .	174
9.5. Media del error por método para agrupamiento por eventos por zona. . .	174
9.6. Métodos seleccionados por función de agrupamiento.	175
9.7. Media de error por método y función de agrupamiento.	176
9.8. Gama de colores para valores de frecuencias de cada error por clúster. . .	178
9.9. Frecuencias por clúster para el agrupamiento por errores (EM)	179
9.10. Frecuencias por clúster para el agrupamiento por errores y tiempo (XMeans)	179
9.11. Frecuencias por clúster para el agrupamiento por eventos por zona (XMeans)	180
9.12. Frecuencias por clúster para el agrupamiento por secuencias	180
9.13. Errores comunes encontrados gracias al visualizador	182
9.14. Cambios en mensajes de tutoría	183
9.15. Cambios en nombres de químicos	183
9.16. Otros cambios en el laboratorio virtual	184
9.17. Resultados de la prueba T para las diferencias de errores cometidos. . . .	187
9.18. Gama de colores para valores de frecuencias de cada error por clúster. . .	187
9.19. Frecuencias en estados de error con cambios de mensajes generales	188
9.20. Frecuencias en estados de error con cambios que incluyen mensajes específicos	188
9.21. Frecuencias en estados de error con cambios en el entorno virtual	189
9.22. Frecuencias en estados de error con cambios de mensaje al final de cada fase	189

ÍNDICE DE ALGORITMOS

1.	Proceso de construcción del autómata	104
2.	Proceso de selección de tutoría	149
3.	Proceso de tutoría para una actividad	153
4.	Proceso de tutoría para el ZDP de un estudiante usando el ME	155
5.	Proceso de tutoría para el ZDP de un estudiante usando el MCCE	159
6.	Proceso de tutoría para la prevención de errores usando el MCCE	161
7.	Proceso de validación	171

CAPÍTULO 1

INTRODUCCIÓN

1.1. Contexto y Motivación

Eco (2001) argumenta que una tesis debe tener como principal característica la originalidad, es decir, una tesis debe pretender “descubrir” lo que no haya sido desarrollado por los demás estudiosos de la misma área. Además, una tesis debe abordar el trabajo en una disciplina específica del conocimiento humano, ya que tratar de abarcar todo el campo de conocimiento sería imposible. En el caso de esta tesis esa disciplina es la informática, y más concretamente, los Sistemas Inteligentes de Tutoría (SIT).

Este trabajo se engloba en el campo de los entornos de aprendizaje por computador, los cuales nacieron inspirados en las teorías constructivistas del aprendizaje, especialmente, las desarrolladas por Piaget (1969); Piaget y Duckworth (1970) y Vygotsky (1978). Los entornos de aprendizaje por computador, más conocidos actualmente como sistemas de *e-learning*, ayudan a los estudiantes en su proceso de aprendizaje. Existe una amplia diversidad de este tipo de sistemas de enseñanza (Lajoie & Derry 1993), como, por ejemplo, los entornos hipermedia, los de enseñanza por medio de la web (los más populares actualmente), los simuladores, etc. Dentro de los simuladores podemos encontrar los entornos virtuales tridimensionales (3D), los cuales en ocasiones incluyen características propias de la realidad virtual.

Algunos de estos sistemas de *e-learning* utilizan sistemas inteligentes de tutoría que guían al alumno en la realización de una tarea, proporcionan soporte para lograr un objetivo de aprendizaje o brindan mensajes de ayuda cuando el estudiante lo necesita (Sleeman & Brown 1982).

Los sistemas inteligentes de tutoría se están utilizando en diversas aplicaciones desde hace ya varios años, siendo una de sus aplicaciones más destacadas el entrenamiento militar. En este sentido, cabe destacar que dentro del laboratorio de investigación

del ejército de Estados Unidos, específicamente en el *Learning in Intelligent Tutoring Environments (LITE) Laboratory*, se puso en marcha un ambicioso proyecto llamado *Generalized Intelligent Framework for Tutoring (GIFT)*¹ para intentar estandarizar las herramientas y métodos utilizados en la construcción de sistemas de tutoría automáticos.

Por otro lado, un área muy conocida dentro de la informática es la minería de datos. Las técnicas y herramientas de minería de datos han ayudado a analizar las grandes cantidades de datos que proporcionan algunos sistemas de información. De esta forma, se puede obtener un conocimiento que puede ser valioso en la toma de decisiones sobre el ámbito en el que han sido tomados los datos, es decir, éstos pasan de ser un “producto” a ser la “materia prima” (Hernández Orallo, Ramírez Quintana, & Ferri Ramírez 2004).

Los entornos de aprendizaje por computador son sistemas de información que pueden generar una gran cantidad de datos respecto al comportamiento que han tenido los estudiantes en ellos. Además, estos datos pueden ser analizados mediante técnicas de minería de datos para descubrir patrones interesantes desde el punto de vista pedagógico. Esto motivó el nacimiento de la disciplina conocida como *Educational Data Mining (EDM)* (Romero & Ventura 2007; 2010; Romero, Ventura, Pechenizkiy, & Baker 2010; Baker 2010).

La minería de datos centrada en la educación ya ha alcanzado resultados prometedores, por ejemplo, en lo que respecta al análisis del rendimiento de los estudiantes o la predicción de sus futuras calificaciones, especialmente, en el campo del *e-learning* por medio de la web. Sin embargo, existen muy pocos trabajos en la literatura de *e-learning* que hayan integrado las técnicas de minería de datos en los sistemas de tutoría inteligente Baker (2014), por ejemplo, para proporcionar una tutoría personalizada para cada estudiante. Es precisamente en este campo en donde esta tesis doctoral pretende realizar una aportación original al estado del arte.

Esta tesis presenta un modelo de comportamiento colectivo de estudiantes, el cual ha sido diseñado para predecir las acciones de los estudiantes mientras utilizan un entorno educativo de entrenamiento procedimental. En un entorno de estas características un estudiante aprende a llevar a cabo un procedimiento, que, a su vez, está formado por una serie de acciones básicas. En esta clase de entornos, además de las acciones correctas, el estudiante tendrá libertad para realizar otras acciones, que en unos casos serán irrelevantes y en otros impedirán la consecución de los objetivos de la práctica.

El modelo de comportamiento colectivo de estudiantes se crea a partir de los registros de actividad o *logs* procedentes de estudiantes con una formación similar que completaron el mismo entrenamiento con anterioridad. Como se detalla más adelante, un SIT equipado

¹<https://gifttutoring.org>

con este modelo puede usar pistas para prevenir ciertos errores de los estudiantes o para facilitarles la realización de ciertas tareas.

Este modelo se inspira en los resultados del análisis etnográfico del comportamiento de estudiantes al momento de realizar una práctica de laboratorio en un entorno virtual 3D de entrenamiento desarrollado por el autor en su tesis de máster (Riofrío-Luzcando, Ramírez, & Berrocal-Lobo 2012; Rico, Ramírez, Riofrío, Berrocal-Lobo, & De Antonio 2012). Este entorno implementa un laboratorio virtual de biotecnología, en el que uno o más estudiantes pueden realizar una práctica de ingeniería genética. Esta práctica consiste en transformar genéticamente un árbol (chopo) para dotarlo de una mayor resistencia a enfermedades producidas por ciertos hongos.

Asimismo, este laboratorio virtual dispone de un tutor automático capaz de supervisar las acciones de los estudiantes y proporcionarles pistas. Ahora bien, este tutor no es capaz de proporcionar una tutoría adaptada a las necesidades de cada alumno. Esto implica entonces que todos los alumnos reciben la misma ayuda en un mismo paso de la práctica o cuando cometen un error independientemente de cuál haya sido su rendimiento en la práctica o su historial académico.

Por otro lado, el tutor automático va registrando las acciones que realizan los estudiantes y los errores que cometen mientras realizan la práctica. Eso significa que el tutor dispone de una gran cantidad de información que podría ser utilizada o bien para inferir las necesidades de aprendizaje de cada alumno durante la práctica, o bien para predecir el comportamiento de alumnos con características similares en la misma situación.

Al observar a los alumnos utilizar este laboratorio virtual, nos dimos cuenta de que había algunos alumnos que cometían errores similares, formulaban preguntas parecidas, tenían dificultades para encontrar las mismas herramientas, etc. Por otra parte, existían otros alumnos que apenas cometían errores y progresaban rápidamente en la práctica. Es decir, nos apercebimos de que existían grupos de alumnos con comportamientos similares. Por esta razón, vimos interesante la idea de agrupar los *logs* de los alumnos de acuerdo al rendimiento que habían demostrado en la práctica, con el fin de identificar grupos de alumnos que sean susceptibles de recibir la misma tutoría.

Otra cuestión que se planteó al utilizar el laboratorio virtual fue la necesidad por parte del profesor de poder visualizar el comportamiento de cada alumno por separado o de todo un grupo tras completar la práctica.

Este laboratorio virtual ha sido utilizado desde el año 2013 hasta el 2016 por los estudiantes de la asignatura de Bioquímica de la Escuela Técnica Superior de Ingenieros de Montes de la Universidad Politécnica de Madrid. Los registros de actividad de estos alumnos han sido de gran ayuda para la realización de esta tesis doctoral, puesto que han

servido para validar la precisión del modelo de comportamiento colectivo y la utilidad del visualizador de dicho modelo.

1.2. Objetivos Generales

De lo planteado anteriormente, y, especialmente, del análisis de nuestras experiencias con el laboratorio virtual de biotecnología, se han derivado los siguientes objetivos que pretende alcanzar esta tesis:

- OG₁* Proponer un modelo de comportamiento colectivo de estudiantes que, utilizando los registros de actividad de alumnos, pueda predecir, con un cierto nivel de certeza, eventos futuros que pueda provocar un estudiante al realizar una práctica en un entorno de entrenamiento procedimental.
- OG₂* Diseñar una visualización interactiva del modelo de comportamiento colectivo de estudiantes, siguiendo las pautas recomendadas en la literatura de usabilidad para la visualización de la información.
- OG₃* Diseñar una estrategia de tutoría que utilice el modelo individual del estudiante propuesto por Clemente (2011); Clemente, Ramírez, y de Antonio (2011) y el modelo de comportamiento colectivo propuesto en esta tesis, para proporcionar una tutoría adaptada a cada estudiante.

1.3. Resumen de la Solución Adoptada

El modelo de comportamiento colectivo de estudiantes propuesto consiste en varios grupos de estudiantes o clústeres, cada uno de los cuales contiene un autómata extendido. Este autómata extendido se representa gráficamente mediante un grafo dirigido, y consiste en una variante de un autómata finito adaptado a los objetivos de este trabajo.

Como se justificará más adelante, la agrupación de los estudiantes en clústeres permite proporcionar una tutoría automática más adaptada a cada tipo de estudiante. Por otro lado, el autómata extendido representa una síntesis del comportamiento de todos los alumnos de un clúster al realizar una práctica. Para ello, este autómata modela mediante transiciones los eventos provocados por los alumnos del clúster al realizar la práctica, e incluye las frecuencias asociadas a cada evento y a cada estado. Básicamente, un evento puede ser una acción correcta o un error del estudiante por acción u omisión.

Este modelo ha sido integrado dentro de la arquitectura multi-agente para un entorno virtual de entrenamiento con tutoría inteligente denominada MAEVIF (de Antonio, Ramírez, Imbert, & Méndez 2005; Imbert, Sánchez, de Antonio, Méndez, & Ramírez 2007; Méndez Pozo 2008). Esta integración se realizó mediante un nuevo agente llamado “Agente de Comportamiento Colectivo de Estudiantes”, que extiende la arquitectura e incorpora el modelo propuesto.

Esta tesis también describe la aplicación de este modelo en una estrategia de tutoría inteligente dirigida a proporcionar una ayuda lo más adaptada posible a cada alumno. Para definir esta estrategia nos basamos en la teoría pedagógica conocida como andamiaje (*scaffolding*) (Wood, Bruner, & Ross 1976), cuyo principal objetivo es conseguir que el estudiante supere su zona de desarrollo próximo (Vygotsky 1978). Asimismo, la estrategia de tutoría inteligente propuesta también se apoya en el modelo individual del estudiante presentado en Clemente (2011); Clemente et al. (2011).

La estrategia de tutoría presentada presupone que a veces no es buena idea dejar que los estudiantes cometan errores para que así aprendan de ellos. Una de los argumentos que respaldan esta idea es que se corre el riesgo de que el alumno se frustre si comete demasiados errores. Otra razón para impedir los errores sería la de preservar el realismo de la simulación que implementa el entorno de aprendizaje, ya que en un entorno real el alumno no debería cometer ciertos errores fatales.

Así pues, en algunos casos es mejor dar a un estudiante la cantidad mínima de apoyo que necesita para que pueda progresar por sí mismo en la resolución de un problema y así superar su zona de desarrollo próximo. De esta manera, cada estudiante no aprende de sus errores, sino de sus avances.

Si es necesario, el SIT puede incrementar gradualmente el nivel de ayuda cada vez que el estudiante cometa un error o gradualmente reducirla conforme el estudiante va progresando (Olney 2014; Holden & Sinatra 2014).

Por otro lado, la estrategia de tutoría propuesta recurre al modelo colectivo cuando la información disponible en el modelo individual de un estudiante no es lo suficientemente completa como para proporcionar una predicción fiable acerca de su futuro rendimiento.

Además, como parte de este trabajo se ha diseñado una herramienta para visualizar de forma interactiva por medio de un grafo dirigido el autómata asociado con cada clúster del modelo propuesto. Este visualizador resalta las peculiaridades del autómata tales como su división por zonas y diferencia cada tipo de estado con distintos colores.

Esta herramienta ofrece tres posibles vistas: una vista general, en la que se presenta un grafo con las rutas tomadas por todos los estudiantes de un clúster; una vista por estudiante, donde se dibuja el autómata con la solución que ha seguido un solo estudiante;

y una vista por fechas, la cual permite visualizar el modelo de los estudiantes que han realizado el proceso de entrenamiento entre unas fechas determinadas.

El objetivo principal de esta herramienta es permitir la identificación de patrones relevantes en el comportamiento de los estudiantes, como, por ejemplo: errores comunes; tareas que les resultan más difíciles de realizar que otras; tareas que no saben cómo realizar por lo que se estancan en el procedimiento; etc. Esto puede ser de gran ayuda para el profesor, en la medida en que le puede permitir adaptar su docencia para solventar las deficiencias reveladas por los patrones de comportamiento. Asimismo, estos patrones de comportamiento también pueden ser tenidos en cuenta a la hora de diseñar o cambiar la estrategia de tutoría aplicada por un Sistema Inteligente de Tutoría en un entorno virtual de entrenamiento.

A continuación, se presenta la validación de la precisión predictiva de acciones que posee el modelo de comportamiento colectivo de estudiantes. Esta validación, además de verificar la capacidad predictiva del modelo, sirvió para confirmar que el agrupamiento en clústeres ayuda a proporcionar una tutoría adaptada a cada tipo de estudiante.

El resultado de la validación del modelo fue que, aunque los estudiantes tuvieron mucha libertad de acción en el entorno de aprendizaje procedimental empleado en el estudio, el modelo es razonablemente confiable al predecir sus acciones y proporciona una clasificación razonablemente precisa de los estudiantes según su desempeño.

Finalmente, se presenta una segunda validación mediante la cual se comprobó la utilidad del visualizador del modelo propuesto. Como parte de esta validación, se utilizó el visualizador para identificar ciertas zonas críticas para los estudiantes durante el desarrollo de la práctica. Posteriormente, se confirmó que al mejorar la tutoría en estas zonas críticas del proceso, los futuros estudiantes cometieron menos errores. Más concretamente, las frecuencias de errores pedagógicamente relevantes se redujeron considerablemente.

1.4. Estructura del Documento

El presente documento se encuentra dividido en tres partes:

I Planteamiento del problema.

II Solución adoptada.

III Resultados y aportaciones.

La primera parte describe los antecedentes de esta tesis por medio de varios capítulos que detallan dichos antecedentes: los fundamentos teóricos necesarios para el desarrollo

del trabajo (capítulo 2); los antecedentes (capítulo 3); el estado del arte (capítulo 4); y la definición del problema (capítulo 5).

El capítulo “Fundamentos Teóricos” detalla las definiciones teóricas que se necesitan en todo el desarrollo de la tesis, empezando por los fundamentos pedagógicos, y pasando por la definición de los entornos de aprendizaje por computador, sus tipos y características, además de detallar las propiedades de los sistemas inteligentes de tutoría que se pueden integrar con dichos entornos. Finalmente, se describen los conceptos básicos de la minería de datos y se explica cómo se ha aplicado la minería de datos en el campo de la educación.

El capítulo “Antecedentes” presentan los trabajos previos del autor y su grupo de investigación que han servido como punto de partida de esta tesis.

El capítulo “Estado del Arte” presenta los trabajos más relevantes relacionados con el presente trabajo. Brevemente, se describen los principales trabajos relacionados con la minería de datos para la educación, especificando aquellos que utilizan datos para generar recomendaciones, predecir resultados de aprendizaje y modelar a los estudiantes. Además, se presentan trabajos sobre sistemas de capacitación procedimental equipados con SITs, cuyos registros de estudiantes han sido analizados mediante la minería de datos.

El capítulo “Definición del Problema” detalla los objetivos específicos, los supuestos del trabajo, así como los requisitos que debe cumplir el modelo propuesto, la visualización de éste y la estrategia de tutoría que lo utilice.

La segunda parte de este documento define la solución que se ha tomado para poder alcanzar los objetivos antes mencionados. Es por esto que esta parte abarca los tres capítulos medulares de esta tesis: modelo colectivo de estudiantes (capítulo 6), visualizador del modelo (capítulo 7) y estrategia de tutoría (capítulo 8).

El capítulo “Modelo de Comportamiento Colectivo de Estudiantes” define el modelo de comportamiento colectivo de estudiantes, que es el principal aporte del presente trabajo. Se detallan sus características, su arquitectura, su integración con MAEVIF, y los algoritmos de construcción y actualización del modelo a partir de los *logs* de los estudiantes.

El capítulo “Visualización del Modelo de Comportamiento Colectivo de Estudaintes” detalla las características de esta herramienta, así como también la forma en la que se solucionaron las dificultades que se plantearon al momento de interaccionar con un digrafo de gran tamaño. Seguidamente, se muestran varios ejemplos de tareas pedagógicas que podría realizar un profesor con esta herramienta.

Al final de esta segunda parte, el capítulo “Estrategia de Tutoría” especifica la estrategia de tutoría basada en los fundamentos pedagógicos definidos en el capítulo 2. Para ello, se explica el proceso de selección del tipo de tutoría más adecuado para el

INTRODUCCIÓN

estudiante en cada momento teniendo en cuenta la información disponible en el modelo individual del estudiante y el modelo de comportamiento colectivo de estudiantes.

La última parte de este documento detalla los resultados de la validación del modelo (capítulo 9) utilizando los registros de actividad de estudiantes recopilados en las prácticas realizadas con el laboratorio virtual de biotecnología. Y, finalmente, se presentan las conclusiones y las propuestas de trabajos futuros que han surgido de esta investigación (capítulo 10).

Parte I

**PLANTEAMIENTO DEL
PROBLEMA**

CAPÍTULO 2

FUNDAMENTOS TEÓRICOS

En este apartado se dará una visión general de los fundamentos teóricos sobre los cuales se asienta todo el trabajo de esta tesis. Para ello, se va a realizar un recorrido por las principales teorías pedagógicas, su aplicación en el mundo de la informática mediante entornos de aprendizaje por computador; así como se van a abordar algunos conceptos y técnicas de minería de datos de interés para esta investigación.

2.1. Fundamentos Pedagógicos

La epistemología ha sido definida por la Real Academia de la Lengua Española como la “teoría de los fundamentos y métodos del conocimiento científico”. No obstante, esta versión lingüística de la definición deja fuera el hecho de que el campo de aplicación de la epistemología va más allá del mero hecho científico y trasciende a lo filosófico; en este sentido, Bunge (1980) la define como: “una rama de la filosofía que estudia la investigación científica y su producto, el conocimiento científico”.

Según la Real Academia de la Lengua Española, la pedagogía es “la ciencia que se ocupa de la educación y la enseñanza”. En consecuencia, se puede afirmar que uno de los objetivos de la pedagogía es organizar los procesos de aprendizaje de manera que los estudiantes aprendan lo máximo posible. Dentro de las teorías del conocimiento existen varias corrientes pedagógicas, entre ellas destaca la corriente constructivista, basada en la doctrina filosófica con el mismo nombre. Esta rama del pensamiento defiende que el conocimiento no se genera de forma ad-hoc sino que es el resultado de un conjunto de construcciones mentales creadas con el propósito de explicar experiencias, las cuales han sido obtenidas por medio de los sentidos (Watzlawick 1984), es decir, son de origen experiencial, y responden también a razones empíricas y lógicas al mismo tiempo.

El constructivismo se fundamenta en las siguientes tres visiones del aprendizaje, los cuales coinciden en criticar las teorías racionalistas que conciben la construcción del conocimiento únicamente desde el razonamiento lógico. Estas tres visiones que se describirán a continuación muestran la importancia de la experiencia a la hora de adquirir el conocimiento.

En primer lugar, el filósofo Kant dice que cada persona forma su propia representación del conocimiento, por lo que existirán distintas versiones del conocimiento válido, es decir, la subjetividad jugará un papel fundamental a la hora de defender este conocimiento (Von Glasersfeld 1984).

En una línea similar, Piaget, Inhelder, y Chipman (1976) en su teoría del desarrollo cognitivo, señala que, durante la exploración activa del dominio del conocimiento, el aprendizaje se produce cuando el alumno descubre una deficiencia en su conocimiento o una inconsistencia entre su representación del conocimiento actual y su experiencia.

Por otro lado, Vygotsky (1978) sostiene que el aprendizaje se produce dentro de un contexto social, y que la interacción entre los alumnos y sus compañeros es una parte necesaria del proceso de aprendizaje (Luria, Leontiev, & Vigotsky 2004). Por tanto, el entorno familiar, social y cultural son elementos claves no solo para la adopción de este conocimiento, sino para la aplicación de éste en la vida real. Por ejemplo, estudios provenientes de la Organisation for Economic Co-operation and Development (2014) (OECD) no miden al conocimiento en sí mismo, sino la forma en la que los alumnos resuelven los problemas y adquieren esos conocimientos, tomando en cuenta distintas variables socioculturales.

El constructivismo puede ser interpretado de tres diferentes formas (Moshman 1982): endógena, exógena y dialéctica. La interpretación endógena propone que el profesor proporcione sus conocimientos al alumno mientras ellos lo asumen mediante un proceso interno propio. Por otro lado, el constructivismo exógeno ayuda al estudiante en su desarrollo cognitivo mediante ejercicios y prácticas, el profesor se centra en transmitir información enseñando estrategias y corrigiendo errores. Por último, el constructivismo dialéctico es en el cual los alumnos aprenden por medio de sus experiencias y errores, y en el que los profesores pueden asistir a los estudiantes por medio de técnicas como el *andamiaje* (*scaffolding*).

A continuación, vamos a abordar con más detalle las teorías de Piaget y Vygotsky en las que se apoya el constructivismo.

2.1.1. Teoría del desarrollo cognitivo

La teoría del desarrollo cognitivo desarrollada por Piaget et al. (1976) es parte de la psicología evolutiva del siglo XX que pretende entender todas las facetas del proceso cognitivo humano. Dicha teoría persigue resolver el problema del conocimiento y de su origen, para ello Piaget investigó mediante diversos experimentos cómo los seres humanos adquirimos conocimientos y cómo pasamos de estados de conocimiento de menor validez a estados de conocimiento de mayor validez. Para el desarrollo de esta teoría Piaget realizó un análisis genético a varios niños y adolescentes (Piaget 1969; Piaget & Duckworth 1970) para detectar la adopción del conocimiento y el proceso de transformación del mismo.

Por influencia de su formación en biología, Piaget entiende la inteligencia humana como un elemento que facilita la adaptación al medio, equivalente a las funciones adaptativas de otros organismos vivos. Esta visión se acerca a teorías de carácter biológico evolutivo. Es decir, su teoría intenta explicar las diferentes formas o estructuras del pensamiento de la misma manera que otras teorías biológicas explican cómo las diversas estructuras orgánicas ayudan a la adaptación de un organismo a su medio.

Según Piaget, el niño trata de comprender al mundo que le rodea. Para esto intenta construir conocimientos acerca de los objetos y las personas a través de una constante interacción en la que interfieren las tres partes (él, los objetos y las personas que lo rodean). Piaget cree que el conocimiento está vinculado a la *acción*, es decir, el conocimiento representa el resultado de la interacción entre sujeto y objeto.

La forma en la que el niño se adapta a su entorno se explica mediante dos procesos básicos: asimilación y acomodación. En la asimilación, el niño adapta la nueva información obtenida para que encaje con lo que ya conoce, mientras que en la acomodación el niño modifica lo que ya conoce para dar sentido a la nueva información. Esto quiere decir que si la nueva información que obtiene el niño es compatible con los esquemas de conocimiento que ya posee la asimila dentro de ellos y alcanza un estado de equilibrio. En cambio, si no es compatible, tendrá que modificar los esquemas de conocimiento ya obtenidos para que esta nueva información se acomode a éstos. Ambos procesos están estrechamente correlacionados y explican el modelo cambiante del conocimiento a lo largo de la vida de los sujetos.

Según Piaget, en el desarrollo cognitivo intervienen cuatro factores que influyen en el cambio de este tipo de estructura en el niño, que son: la maduración biológica, la influencia del medio social, la experiencia y el equilibrio.

Con respecto a la experiencia, ésta puede ser de tres tipos: el primer tipo es obtenido por medio de la repetición de una acción varias veces; el segundo tipo se refiere a la experiencia física con el ambiente; y el último tipo es la experiencia lógico-matemática.

Por otro lado, el equilibrio permite a los sujetos organizar los otros tres factores (maduración biológica, influencia del medio social y experiencia) en un conjunto lógico que mantenga un equilibrio de sus estructuras cognitivas. Piaget pensaba que el desequilibrio es un estado insatisfactorio para el ser humano, por ello los sujetos modifican las estructuras cognitivas con tal de encontrar un equilibrio.

Piaget divide el desarrollo cognitivo del niño en cuatro etapas: etapa sensoriomotora, etapa pre-operacional, etapa de operaciones concretas y etapa de operaciones formales.

La primera etapa se da desde el nacimiento hasta los dos años de vida, y se refiere al período en el que el bebé aprende a responder por medio de reflejos o acciones motoras a estímulos recibidos a través de sus sentidos. Seguidamente, la pre-operacional se da entre los dos y los siete años, y a esta edad el niño es capaz de representar los objetos o personas de forma simbólica, es decir, son capaces de imaginar y representar objetos. A diferencia de las anteriores, en las dos últimas etapas interviene la educación escolar. La fase de operaciones concretas se da entre los siete y once años, y es en ésta en la que el niño aprende a utilizar operaciones mentales y la lógica para sacar conclusiones. Finalmente, en la etapa operaciones formales, entre los once y doce años, la persona empieza ya a pensar de forma abstracta.

2.1.2. Teoría del aprendizaje sociocultural

Esta teoría nace de la crítica a los métodos cognitivistas y naturalistas (Vygotsky 1987), ya que estos métodos no explicaban de una forma satisfactoria los procesos de adquisición del conocimiento, al apoyarse únicamente en métodos basados en las ciencias naturales y en cuestiones observables, y no en aspectos psicológicos (Lucci 2006).

Vygotsky (1978) cree que, para entender el desarrollo cognitivo de una persona, es fundamental primero comprender el entorno social, histórico y político en el que se desenvuelven los sujetos, ya que el individuo está en constante interacción con su entorno. Esta teoría coincide con la de Piaget en que el aprendizaje se genera gracias a la interacción del sujeto con el ambiente. Sin embargo, Vygotsky considera que este ambiente está compuesto por objetos y personas, no sólo por objetos como detalla Piaget. De esta forma esta visión sociocultural del aprendizaje concibe al hombre más como una construcción social que biológica.

Dentro de esta visión, el lenguaje es considerado como una herramienta fundamental para conocer las condiciones socioculturales del alumno, además de ser esencial para el intercambio de ideas entre estudiantes. Tomando en cuenta esto, Vygotsky (1978) sostiene que el estudio colaborativo es muy beneficioso en el desarrollo del aprendizaje.

“El aprendizaje despierta una variedad de procesos de desarrollo que son capaces de operar únicamente cuando el niño interactúa con otras personas y en colaboración con sus compañeros”

Vygotsky (1978)

Al tomar en cuenta el entorno social del niño, Vygotsky (1978) defiende que la interacción con un mediador humano o instructor (profesor, padres, etc.) es fundamental en su aprendizaje. De esta forma, Vygotsky (1978) propone el “modelo de mediación del aprendizaje”, el cual inserta el concepto del mediador dentro de la teoría inicial de estímulo-respuesta de Piaget. Es decir, cambia la percepción de que los comportamientos se producen independientemente del contexto únicamente marcado por el incentivo directo de un estímulo, de tal manera que entre el estímulo (S) y la respuesta (R) se encuentra un complejo proceso de mediación (figura 2.1). Esta idea de mediación se expresa como una estrecha relación entre el sujeto, el objeto y los artefactos mediadores (figura 2.2)

Figura 2.1 Modelo de mediación del aprendizaje de Vygotsky (1978).

Figura 2.2 Tríada de sujeto, objeto y artefactos mediadores.

Como parte de este modelo de mediación en el aprendizaje, Vygotsky (1978) formuló otro concepto llamado **zona del desarrollo próximo (ZDP)**, que se explica en la siguiente sección.

2.1.3. Zona del desarrollo próximo

Es la zona del conocimiento que se encuentra entre lo que actualmente conoce el alumno y lo que no conoce, es decir, la zona que tiene que cruzar el estudiante, con un poco de ayuda, para llegar a su objetivo de aprendizaje.

Vygotsky (1978) la define como “la distancia entre el nivel actual de desarrollo, determinado por la resolución de problemas de forma independiente, y el nivel de desarrollo potencial, determinado por la resolución de problemas bajo la guía de un adulto o en colaboración con compañeros más competentes”.

Figura 2.3 Zona del Desarrollo Próximo.

Supongamos que tenemos dos estudiantes con “edad mental” similar, entonces si a sólo uno de ellos se le otorga un poco de ayuda para superar su ZDP, tendrá más posibilidades de llegar a una “edad mental” superior. Es decir, aunque las competencias iniciales de ambos estudiantes sean similares, las futuras potencialidades de cada uno no lo son.

Es necesario clarificar el concepto de “edad mental” utilizado por Vygotsky (1984), el cual es el resultado de una serie de tests realizados a varios individuos con la misma edad física, pero en el que según el resultado de estas pruebas, los individuos pueden tener una “edad mental” diferente según las destrezas demostradas.

Además, se puede decir que “lo que el estudiante puede hacer hoy con ayuda, lo podrá hacer mañana por sí solo” (Vygotsky 1978). Por ejemplo, supongamos que un niño sabe conducir una bicicleta con ruedas auxiliares, pero no sabe conducir una de montaña. Entonces cabe esperar que con ayuda de un instructor que le sostenga la silla puede llegar a aprender a conducir sin las ruedas auxiliares y finalmente será capaz de conducir una bicicleta de montaña (figura 2.4).

Esta teoría critica la enseñanza tradicional (Freire 2005), que sostiene que el profesor no puede enseñar nada más allá del nivel cognitivo del alumno. Además, critica también la creencia según la cual, se considera al estudiante un “ignorante” y al profesor un “sabio”, y por lo tanto, sólo podía existir un único conocimiento válido, el del profesor.

Figura 2.4 Ejemplo de una ZDP.

2.1.4. Teoría de la actividad

La teoría de la actividad se centra en los procesos de aprendizaje mediante actividades, como su propio nombre indica. Se formuló como parte del modelo de mediación propuesto por Vygotsky (1978) y desarrollado en años posteriores por su discípulo Leontiev (1981).

Leontiev (1981) propone una descomposición jerárquica para una actividad en acciones y éstas, a su vez, en operaciones. Esto quiere decir que el comportamiento humano puede ser dividido en estos tres niveles. En paralelo, esta división (Marsh 2003) está asociada con objetivos, metas y condiciones respectivamente (figura 2.5).

Figura 2.5 Descomposición de la actividad (Marsh 2003; de Antonio & Troncoso 2008).

Una actividad es el conjunto de acciones que siguen un proceso determinado para lograr un objetivo. Esto implica que las actividades se basen en motivaciones o necesidades. Por ejemplo, una actividad sería escribir la memoria de la tesis doctoral y la motivación obtener el grado de doctorado.

Las acciones son los pasos individuales que son necesarios para completar una actividad. Al contrario que en una actividad, la motivación no es importante para realizar una

acción. Sin embargo, cada acción es necesaria para cumplir una necesidad. Siguiendo con el ejemplo de actividad anterior, algunos ejemplos de acciones son: planificar los capítulos y su contenido, y buscar bibliografía que fundamente la tesis.

Las operaciones constituyen el nivel de abstracción más bajo de esta teoría, y representan las ejecuciones de una acción. Éstas no tienen metas y normalmente se ejecutan de forma secuencial y automática. Por ejemplo, operaciones para buscar bibliografía pueden ser: seleccionar los conceptos que necesitan fundamentación, seleccionar las fuentes de acceso a bibliografía especializada, buscar el concepto deseado, etc. Las operaciones están ligadas a las condiciones necesarias para la realización de cada acción, por ejemplo: la selección de conceptos de los cuales se va a buscar bibliografía están relacionada con la temática de la que trata la tesis.

2.1.5. Tipos de problemas

Existe un sinnúmero de tipos de problemas, lo cuales requieren diferente solución (*problem-solving*), como, por ejemplo, los relacionados con tareas de razonamiento analógico (Robins & Mayer 1993), argumentativo (Robins & Mayer 1993), científico (Robins & Mayer 1993), entre otros. Estos estudios revelan que una efectiva resolución del problema está estrechamente relacionada con el dominio.

Sin embargo, la habilidad para resolver problemas depende no solamente de la forma en la que está estructurado el dominio del conocimiento, sino también de la estructura del problema.

Según Kitchner (1983), los problemas pueden ser clasificados según su estructura en dos tipos: bien definidos (*well-defined*) y mal definidos (*ill-defined*). Los problemas bien definidos son aquellos en los cuales existe una o varias soluciones las cuales son conocidas y correctas. Los problemas mal definidos son aquellos cuyas soluciones pueden ser diferentes dependiendo de los supuestos, evidencias u opiniones que se tome.

Es decir, en los problemas bien definidos existe una solución correcta que puede ser alcanzada con un procedimiento claramente definido, mientras que los problemas mal definidos pueden tener múltiples soluciones o ninguna, y no poseen un procedimiento que garantice alcanzar la meta deseada; es así que estos tipos de problemas requieran procesos cognitivos separados (Schraw, Dunkle, & Bendixen 1995).

2.1.6. Andamiaje

El **andamiaje** o *scaffolding* (Wood et al. 1976) es explicado normalmente en términos del ZDP como una herramienta para ayudar al alumno a cruzar dicha zona. Esta estrategia

de enseñanza destaca el papel del profesor como un tutor, el cual guía al alumno al resolver un problema. En este sentido, el tutor controla los elementos de la tarea que están inicialmente fuera de las capacidades del alumno, permitiéndole así concentrarse únicamente en aquellos elementos que están dentro de su rango de competencia (Wood et al. 1976). Asimismo, el tutor facilita las estrategias y procesos necesarios para conseguir un aprendizaje eficaz, motiva al alumno para que alcance su objetivo, así como provee modelos, otorga consejos y responde preguntas del estudiante.

En la propuesta original concebida por Wood et al. (1976), se definen las siguientes funciones del tutor:

- Ganar el interés del estudiante y promover un compromiso de éste con su objetivo de aprendizaje.
- Disminuir la dificultad de la tarea hasta un nivel apropiado para el estudiante.
- Mantener la atención del alumno en la tarea actual proveyéndole apoyo afectivo y motivacional.
- Destacar las características relevantes de la tarea, por ejemplo, resaltando los resultados incorrectos.
- Controlar la frustración del estudiante.
- Demostrar los caminos hacia la solución ideal basándose en el intento actual del alumno.

A primera vista, se puede apreciar que el término *andamiaje* es una metáfora del levantamiento de andamios en la construcción de edificaciones, y originalmente fue pensado de esa forma, debido a que la idea nació del estudio original de Wood et al. (1976) donde la tarea que tenían que realizar los niños era construir una pirámide con bloques de madera. Ahora bien, el andamiaje ya no se refiere a una metáfora de construcción, puesto que ha sido redefinida en términos de una teoría para la optimización del aprendizaje durante el proceso de tutoría (Olney 2014).

Pol, Volman, y Beishuizen (2010) encontraron tres características del proceso del andamiaje: contingencia, desvanecimiento (*fading*) y transferencia de responsabilidad. Se denomina contingencia a la adaptación del soporte o ayuda al ZDP de cada estudiante. El desvanecimiento es la reducción gradual de esta ayuda debido a que el alumno está aprendiendo. Y la transferencia de responsabilidad se da gracias a que la asistencia va desvaneciéndose, el estudiante puede ir asumiendo un mayor rol dentro de su aprendizaje o, en otras palabras, éste puede responsabilizarse más de su tarea.

Previo a iniciar el proceso es importante conocer qué se tiene que andamiar. Para esto, se deben evaluar los conocimientos previos del alumno, así como el nivel de desarrollo y la experiencia en el campo. Por ejemplo, estudiantes con bajo nivel cognitivo del área de conocimiento a tratar pueden requerir tanto un proceso de andamiaje acompañado de una exposición de ciertos contenidos del dominio, mientras que estudiantes con un alto conocimiento en la materia pueden necesitar únicamente un proceso de andamiaje (Azevedo, Moos, Greene, Winters, & Cromley 2007).

Cuando un profesor ideal utiliza estrategias de andamiaje en su enseñanza, no ayuda al estudiante simplemente dándole la respuesta a su pregunta, sino que le proporciona herramientas o pistas que le ayudan a encontrar por sí mismo la respuesta. Por ejemplo, si el estudiante no sabe cómo encontrar el valor de un lado de un triángulo teniendo los valores de los otros dos, el profesor podría darle una pista indicándole que recuerde cómo se calcula el perímetro de un triángulo en vez de darle la respuesta final con el valor de ese lado. Otro ejemplo, sería el facilitar a un estudiante una tabla en la que pueda categorizar los conceptos importantes de un texto que debe leer y comentar.

Trabajos posteriores a la primera formulación del andamiaje, lo definen en términos de tutoría contingente (Wood 2003). La idea principal es que trabajando colaborativamente con el alumno el tutor puede encontrar el nivel apropiado en el que éste tenga éxito en el siguiente paso de la tarea, es decir, la guía del tutor tiene que ser cuidadosamente adaptada a las necesidades actuales del estudiante. Wood (2003) propone tres dimensiones para la tutoría contingente:

- Contingencia instruccional - formas de dar apoyo a la tarea. Se centra en cómo el tutor ajusta la ayuda ofrecida al alumno. Es decir, el tutor debe recordar cuánta ayuda se ha dado anteriormente al estudiante, y aumentar o disminuir ésta según el desempeño del mismo.
- Contingencia de dominio - en qué centrarse después. Para decidir en qué deben trabajar los estudiantes a continuación, hay que controlar la cantidad de incertidumbre podrían soportar los alumnos. Si tienen mucha incertidumbre, o muchas formas de equivocarse, entonces se puede dar la ayuda que se desee. Pero, si se encuentran abrumados y no saben cómo utilizar la ayuda, el tutor debe proveerles bases sólidas para que la siguiente tarea sea exitosa.
- Contingencia temporal - si hay que intervenir y cuándo. El tutor debe saber de antemano cuánto tiempo esperar o cuántos intentos permitir antes de intervenir con ayuda.

La mayoría de las técnicas de andamiaje se pueden clasificar en *suaves* o *duras* (Simons & Klein 2007). Los andamios suaves se refieren a la ayuda dinámica que puede otorgar un profesor, por ejemplo, un maestro que circula alrededor del aula de clase proveyendo asistencia a los alumnos en sus tareas. En cambio, un andamio duro es el soporte estático que el profesor desarrolla de antemano sabiendo que la tarea va a ser difícil para el alumno, por ejemplo, una herramienta que le ayude al alumno a encontrar la ecuación adecuada para el cálculo del valor del lado de un triángulo.

2.1.7. Aprender haciendo

El concepto “aprender haciendo” (*learning by doing*) fue acuñado por Arrow (1962) en su teoría económica, la cual propone que el crecimiento de la productividad está asociado a la acumulación de experiencia en la producción de un producto Arrow (1962), es decir, la empresa se hace más eficiente mientras más produce. Además, propone una generalización de esta idea que puede ser aceptada tanto por psicólogos como por economistas. Ésta dice que “el aprendizaje es el producto de la experiencia” y que “el aprendizaje sólo puede tomar lugar al intentar resolver un problema y por lo tanto sólo se produce durante una actividad”.

Los cursos de aprendizaje que siguen la filosofía del “aprender haciendo” fomentan el interés y generan una mayor motivación en el estudiante gracias a que captan su atención más que la educación tradicional (Brookfield 1988; Lewin 1951).

Figura 2.6 Cono de la experiencia de Dale (1969).

Esta filosofía de aprendizaje queda reflejada como la de más alto nivel instruccional en el “Cono de la Experiencia” de Dale (1969). Este modelo representa la relación de las tasas de retención de lo aprendido con las formas de obtener la información.

Como se puede apreciar en la figura 2.6, la mayor tasa de retención se produce mediante la experiencia propia, es decir, cuando el sujeto realiza una tarea a aprender por sí mismo ya sea en un entorno real o simulado.

2.1.8. Aprendizaje autorregulado

En el aprendizaje autorregulado el estudiante se esfuerza por monitorizar y tomar control de su propio proceso de aprendizaje (Zimmerman 1990). Los alumnos que siguen este proceso, utilizan métodos metacognitivos, acciones y comportamientos estratégicos, así como componentes motivacionales; de tal manera que pueden fijarse objetivos, monitorizar su progreso hacia esos objetivos y regular su cognición, motivación y comportamiento para llegar a ellos.

Este tipo de aprendizaje se compone de tres fases que interaccionan cíclicamente entre ellas: fase de reflexión previa, fase de ejecución y fase de autorreflexión (Zimmerman & Campillo 2003).

En la primera fase se fijan los objetivos y se planifica el proceso de aprendizaje. Mientras que, en la fase de ejecución, el estudiante realiza tareas de control y observación para optimizar su rendimiento. Finalmente, el alumno reflexiona sobre su rendimiento general; esta reflexión puede motivarle a modificar su estrategia de aprendizaje o sus objetivos y volver a empezar.

A través de este modelo, se ha llegado a la conclusión que el alumno ideal, además de poseer las habilidades cognitivas para aprender algo nuevo, tiene también la conciencia metacognitiva de qué comportamientos o estrategias emplear, conoce qué recursos posee y cuándo es mejor utilizarlos, y también sabe cómo motivarse para llegar a su objetivo (Bransford et al. 2005).

2.2. Entornos de Aprendizaje por Computador

Inspirados en las teorías constructivistas de aprendizaje, nacieron los **entornos de aprendizaje por computador** o *computer based learning environment* (CBLE), también llamados sistemas de *e-learning*. Estos entornos son sistemas computacionales que ayudan a los estudiantes en su proceso de aprendizaje.

Existe una diversidad de CBLE (Lajoie & Derry 1993), como los entornos hipermedia, los cuales presentan información al estudiante por medio de audio, imágenes, vídeo o texto. Otro tipo de CBLE son los sistemas de enseñanza por medio de la web, entre estos, los más usados actualmente son los *Learning Management Systems (LMS)* como *Moodle*, o también los *Massive open online courses (MOOC)* como *Coursera*.

Por otro lado, cabe destacar el entrenamiento por simuladores, los cuales son un tipo de entornos que presentan realidades simuladas gráficamente. Además, permiten al estudiante realizar tareas similares a las que se pueden realizar en la vida real, lo que ayuda al ahorro de costos y a evitar accidentes. Dentro de éstos existen dos tipos, los simuladores de entrenamiento y los *serious games* o *game-based learning systems*.

Los simuladores de entrenamiento reproducen una tarea que se puede realizar en la vida real. Ahora bien, aunque muchos de estos se desarrollen en entornos gráficos 2D o 3D, no significa que sean videojuegos. Es en esta característica en la que se diferencian de los *serious games*, ya que éstos no necesariamente reproducen escenarios reales como sí hacen los simuladores de entrenamiento, aunque ambos tipos de entornos no son creados solamente para el entretenimiento (Narayanasamy, Wong, Fung, & Rai 2006).

Los *serious games* pretenden que el estudiante aprenda a partir del entretenimiento otorgado por un videojuego (Michael & Chen 2005), por ejemplo, *Crystal Island* (Rowe et al. 2009), el cual reproduce una isla en la que se ha propagado una epidemia y los estudiantes tienen que llevar a cabo varias misiones científicas para salvar a la población.

Muchos de estos sistemas de educación utilizan sistemas inteligentes de tutoría que guían al alumno en la realización de la tarea, proporcionan soporte para lograr un objetivo de aprendizaje o brindan mensajes de ayuda cuando el estudiante lo necesita. Este tipo de sistemas son tratados en la sección 2.2.2.

Un tipo de simuladores de entrenamiento son los entornos virtuales tridimensionales (3D), que en ocasiones incorporan tecnologías de realidad virtual.

2.2.1. Entornos virtuales 3D

En términos generales, un entorno virtual es cualquier espacio interactivo al que se accede por medio de un computador. El ejemplo más moderno son las redes sociales, los cuales son entornos virtuales web interactivos y colaborativos.

Un entorno virtual 3D es aquel que además de ser interactivo y accesible por medio de un computador, presenta gráficos tridimensionales, como algunos videojuegos. Dentro de estos entornos se puede encontrar a los entornos de realidad virtual, que utilizan tecnologías inmersivas, tales como un casco o un guante, para mejorar la experiencia del usuario.

El concepto de realidad virtual tiene un poco más de medio siglo, pese a que existe la percepción de que es un concepto moderno, gracias a la reciente aparición en el mercado de cascos virtuales como el Oculus Rift con un coste al alcance de muchos usuarios. Se considera a Sensorama, patentado por Heilig en 1962, como el primer entorno de realidad virtual (Gigante 1993).

En la literatura existen varias definiciones de realidad virtual como:

“Realidad virtual es el conjunto de simulaciones electrónicas de entornos expertos que permiten al usuario final interactuar en situaciones tridimensionales realistas mediante dispositivos montados en la cabeza y ropa cableada.”
Coates (1992)

“La Realidad Virtual es un mundo alternativo lleno de imágenes generadas por ordenador que responden a los movimientos humanos. Estos ambientes simulados se visitan generalmente con la ayuda de un costoso traje que cuenta con gafas de vídeo estereofónicas y guantes de datos de fibra óptica.”
Greenbaum (1992)

Sin embargo, Steuer (1992), comenta que estas definiciones están centradas en el hardware y por ello propone que el concepto de realidad virtual debe ser replanteado enfocándose en la experiencia del usuario, para lo que define el concepto de telepresencia.

Actualmente se pueden diferenciar los siguientes tipos de entornos virtuales (Saracevic 2014):

- Espacios compartidos. Se caracterizan por estar poblados por usuarios geográficamente distribuidos.
- Entornos basados en texto. Como *Multi-User Dungeons* (MUD) o entornos multiusuarios orientados a objetos (MOO – MUD Orientado a Objetos).
- Chats multiusuario (MUC).
- *Massive Multiplayer Online Role Playing Game* (MMORPG).
- Entornos interactivos multiusuario 2D y 3D.
- Hábitats virtuales. Los entornos más comunes de este tipo son los mundos virtuales.

Los entornos virtuales 3D pueden ser simuladores donde interviene un solo usuario o entornos multiusuarios conocidos como mundos virtuales. Estos últimos, según

Papadopoulos, Pentzou, Louloudiadis, y Tsiatsos (2013), se definen como “una red síncrona y persistente de personas, representadas por un avatar, facilitada por una red de computadores”.

Los primeros mundos virtuales no eran 3D. El primero de estos entornos fue *Multi-User Dungeons* (Bartle 2004), a partir de éste, los mundo virtuales han evolucionado gracias a la industria de los videojuegos. Entre las plataformas para desarrollar mundos virtuales 3D más populares se encuentran SecondLife, Opensimulator o Active Worlds.

Además de las aplicaciones lúdicas de los entornos virtuales 3D, con los años estos entornos han ido ganándose un espacio entre los CBLEs, ya que desde un punto de vista constructivista existe buenas razones para utilizar estos entornos con fines educativos (Dalgarno 2002). Actualmente, los entornos virtuales 3D son utilizados tanto en la educación informal como en la formal (Livingstone, Kemp, & Edgar 2008). La educación formal es aquella que está regida por normas y leyes, y es impartida dentro de instituciones oficiales; mientras que la informal, es aquella que se adquiere fuera de estas instituciones, como, por ejemplo, en museos (Dib & Barojas 1988).

Dalgarno y Lee (2010) agrupan las aplicaciones educativas de los entornos virtuales 3D en tres grupos: simulaciones 3D y micromundos; entornos 3D como interfaces para la enseñanza de recursos; y entornos virtuales multiusuario 3D de enseñanza. A continuación, vamos a abordar las características de cada uno de estos grupos.

Simulaciones 3D y micromundos

Los simuladores 3D son quizá el tipo de entorno tridimensional más utilizado para entrenamientos, y se caracterizan por modelar tridimensionalmente lugares y objetos reales. En cambio, los micromundos son entornos de aprendizaje exploratorio, que simulan entornos abstractos para la enseñanza de conceptos Rieber (1992; 2005). Un ejemplo de micromundo es LOGO (Papert 1980).

El mayor beneficio pedagógico de las simulaciones 3D tiene que ver con el tipo de interacción que puede experimentar el estudiante con los objetos, en la medida en que la tecnología permite simular el comportamiento físico de los objetos necesarios en un entrenamiento en 3D, en lugar de restringir el movimiento y el comportamiento a dos dimensiones. De una forma muy similar, los micromundos ayudan al estudiante a entender los conceptos mediante el aprovechamiento de una representación espacial del dominio de conocimiento.

El hecho de que los estudiantes puedan practicar habilidades o realizar tareas en un entorno simulado, evita los riesgos que dichas tareas pueden tener en la realidad (cortarse,

quemarse, contagiarse, etc.) y elimina los costos del material necesario para realizar las tareas en el mundo real.

La personalización que ofrecen las simulaciones 3D y los micromundos es potencialmente motivadora y atrayente para los alumnos, ya que crea un entorno en el que puedan tomar sus propias decisiones, y por tanto así pueden alcanzar sus propios objetivos de aprendizaje.

Entornos 3D como interfaces para el acceso a recursos de aprendizaje

Existen estudios que manifiestan que muchos estudiantes tienen problemas al momento de acceder a recursos educativos en los sistemas hipermedia de enseñanza, fenómeno al que denominan “perdidos en el hiperespacio” (McKnight, Dillon, & Richardson 1991). Los entornos 3D, al proporcionar una sensación de inmersión, permiten una experiencia no simbólica en primera persona, al contrario que en la mayoría de los demás sistemas de información que presentan una experiencia simbólica en tercera persona (Winn 1993). Este tipo de experiencia de los entornos 3D ofrece a los estudiantes una navegación intuitiva a través de los recursos.

Además, como se ha visto anteriormente, según la teoría constructivista y el cono de la experiencia no hay mejor forma de aprender que por medio de la experiencia directa. Por lo cual no existe mejor recurso de aprendizaje que el modelado en un contexto lo más parecido posible al contexto real en el cual el conocimiento va a ser aplicado.

Por ejemplo, Boulos, Hetherington, y Wheeler (2007) crearon un entorno virtual en el cual los estudiantes pueden aprender la forma correcta de realizar una auscultación cardíaca mientras escuchan sonidos después de tocar objetos 3D. Este entorno, además, muestra el impacto de la comida rápida en la salud a través del uso de objetos interactivos.

Entornos virtuales multiusuario 3D de enseñanza

Al igual que los mencionados anteriormente, estos sistemas aportan una mejor experiencia al momento de explorar el entorno educativo que los sistemas tradicionales de educación, si bien los entornos multiusuario agregan a esto la interacción con otros estudiantes.

Esta interacción con otros individuos es la ventaja más importante de este tipo de entornos, ya que no solo les permite comunicarse, sino también practicar otras formas de interacción como realizar tareas conjuntamente, lo cual permite llevar a cabo una enseñanza colaborativa más efectiva que en los entornos educativos 2D.

Dalgarno y Lee (2010), basándose en las ventajas educativas mencionadas anteriormente, concluyen que los entornos 3D de entrenamiento poseen las siguientes cinco potencialidades:

1. Pueden facilitar tareas de enseñanza que conducen al desarrollo de una mejor representación del conocimiento espacial del dominio explorado.
2. Pueden usarse para facilitar tareas de aprendizaje experimental que serían poco prácticas o imposibles de realizar en el mundo real.
3. Pueden ser usados para tareas de aprendizaje que incrementen la motivación intrínseca y el compromiso de los estudiantes.
4. Pueden ser utilizados para facilitar tareas de enseñanza que mejoren la transferencia de conocimiento y habilidades a situaciones reales a través de la contextualización del aprendizaje.
5. Pueden ser usados para facilitar tareas que conlleven un aprendizaje colaborativo más efectivo que el que es posible con sistemas de educación 2D.

2.2.2. Sistemas inteligentes de tutoría

El artículo publicado por Carbonell (1970) es considerado el primer trabajo acerca de un **Sistema Inteligente de Tutoría** (SIT), en él se detalla un sistema llamado Scholar, el cual es un conjunto de programas que utilizan una representación semántica del conocimiento del estudiante para crear un diálogo con él. Pero no es hasta 1982 cuando Sleeman y Brown describe con el término *Intelligent Tutoring Systems* a los sistemas capaces de dar este tipo de instrucción, y así los diferencia de los demás CBLEs.

Murray (1999) considera a los SITs como sistemas instruccionales basados en computador que utilizan técnicas de inteligencia artificial para definir modelos de contenido (tema a enseñar), es decir, especifican “qué” enseñar en cada momento y “cómo” enseñarlo mediante estrategias de enseñanza.

Wenger (1987) propone la que actualmente se conoce como arquitectura clásica de un SIT (figura 2.7). Esta arquitectura está conformada por cuatro módulos: comunicación, experto, estudiante y de tutoría.

El módulo de comunicación es simplemente la interfaz entre el SIT y CBLE, es decir, es el encargado de recibir las peticiones de tutoría que el CBLE cree pertinentes y las envía al tutor automático, posteriormente devuelve la respuesta del SIT. El módulo de tutoría es el núcleo del SIT y posee la estrategia de tutoría. El módulo del estudiante

Figura 2.7 Arquitectura clásica de un SIT.

es el que contiene la información acerca del alumno, especialmente sus datos cognitivos, lo cual permite adaptar la tutoría a las necesidades de cada alumno. Finalmente, el módulo experto es aquél que contiene la información del dominio de conocimiento, es decir, los temas que se desean enseñar o el procedimiento de la tarea que el estudiante debe aprender a realizar.

Andamiaje en sistemas inteligentes de tutoría

El andamiaje fue pensado originalmente para enseñanza supervisada por un tutor humano, pero se ha modificado para que pueda ser aplicada en entornos virtuales de aprendizaje supervisados por SITs. Yelland y Masters (2007) propone una nueva categoría de andamiaje a la que llama “andamiaje técnico” (*“technical scaffolding”*), a la cual define como el uso de un computador como medio para realizar el andamiaje.

Jackson, Krajcik, y Soloway (1998) exponen que “implementar software educativo con andamiaje ofrece la oportunidad de otorgar un apoyo individualizado adaptado a cada estudiante teniendo en cuenta sus habilidades y su entorno; así como también facilita el crecimiento del estudiante adaptando el funcionamiento del sistema conforme el estudiante va desarrollando sus destrezas.”

En una enseñanza tradicional, un tutor humano puede darse cuenta intuitivamente del nivel de aprendizaje alcanzado por un alumno, mientras que un SIT puede inferir esto a partir de la información previamente obtenida del estudiante (Holden & Sinatra 2014). Una vez determinado dicho nivel, el tutor automático puede seleccionar el tipo de guía y la realimentación más adecuada para el alumno en el momento actual.

Saber cuándo desvanecer la instrucción es un reto inherente al andamiaje con un tutor humano, pero es incluso más difícil en una tutoría computarizada. En este tipo de tutoría es importante calibrar adecuadamente el nivel de andamiaje para cada estudiante,

ya que, si es excesivo, el problema será demasiado sencillo y el estudiante perderá interés. Por el contrario, si el nivel de andamiaje es insuficiente, el estudiante podría desanimarse (Lajoie 2005).

Otro reto al que deben enfrentarse los tutores automáticos es el de poder reconocer los cambios de estado anímico del estudiante, con el fin de poder cambiar la estrategia de tutoría, para, por ejemplo, atraer nuevamente la atención del alumno. En el caso de un tutor humano estos cambios anímicos los detecta intuitivamente el propio tutor. En cambio, en el caso de un tutor automático, si se diseña bien este tutor, los podría detectar con mayor precisión que un tutor humano (D'mello & Graesser 2012).

Aprendizaje autorregulado en sistemas inteligentes de tutoría

Como se ha visto anteriormente, un estudiante ideal es consciente de sus conocimientos y sus falencias, y sabe cómo emplear las herramientas que tiene disponibles para mejorar su aprendizaje. Pero la mayoría de los alumnos se alejan mucho de esta idealidad.

La experiencia de estudiar mediante un sistema educacional con tutor automático por si solo ya es autorregulado y requiere que el estudiante vincule las interacciones con el sistema con sus objetivos de aprendizaje. Una estrategia de regulación en un SIT es la *corregulación*, este concepto interviene en la autorregulación de forma colaborativa y trata a la regulación como un proceso que se debe realizar entre el alumno y un tutor u otra persona, es decir un acuerdo entre varias partes interesadas (Järvelä & Järvenoja 2011). Por esto, mientras el SIT regula sus intervenciones pedagógicas de acuerdo a lo que el estudiante va haciendo, éste regula como debe ser interpretada esta realimentación (Goldberg & Spain 2014).

Otra estrategia es la *regulación externa*, en la cual el sistema selecciona los objetivos de estudio y los elementos necesarios en el proceso de aprendizaje (Azevedo et al. 2007). Esta estrategia es útil para el diseño de agentes pedagógicos (Hadwin, Wozney, & Pontin 2005).

Para abordar las fases del aprendizaje autorregulado propuestas por Zimmerman y Campillo (2003), existen varias estrategias que utilizan los SITs. En la fase de reflexión previa del aprendizaje autorregulado, el SIT puede asistir al estudiante en un correcto establecimiento del plan inicial de ejecución para llegar a los objetivos fijados. En esta fase el SIT también puede ayudar a seleccionar correctamente la estrategia de autoevaluación más adaptada al estudiante, ya que los estudiantes comúnmente sobreestiman sus habilidades metacognitivas, pues se basan en su estimación de cuan familiarizados están con el tema de aprendizaje.

Las estrategias para abordar las fases de ejecución y de autorreflexión por un SIT dependen de las características de la tarea de aprendizaje. Una estrategia muy utilizada es la llamada búsqueda de ayuda (help-seeking) (Pintrich 2004), ésta es la capacidad que tiene al alumno de solicitar ayuda cuando él cree pertinente. Es decir, cuando no sepa cómo resolver un problema o qué paso es el siguiente que tiene que realizar o cuando no sepa como corregir un error. En un CBLE este tipo de ayuda se la hace bajo demanda del alumno, un ejemplo es el presentar un mensaje acerca de cuál es la siguiente acción que tiene que realizar. Otro ejemplo puede ser presentar un glosario de términos y/o conceptos en el proceso actual que ayudarán al estudiante cuando él lo solicite.

Otra estrategias es la autoexplicación (Chi & VanLehn 2010), la cual promueve que los estudiantes expliquen con sus propias palabras lo que han aprendido. Esta estrategia ayuda al SIT a darse cuenta cuales son los conceptos que necesitan una mayor asistencia para su comprensión. Una de las formas para realizar una auto-explicación dentro de un CBLE es por medio de un menú que permita al estudiante seleccionar la explicación que demuestra mejor lo que se ha aprendido (Aleven & Koedinger 2002). Esta forma de autoexplicación está comprobada que es significativamente mejor a que los estudiantes ingresen la explicación en una caja de texto (C. I. Johnson & Mayer 2010).

2.3. Minería de Datos

Gracias a la necesidad de las personas de utilizar sistemas informáticos, tanto en sus trabajos como fuera de ellos, éstos han ido almacenando una gran cantidad de datos que diariamente se incrementan exponencialmente. Mediante el correcto análisis de una gran cantidad de datos obtenidos por uno de estos sistemas, se puede obtener un “conocimiento” que puede ser valioso en la toma de decisiones sobre el ámbito en el que han sido tomados los datos, es decir, éstos pasan de ser un “producto” a ser la “materia prima” (Hernández Orallo et al. 2004).

Hall, Witten, y Frank (2011) definen la minería de datos como “el proceso automático o semiautomático de descubrir patrones en los datos”. Por su parte, Tan, Steinbach, y Kumar (2013) describen la utilidad de las técnicas de minería de datos indicando que se emplean “para buscar en grandes colecciones de datos con el fin de encontrar nuevos y útiles patrones que de otra manera permanecerían desconocidos”.

De estas definiciones se deduce que el resultado del proceso de minería sobre los datos generados por un sistema informático son los patrones, los cuales, entre otras aplicaciones, pueden permitir realizar predicciones no triviales sobre datos futuros, las cuales, a su vez, puedan facilitar la toma de decisiones para producir beneficios a la organización.

El conocimiento extraído se representa en un “modelo de conocimiento”, el cual no tiene los datos analizados, sino las reglas, relaciones, patrones o estadísticas que se podrían aplicar a nuevos datos para obtener predicciones o relaciones descriptivas.

Los modelos generados pueden ser de dos tipos (Hernández Orallo et al. 2004): **predictivos** o de aprendizaje supervisado y **descriptivos** o de aprendizaje no supervisado. Los predictivos tienen como finalidad hacer estimaciones sobre los valores futuros de variables relevantes para el negocio de la organización; mientras que la finalidad de los descriptivos es identificar patrones que resuman o expliquen los datos.

2.3.1. Descubrimiento de conocimiento en base de datos

El proceso conocido como *Knowledge Discovery in Databases* (KDD) es definido por U. M. Fayyad, Piatetsky-Shapiro, y Smyth (1996); U. Fayyad, Piatetsky-Shapiro, y Smyth (1996) como “el proceso no trivial de identificar patrones en datos, los cuales sean válidos, novedosos, potencialmente útiles, y comprensibles”.

Figura 2.8 Descubrimiento de conocimiento en base de datos (KDD).

Como se puede ver en la figura 2.8, el proceso de KDD es iterativo, es decir, dependiendo de los resultados de cada fase se puede regresar a cualquiera de las anteriores.

Este proceso se inicia con la “preparación de los datos”. En principio, es necesario recopilarlos de distintos repositorios originales e integrarlos en una sola base de datos con un formato común (*Data Ware House*). Tras conseguir esta recopilación de datos se debe realizar una limpieza sobre ella para eliminar los datos incorrectos. Posteriormente, se transforman los datos dependiendo de la estrategia de minería que se desee aplicar sobre ellos y luego se seleccionan los atributos que se consideren relevantes para el análisis. Seguidamente, se realiza el proceso de minería (véase la sección 2.3.2) para encontrar patrones utilizando algoritmos procedentes de la inteligencia artificial.

El proceso de KDD termina con la evaluación e interpretación de los patrones, ya sea por parte de otro sistema informático o un experto humano, para su posterior difusión de los resultados.

2.3.2. Fase de minería de datos en proceso de descubrimiento de conocimiento en base de datos

La fase de minería de datos es la etapa principal del KDD. Su objetivo consiste en producir patrones que el usuario pueda utilizar para crear nuevo conocimiento. Estos patrones y las relaciones con los datos se describen en modelos construidos gracias al análisis de los datos.

Antes de ejecutar un proceso de minería sobre unos datos se deben tomar tres decisiones importantes:

1. Determinar el tipo de tarea de minería más apropiado.
2. Elegir el tipo de modelo.
3. Seleccionar un método y un algoritmo que resuelva la tarea y obtenga el tipo de modelo antes definido.

Es importante diferenciar los conceptos de “tarea de minería” y de “método de minería”. Una tarea o tipo de tarea de minería de datos representa el problema que se desea resolver mediante el análisis de los datos, por ejemplo, un agrupamiento o una clasificación. En cambio, un método o técnica de minería es la forma en la que se resuelve dicho problema, por ejemplo, un agrupamiento jerárquico o una clasificación por medio de árboles de decisión. Por otro lado, para cada método pueden existir diferentes algoritmos como, por ejemplo, el *K-Means* para agrupamientos o el *id3* para árboles de decisión.

Los tipos de tareas y de modelos están estrechamente vinculados, puesto que para generar modelos predictivos se deben realizar tareas de clasificación o regresión, mientras que para generar modelos descriptivos se utilizan el agrupamiento (*clustering*), el análisis correlacional y las reglas de asociación (Hernández Orallo et al. 2004; Tan et al. 2013). A continuación, se definen las principales tareas de minería de datos.

Tareas de minería de datos

Como se dijo anteriormente, dentro de las tareas predictivas encontramos la clasificación y la regresión.

Clasificación:

Este tipo de tarea pretende crear un modelo clasificador a partir de un conjunto de instancias para las que se conoce su clase (registros en la base de datos). Para ello, se seleccionan uno o varios de sus atributos, llamados “clases de la instancia” (Weiss & Kulikowski 1991), y se cataloga a dichas instancias según estos atributos relevantes. Así, el modelo clasificador podrá indicar a qué clase pertenece una instancia que se desee clasificar en el futuro.

Regresión:

Ésta es una tarea que intenta predecir valores utilizando una función de regresión, la cual mapea un elemento de la base de datos con una variable de predicción de valor real (U. Fayyad et al. 1996). Por ejemplo, predecir un índice bursátil usando otros indicadores económicos, o, prevenir la cantidad de precipitaciones en una región basándose en las características de la corriente en chorro (Tan et al. 2013).

Los tipos de tareas descriptivas son: el agrupamiento o *clustering*, correlaciones y reglas de asociación.

Agrupamiento:

Se busca identificar un conjunto finito de categorías o grupos que sirvan para agrupar los datos o instancias dadas (U. Fayyad et al. 1996). Los grupos son creados teniendo en cuenta en qué medida se parecen entre sí las instancias. Para ello, se maximiza la similitud de una o varias características de los elementos de un mismo grupo y se minimiza dicha similitud entre elementos de diferentes grupos. Aunque no es su cometido más habitual, el agrupamiento se puede utilizar también para crear un modelo clasificador cuando no se conocen a priori las categorías a las que pertenecen las instancias del conjunto de entrenamiento.

Correlaciones:

Es una tarea descriptiva que se utiliza para examinar el grado de dependencia lineal existente entre los valores de dos atributos numéricos (Hernández Orallo et al. 2004). Por ejemplo, si se desea saber si el grosor de la protección de los cables eléctricos influye en la frecuencia de que ocurran incendios.

Reglas de asociación:

Su objetivo es encontrar coocurrencias de eventos frecuentes Hand, Mannila, y Smyth (2001). Es decir, encontrar relaciones entre atributos del estilo si $A = 1 \wedge B = 1 \Rightarrow C = 1$.

Métodos o técnicas de minería de datos

Como se comentó anteriormente, los métodos o técnicas de minería de datos son la forma en la que se resuelven las tareas de minería. Una técnica puede dar solución no solo a un tipo de tarea, sino a varios.

Las principales técnicas de minería de datos son las siguientes (Hernández Orallo et al. 2004):

Algebraicas y estadísticas:

Generan modelos mediante funciones algebraicas como las lineales, no lineales y distribuciones; o también mediante valores agregados estadísticos como medias, varianzas, correlaciones, etc. Algunos de las técnicas más utilizadas son la regresión lineal, regresión logarítmica y regresión logística.

Bayesianas:

Usan probabilidades a priori o condiciones inversas basadas en el teorema de Bayes para clasificar los datos U. Fayyad et al. (1996). Entre estos métodos, dos de los algoritmos más utilizados son el clasificador bayesiano naive y el de maximización esperada (EM por sus siglas en inglés).

Conteos frecuentes y tablas de contingencia:

Como su nombre indica, estas técnicas cuentan sucesos o conjuntos de sucesos que ocurren más de una vez. Cuando la base de datos es muy grande existen varias formas de que este conteo sea rápido, como, por ejemplo, comenzar por pares de sucesos e ir incrementando aquellos conjuntos cuya frecuencia supere un cierto valor. El algoritmo más popular es el *Apriori* (Agrawal & Srikant 1994).

Árboles de decisión y sistemas de aprendizaje de reglas:

Mediante este tipo de técnicas se generan reglas de clasificación, y se basan en dos tipos de algoritmos, los que promueven la filosofía de “divide y vencerás” como el ID3 (Quinlan 1986) y los que fomentan el “separa y vencerás” como el CN2 (Clark & Niblett 1989).

Relacionales, declarativas y estructurales:

Utilizan programas implementados en lenguajes declarativos para encontrar patrones en múltiples tablas de una base de datos relacional (Džeroski 2001), a diferencia de los métodos tradicionales que utilizan datos desnormalizados. La técnica más representativa es la programación lógica inductiva (Džeroski 2001).

Redes neuronales:

Utilizan redes neuronales para encontrar patrones en los datos, es decir, aprenden un modelo mediante el entrenamiento de las señales de comunicación que se envían entre nodos o neuronas, estas señales se envían entre ellos utilizando conexiones con pesos (Singh & Chauhan 2009).

Basadas en núcleos:

Estos métodos son clasificadores lineales que utilizan funciones de núcleo para aumentar la dimensionalidad y que maximizan el margen entre los grupos o clases formadas con los datos. La técnica más conocida es la máquina de vectores de soporte (Wang 2005).

Estocásticas y difusas:

Dentro de este tipo de técnicas se encuentran las que forman parte de la computación flexible (*soft computing*) (Mitra, Pal, & Mitra 2002). Este tipo de computación permite obtener soluciones inexactas para problemas reales ambiguos y/o complejos, pero en un tiempo de cálculo razonable. Entre estos métodos se encuentran, además de las redes neuronales explicadas anteriormente, los basados en la lógica borrosa, los conjuntos aproximados, los algoritmos evolutivos y los algoritmos genéticos.

Basadas en casos, en densidad o distancia:

Estos métodos permiten realizar clasificaciones basándose en las distancias que existen entre los elementos (Knorr & Ng 1998), por medio de funciones de densidad o simplemente calculando los vecinos más próximos. Uno de los algoritmos más famosos es el *k-means*.

2.3.3. Análisis de agrupamiento o clustering

Consiste en dividir un conjunto de datos en subconjuntos (Han, Kamber, & Pei 2012). Cada grupo es conocido como clúster por su definición en inglés (*cluster*), las instancias u objetos dentro de un mismo clúster son similares entre sí con respecto a una o varias de sus características.

El agrupamiento es una de las técnicas de minería de datos más utilizada. Por ejemplo, en el campo de la inteligencia empresarial se puede utilizar para obtener grupos de un gran número de clientes donde dichos clientes comparten características similares como preferencias de compra.

Debido a la gran variedad de algoritmos de agrupamiento existentes en la literatura científica, es complicado categorizarlos todos. Sin embargo, en general, la mayoría de los métodos más utilizados pueden ser clasificados en una de estas cuatro categorías (Han et

al. 2012): métodos de partición, métodos jerárquicos, métodos basados en la densidad y métodos basados en grillas.

Métodos de partición

Dado un número de objetos, estos métodos construyen un número de particiones de los datos, donde cada una de ellas representan a un clúster. Es decir, divide a las n instancias presentes en los datos en k grupos siempre que $k \leq n$. La mayoría de estos métodos construyen estos grupos basándose en la distancia que existe entre los objetos, es decir, dado el número de grupos que se desea obtener, crean una partición inicial y posteriormente emplean una técnica de reubicación iterativa, la cual mueve cada objeto de grupo en grupo analizando cuán cerca están los demás miembros del grupo y cuán lejos están los miembros de otros. Estos métodos son eficientes en bases de datos con un tamaño pequeño o mediano y los algoritmos más utilizados son el *k-means* y el *k-medoids*.

Métodos jerárquicos

Estos métodos se derivan de la idea de que el número de grupos que existen en los datos es algo relativo y depende de la separación o umbrales que se establezcan. Por esto, estos métodos descomponen jerárquicamente el conjunto de datos, es decir, construyen un árbol en el que cada nodo es un subconjunto de los datos, que puede ser utilizado como clúster, y cada hoja es un dato.

Los métodos jerárquicos pueden clasificarse de dos formas, por aglomeración (*bottom-up*) o por división (*textittop-down*). Los métodos por aglomeración comienzan con cada elemento conformando un único grupo, y posteriormente, realizan combinaciones sucesivas de objetos o grupos dependiendo de su cercanía hasta que todos converjan en un grupo. Los métodos por división, en cambio, realizan el proceso contrario, es decir, comienzan con un grupo formado por todos los elementos, y lo divide en varios grupos, los cuales, a su vez, van siendo divididos iterativamente en grupos más pequeños (dependiendo de su cercanía) hasta que cada grupo contiene un único objeto.

Métodos basados en la densidad

Al contrario que los métodos antes explicados que utilizan la distancia entre objetos para el cálculo del agrupamiento, éstos utilizan la densidad que existe en cada grupo. Es decir, van aumentando la población de un grupo mientras que la densidad en la vecindad exceda un límite pre-establecido. Un ejemplo de algoritmo de este tipo de método es el DBSCAN (Ester, Kriegel, Sander, & Xu 1996).

Métodos basados en cuadrículas

Dividen el espacio de objetos en un número finito de celdas que forman una estructura de cuadrícula. La mayor ventaja de estos métodos es su rápido procesamiento debido a que el uso de cuadrículas normalmente proporciona una solución eficiente, y en este caso, el tiempo de procesamiento es independiente del número de objetos y sólo depende del número de celdas.

2.3.4. Minería de datos secuenciales

Una **secuencia** es una lista ordenada de eventos, los cuales pueden ser de tres tipos distintos (Han et al. 2012): series temporales, secuencias biológicas o secuencias simbólicas.

Las **series temporales** consisten en secuencias de eventos que se producen en intervalos iguales de tiempo, por ejemplo, procesos naturales o económicos que se producen anualmente.

Las **secuencias biológicas** se refieren a la organización secuencial de los aminoácidos en la cadena proteica y a la secuencia de ADN o secuencia genérica. Las secuencias de este tipo son muy grandes y complejas, además de que los intervalos entre eventos toman un papel muy relevante.

A diferencia de los otros dos tipos, las **secuencias simbólicas** son aquellas en las que los intervalos de tiempo entre eventos no son relevantes y los eventos no tienen por qué producirse en intervalos de tiempo iguales. Por ejemplo, las secuencias de compras de clientes o la secuencia de navegación dentro de una página web (Hernández Orallo et al. 2004).

A continuación, por ser de mayor interés para esta tesis, nos vamos a centrar en las tareas de minería de datos que se pueden realizar con secuencias simbólicas.

Minería de patrones secuenciales

Un patrón de secuencia es una sub-secuencia frecuente existente dentro de una secuencia o un conjunto de secuencias (Han et al. 2012). Formalmente esto quiere decir:

$$\begin{aligned} \alpha = (a_1 a_2 \cdots a_n) \text{ es sub-secuencia de } \beta = (b_1 b_2 \cdots b_m) \\ \Downarrow \\ \exists \text{ enteros } 1 \leq j_1 < j_2 < \cdots < j_n \leq m | a_1 \subseteq b_{j_1}, a_2 \subseteq b_{j_2}, \dots, a_n \subseteq b_{j_n} \end{aligned} \quad (2.1)$$

Por ejemplo (a, b, d) y (ab, d) son sub-secuencias de la secuencia (abc, be, de, a) . La minería de patrones de secuencia consiste en “descubrir un conjunto de secuencias que sean frecuentes en una secuencia o en un conjunto de secuencias” (Han et al. 2012).

Clasificación con secuencias simbólicas

El objetivo de la clasificación de secuencias es poder predecir la clase a la que pertenece toda la secuencia de eventos. Un ejemplo de aplicación de la clasificación de secuencias es el reconocimiento de correo *spam*, es decir, que dada la secuencia de palabras existentes en un correo electrónico, dicho correo se pueda catalogar o no como *spam* (Hernández Orallo et al. 2004).

Una de las técnicas más utilizadas es la denominada “Ventanas Deslizantes”, la cual permite convertir un problema de predicción secuencial en uno de predicción normal. Al aplicar esta técnica, se parte de un parámetro definido N que corresponde al tamaño de la ventana y se crean dimensiones correspondientes a N -gramas de elementos. Esta técnica se ha utilizado para detectar fraudes en el uso de telefonía móvil (Hofmeyr, Forrest, & Somayaji 1998). Una mejora de esta técnica se denomina “Ventanas Deslizantes Recurrentes”.

Otra técnica utilizada es la de los modelos ocultos de Markov, la cual se ha utilizado en la clasificación documental Diligenti, Frasconi, y Gori (2003).

Agrupamiento con secuencias simbólicas

Como ya se ha descrito anteriormente, el agrupamiento en minería de datos consiste en separar los datos en grupos, de tal forma que los elementos de un mismo grupo mantengan características similares entre sí y diferentes a las de los elementos de otros grupos. Para el caso de los datos secuenciales consiste en la búsqueda de grupos de secuencias con alta cohesión (Hernández Orallo et al. 2004).

La mayoría de las técnicas primero establecen una medida de similitud entre secuencias (Gusfield 1997), y posteriormente utilizan algoritmos de agrupamiento tradicionales tales como el *k-means*. Otra técnica se basa en utilizar modelos de Markov para identificar las secuencias y determinar su probabilidad (Smyth 1997).

Un ejemplo de algoritmo de este tipo de clusterización es llamado ***Microsoft Sequence Clustering*** (Microsoft 2016), el cual en un inicio utiliza cadenas de Markov para analizar e identificar secuencias ordenadas. Posteriormente, utiliza este resultado como entrada de una implementación del algoritmo *Expectation Maximization*, mediante el cual se generan los grupos.

Reglas de asociación con secuencias simbólicas

El objetivo de este tipo de reglas radica en encontrar patrones del tipo si $A, B \Rightarrow C$. Por ejemplo, si un cliente ha comprado un producto A, al día siguiente compró B entonces es muy probable que después compre C.

Uno de los algoritmos más populares es el *Apriori* (Agrawal & Srikant 1994), el cual busca las secuencias más frecuentes basándose en el conocimiento previo (a-priori) de los ítems y sub-secuencias frecuentes. Otro algoritmo es el *Generalized Sequential Patterns (GSP)* (Srikant & Agrawal 1996), que a diferencia del anterior, permite agregar al proceso de minería algunas restricciones de tiempo, es decir, secuencias de una determinada duración.

Dentro de la minería de datos de reglas de asociaciones, especialmente en las que existen ítems frecuentes, existen dos conceptos importantes que cabe destacar, el soporte de un ítem y la confianza de una regla.

El **soporte** o **cobertura** (*support*) de un ítem se define como el porcentaje o número veces que dicho ítem del conjunto se encuentra en la base de datos, mientras que la **confianza** o **precisión** (*confidence*) de una regla representa al porcentaje o número de veces que la regla se cumple en la base de datos (Hernández Orallo et al. 2004; Hasher, Hornik, Grun, & Buchta 2005). Las siguientes ecuaciones, indican como calcular el soporte de un ítem X (ecuación 2.2) y la confianza de una regla $X \Rightarrow Y$ (ecuación 2.3) de una base de datos con T ítems (Agrawal, Imieliński, & Swami 1993):

$$\text{soporte}(X) = \frac{f_X}{T} \quad (2.2)$$

$$\text{confianza}(X \Rightarrow Y) = \frac{\text{soporte}(X \cup Y)}{\text{soporte}(X)} \quad (2.3)$$

En la ecuación 2.2 f_x representa al número de veces o frecuencia que X se encuentra en la base de datos, la cual se divide entre el total de ítems en la base de datos (T) para obtener el soporte de X .

Se puede apreciar en la ecuación 2.3 que la confianza de la regla $X \Rightarrow Y$ se define como la división de dos soportes, la particularidad está en el numerador, el cual, es el soporte de la unión de ítems en X y Y , es decir, la probabilidad de encontrar dicha regla dentro de la base de datos.

Modelos secuenciales basados en grafos

Otro tipo de modelo para simbolizar patrones secuencias son los grafos, los cuales no solo pueden representar las relaciones entre objetos, sino también algunas características de estos como la frecuencia de un objeto dentro de la base de datos.

Un **grafo** es un conjunto de vértices o nodos conectados entre sí por medio de arcos o aristas (Bondy & Murty 1976). Tan et al. (2013) define un grafo como “una estructura de datos que puede ser utilizada para representar relaciones entre un conjunto de entidades”.

Formalmente un grafo se define como $G(V, E, \phi)$ donde V representa al conjunto de vértices del grafo, E a su conjunto de arcos y ϕ define la función que asocia cada arista de G con un par de nodos. Cada arista asocia dos nodos, que pueden ser el mismo nodo, en cuyo caso el arco representa un bucle. El número de nodos es conocido como el *orden* del grafo y el de arcos como su *tamaño*.

La función ϕ para un arco está definida por la siguiente ecuación:

$$\phi(e) = (v_1, v_2) | v_1 \wedge v_2 \in V \quad (2.4)$$

Figura 2.9 Ejemplo de un grafo.

Un grafo puede contener etiquetas en sus aristas. Cuando las etiquetas son elementos de un conjunto ordenado, como, por ejemplo, los números enteros o reales, se las conoce como pesos y al grafo toma la característica de ser ponderado.

Existen varios tipos de grafos dependiendo de su número de vértices, número de aristas, interconectividad, y su estructura. Entre los diferentes tipos vamos a centrarnos, por ser de relevancia para el presente trabajo, en los grafos dirigidos.

Un **grafo dirigido (dígrafo)** es aquel en el que cada arista tiene una orientación asignada (Bondy & Murty 1976). Es decir, si a es un arco, u y v son vértices tal que $\phi(a) = (u, v)$, entonces a une a u y v estando u en la cola de a y v en su cabeza (Bang-Jensen & Gutin 2008), como muestra la figura 2.10.

En un dígrafo se puede especificar una ruta o camino dirigido, el cual se define como una secuencia finita no nula $W = (v_0, a_1, v_1, a_2, v_2, \dots, a_k, v_k)$ (Bondy & Murty 1976)

Figura 2.10 Ejemplo de un digrafo.

donde $\forall i = 1, 2, \dots, k \exists a_i, v_i, v_{i-1}$, donde a_i es un arco en cuya cabeza se encuentra un vértice v_i y en cuya cola otro v_{i-1} .

Los dígrafos se pueden aplicar para resolver el “problema de la secuencia de trabajo” (Bondy & Murty 1976), en el cual existe un número de tareas que se debe realizar en un orden determinado en una máquina que tiene que ser ajustada durante un tiempo después de cada una de estas tareas.

Otra aplicación de los dígrafos se encuentra en la planificación de rutas dentro de sistemas de información geográfica. Como, por ejemplo, el presentado por Edelkamp y Schrödl (2003), quienes construyen un grafo dirigido con los datos geográficos y obtienen las rutas óptimas entre dos lugares utilizando el algoritmo de Dijkstra o el A*.

Mannila, Toivonen, y Verkamo (1995); Mannila, Toivonen, y Inkeri Verkamo (1997) consideran a los grafos dirigidos acíclicos como la forma “obvia” de describir un episodio. Y partiendo de esta representación, Mannila et al. (1995); Mannila et al. (1997) sostienen que se pueden encontrar episodios frecuentes que ocurran dentro de una secuencia de eventos. Para esto, definen los conceptos de secuencia de eventos y episodios.

Una **secuencia de eventos** se especifica mediante un instante de inicio, otro de fin y un conjunto de eventos donde cada uno de ellos está asociado con el instante en el que se produce. Es decir, dado un conjunto de eventos E la secuencia de eventos s es una tripleta (s, T_s, T_e) , donde $s = \{(A_1, t_1), (A_2, t_2), \dots, (A_n, t_n)\}$. En esta fórmula $A_i \in E$, T_s corresponde al instante inicial de la secuencia, mientras T_e al instante final, por lo tanto, $T_s \leq t_i < T_e$.

Un **episodio** es una colección de eventos ordenados que ocurren conjuntamente. Un episodio puede ser especificado mediante un grafo dirigido acíclico, ya que si un evento está definido como (A, t) , es posible generar un grafo con todos los eventos de tipo A ordenados por t . Es así como uno o varios grafos pueden describir las relaciones entre eventos básicos usados para modelar las secuencias (Mannila & Meek 2000).

La figura 2.11 muestra los tipos de episodios que se pueden dar en una secuencia de eventos. El primero (2.11a) representa un episodio serial, es decir, los eventos de tipo E y F ocurren en ese orden. El segundo (2.11b) es un ejemplo de un episodio paralelo, en el cual no importa el orden en el cual se ejecuten los eventos. Finalmente el ejemplo 2.11c representa un evento no-serial y no-paralelo, es decir, que el evento de tipo C ocurre solamente cuando antes se hayan producido los eventos del tipo A y B .

Figura 2.11 Ejemplos de episodios (Mannila, Toivonen, & Inkeri Verkamo 1997).

Se puede realizar minería de datos sobre datos representados como un grafo para encontrar patrones. Generalmente el resultado de realizar tareas de minería sobre un grafo es otro grafo, el cual puede ser un sub-grafo del original o expresiones más abstractas (Cook & Holder 2006). Un tipo de tarea de minería que se puede aplicar sobre un grafo es la conocida como minería de sub-grafos frecuentes, la cual puede ser resuelta por medio del algoritmo *Apriori* (Tan et al. 2013).

Los dígrafos suelen ser utilizados para representar gráficamente a los autómatas finitos (Kaminski & Pinter 1992), es decir, mediante un diagrama de transición de estados Hopcroft, Motwani, y Ullman (2006). Un grafo dirigido se define como $G(V, E, t, h)$, como se puede ver, esta definición difiere de la de un grafo normal en la función de asociación de las aristas, ya que ésta presenta dos funciones. $t(e)$ representa a la función de fin (*tail*) de la arista e mientras que $h(e)$ al inicio (*head*).

Un autómata finito se define por una 5-tupla $(Q, \Sigma, \delta, q_0, F)$ Hopcroft et al. (2006), donde Q es el conjunto de estados, Σ un alfabeto de entrada finito, q_0 el estado inicial elemento de Q , F el conjunto de estados finales que son un subconjunto de Q y δ es la función de transición entre estados de Q para mapear Σ .

2.3.5. Herramientas para la minería de datos

Este tipo de herramientas están diseñadas para dar soporte a las tareas de minería de datos (Hernández Orallo et al. 2004). Existen herramientas comerciales, tales como SPSS Modeler¹, SAS Enterprise Miner²; y gratuitas como, WEKA³ o RapidMiner⁴.

A continuación, se describen con más detalle las dos últimas, ya que han sido utilizadas en el presente trabajo. Se han seleccionado éstas, debido a que son dos de las herramientas

¹<https://www.ibm.com/marketplace/cloud/spss-modeler/us/en-us>

²http://www.sas.com/en_us/software/analytics/enterprise-miner.html

³<http://www.cs.waikato.ac.nz/~ml/weka/>

⁴<https://rapidminer.com>

más utilizadas para el análisis de minería de datos. Además, poseen librerías gratuitas y de código libre que incluyen sus algoritmos para ser incorporados en otras aplicaciones.

Waikato Environment for Knowledge Analysis

Waikato Environment for Knowledge Analysis (WEKA) (Hall et al. 2009) es una plataforma de libre distribución para el análisis de datos, desarrollada en Java por un grupo de investigadores de la universidad de Waikato (Nueva Zelanda).

El proyecto nació en 1993 financiado por el gobierno neozelandés con el propósito de construir una herramienta que permita aplicar técnicas de aprendizaje automático, y que además permita investigar sus aplicaciones en áreas importantes de la economía de Nueva Zelanda. (Hall et al. 2009)

WEKA es una herramienta visual que posee cinco interfaces (figura 2.12): *explorer*, *experimenter*, *knowledgeflow*, *workbench*, *simple cli*.

Figura 2.12 Seleccionador de interfaz de WEKA.

La primera interfaz (*Explorer*, mostrada en la figura 2.13) es la herramienta principal que ofrece WEKA. Ésta permite importar datos tanto en su formato propio (ARFF) como en otros formatos como CSV o JSON. Estos datos se pueden preprocesar, es decir, seleccionar los atributos con los que se desea trabajar, discretizarlos, tratar valores desconocidos y/o transformar atributos numéricos. Además, permite crear modelos de clasificación, agrupamiento o asociación a partir de los datos preprocesados.

El entorno *Experimenter* se utiliza para ejecutar experimentos para evaluar la capacidad predictiva de los modelos creados por medio del *Explorer*.

El *KnowledgeFlow* permite crear todo el proceso de minería por medio de flujos de trabajo y de forma gráfica. Es decir, facilita la creación de un proceso que automáticamente obtenga datos, los preprocese, cree el modelo, lo valide y presente los resultados.

Figura 2.13 *Explorer* de WEKA.

Weka contiene también una interfaz de consola llamada *Simple CLI*, la cual permite realizar las mismas operaciones que el explorador, pero mediante comandos de texto.

El *Workbench* es simplemente la unificación de las interfaces antes mencionadas en una sola ventana, la cual permite al usuario configurar su perspectiva, es decir, seleccionar que desea ver y que no en la interfaz de usuario.

RapidMiner

Aunque actualmente RapidMiner es una empresa constituida en Alemania, esta herramienta nació originalmente con el nombre de YALE (*Yet Another Learning Environment*) como un proyecto de la Universidad Técnica de Dortmund (Mierswa, Wurst, Klinkenberg, Scholz, & Euler 2006).

YALE era una herramienta de libre distribución que permitía la ejecución de experimentos de aprendizaje automático. Sin embargo, actualmente RapidMiner (figura 2.14) permite realizar muchas más tareas pero ya no es totalmente gratuito. El valor de su licencia depende de la cantidad de datos que se desea analizar, oscilando entre la gratuidad para menos de diez mil registros y un precio de diez mil dólares para una cantidad ilimitada de datos⁵.

⁵Precios publicados en su web (acceso 17 de agosto de 2016: <https://rapidminer.com/pricing/>)

Figura 2.14 RapidMiner.

RapidMiner permite el diseño de un proceso de minería de datos basado en flujos de trabajo (Hofmann & Klinkenberg 2013). Cada flujo de trabajo se especifica por medio de operadores que permiten realizar diferentes tareas del proceso de minería de datos.

Estos operadores permiten realizar tareas como la obtención de datos, desde archivos en formato Excel o CSV o desde repositorios creados previamente a partir de archivos en esos mismos formatos o de un motor de base de datos.

El preprocesado de datos es relativamente sencillo gracias a operadores que permiten seleccionar atributos, cambiar tipos de datos, realizar operaciones matemáticas sobre los mismos, etc.

Posee una gran cantidad de operadores de aprendizaje organizados en carpetas atendiendo al tipo de modelo que se desea crear (predictivo, de agrupación, asociación, correlación, etc.). Además, posee operadores para realizar validaciones que permitan estimar el desempeño de un operador de aprendizaje.

RapidMiner está implementado con una arquitectura modular, lo que facilita la instalación de complementos desarrollados por terceros como la Extensión WEKA, que incorpora las implementaciones de los algoritmos que contiene esta herramienta para poder agregarlos a un flujo de trabajo como un operador más de RapidMiner (como se muestra en la figura 2.14).

Además de las facilidades que proporciona RapidMiner a la hora de crear un procesos de minería de datos, también posee una interfaz muy sencilla de utilizar para visualizar datos, con una gran variedad de gráficas a disposición del usuario (figura 2.15).

Figura 2.15 Visualización de datos en RapidMiner.

CAPÍTULO 3

ANTECEDENTES

En este capítulo se presentan los trabajos precedentes realizados por el autor de esta tesis y su grupo de investigación de los cuales ha nacido la idea y la necesidad de esta tesis. Estos son: la arquitectura multiagente de un entorno virtual de entrenamiento con tutoría inteligente, llamado MAEVIF (sección 3.1); el modelo del estudiante que está presente en dicha arquitectura (sección 3.2); y un laboratorio virtual de biotecnología (sección 3.3).

3.1. MAEVIF

MAEVIF (de Antonio et al. 2005; Imbert et al. 2007; Méndez Pozo 2008) es una arquitectura multiagente de un entorno virtual de entrenamiento con tutoría inteligente. Fue desarrollada por el grupo de investigación de Interacción Persona-Ordenador (Laboratorio Decoroso Crespo) de la Escuela de Ingenieros Informáticos de la Universidad Politécnica de Madrid (grupo en el que colaboramos).

Esta propuesta nació debido a que la arquitectura tradicional de los SITs (sección 2.2.2) no se ajusta bien a los requerimientos de un entorno virtual de entrenamiento, ya que este tipo de entornos comúnmente son multiusuario y también pueden ser usados para entrenamiento colaborativo (de Antonio et al. 2005).

Otra funcionalidad que agrega MAEVIF, comparada con la arquitectura tradicional de los SITs, es la posibilidad de que el tutor automático tenga conocimiento explícito de los objetos 2D/3D del entorno virtual, como su posición, forma, estado, etc., para que pueda presentar esta información en sus mensajes de tutoría o incluso interactuar con estos objetos.

Gracias a que MAEVIF es una arquitectura multiagente, además de que se puede adaptar fácilmente a diferentes requerimientos de los entornos virtuales, posee la ca-

racterística de ser fácilmente extensible, es decir, permite agregar fácilmente nuevas funcionalidades mediante nuevos agentes que ayuden a mejorar la calidad de la tutoría automática o a adaptarla a las características de cada entorno de entrenamiento. Así, esta arquitectura está en constante evolución y desarrollo gracias a varios proyectos de investigación y tesis doctorales como la presentada por Clemente et al. (2011); Clemente (2011) (detallada en la sección 3.2).

3.1.1. Arquitectura

La figura 3.1 presenta la arquitectura multiagente de MAEVIF. En ella se puede apreciar que los usuarios visualizan el entorno por medio de una aplicación gráfica cliente, la cual accede a los servicios de cada agente a través de una capa intermedia o *Middleware*

Figura 3.1 Arquitectura de MAEVIF.

En la figura 3.1 principalmente se pueden apreciar los agentes que forman parte de MAEVIF, los cuales se enumeran a continuación:

- Agente de comunicación global. Su objetivo es la comunicación entre el entorno virtual y el sistema de tutoría.
- Agente de comunicación con el estudiante. Se encarga de la comunicación entre los estudiantes y el SIT. Existe una instancia de este agente para cada estudiante.
- Agente de planificación. Proporciona al SIT el plan o secuencia de acciones que se deben realizar en el proceso de entrenamiento. Si el estudiante se desvía del

plan inicial previsto, esta secuencia se puede calcular a partir de la última acción realizada por el estudiante.

- Agente experto. Posee la información de todas las acciones que puede y no puede realizar el estudiante durante el proceso de entrenamiento, así como también sus pre y post condiciones. Esta información se utiliza para validar las acciones realizadas por el alumno, así como para el cálculo del plan realizado por el agente de planificación.
- Agente de tutoría. Es el núcleo del SIT, su objetivo es proponer actividades a los estudiantes dependiendo del conocimiento que estos ya posean. Se encarga también del seguimiento de cada alumno en una actividad comparando la secuencia de acciones realizadas por este con el plan proporcionado por el agente de planificación. Si se trata de un entrenamiento colaborativo, este agente forma los equipos y otorga a cada participante un rol dentro de éste.
- Agente de tutor virtual. A cada estudiante se le asigna un avatar tutor, de tal forma que este agente se encarga de controlar la representación gráfica de dicho avatar.
- Agente de simulación. Realiza la simulación de una acción realizada por el estudiante, o del posible efecto de una acción no realizada aún. Esto último puede ser útil, si un estudiante necesita saber qué puede suceder después de ejecutar una acción determinada.
- Agente de mundo. Posee la información geométrica y semántica tanto de los objetos 2D/3D del entorno virtual como de los avatares. Esta información se puede utilizar para la tutoría, así como también para la validación de las acciones realizadas por el estudiante.
- Agente de trayectoria. Si el agente de tutoría detecta que la siguiente acción que debe realizar el estudiante tiene que ser ejecutada en otro lugar del entorno virtual, se pide a este agente que calcule la trayectoria óptima para llegar a dicho lugar. Esto permite guiar al alumno hasta ese lugar o evaluar la calidad de la trayectoria seguida por el estudiante respecto a la trayectoria óptima.
- Agente de percepción. Para que los tutores virtuales actúen de forma realista se les dota de capacidades perceptuales. De esta forma, éstos solo pueden ver lo que está sucediendo en su campo de visión (lugares, objetos y avatares). Esta información

puede ser útil también para averiguar si el estudiante está distraído o si necesita girarse para realizar o seguir con una demostración.

- Agente de modelado del estudiante. Guarda todas las acciones realizadas por cada estudiante, sus evaluaciones y otros tipos de conocimientos que se tenga de éste. Esta información es utilizada para proporcionar una tutoría adaptada a cada alumno. El modelo utilizado por este agente es explicado en el siguiente apartado.

3.2. Modelo del Estudiante de MAEVIF

Como se ha mencionado en la sección anterior, MAEVIF ha sido desarrollada con la finalidad de adaptarse a un entorno virtual de entrenamiento, en especial gracias a su característica multiusuario, por esta razón ha sido necesaria la especificación e implementación de un modelo que defina el conocimiento de cada estudiante respecto a los objetivos de aprendizaje que podrían compartir varios estudiantes (de Antonio et al. 2005).

Este modelado del estudiante se desarrolló como una tesis doctoral dentro del mismo grupo de investigación de MAEVIF, disertación realizada por Clemente (2011). Éste se enfoca en cómo modelar semánticamente el conocimiento adquirido por un estudiante durante el entrenamiento y cómo inferir en cada momento lo que sabe o no sabe un estudiante a la vista de su comportamiento y sus características personales (Clemente et al. 2011; 2014).

Todas las acciones que ejecute o intente ejecutar el estudiante dentro del entorno virtual de entrenamiento, luego de ser validadas por el agente tutor, son almacenadas por el agente de modelado del estudiante en forma de una traza de actividad de dicho alumno. Tras cada nueva acción, este modelo actualiza sus creencias sobre el conocimiento del estudiante aplicando razonamiento no-monótono. Asimismo, el modelo también se puede actualizar si el estudiante formula alguna pregunta o el tutor automático proporciona al alumno alguna pista durante la práctica.

El agente de modelado del estudiante se compone de una ontología y un método de diagnóstico pedagógico. La ontología modela distintos aspectos del estudiante así como la materia que se le pretende enseñar, mientras que el método de diagnóstico utiliza esta representación para realizar inferencias sobre el conocimiento actual de cada estudiante.

de sus conocimientos (cambios en los grados de adquisición de los objetivos de aprendizaje) a lo largo del proceso de entrenamiento.

- Ontología de información de un estudiante (*Student Information Ontology*). Esta ontología referencia a las tres anteriores, es decir, es la ontología raíz de toda la información de un estudiante.
- Ontología de objetivos de aprendizaje (*Learning Objective Ontology*). Permite especificar objetivos de aprendizaje con diferentes niveles de abstracción en los dominios afectivo, psicomotor y cognitivo del alumno. El mecanismo de diagnóstico presentado por Clemente (2011) permite inferir el estado de adquisición de los objetivos del estudiante a partir de su comportamiento registrado en la ontología de trazas.
- Ontología de objetos de aprendizaje (*Learning Object Ontology*). Describe los principales tipos de elementos de conocimiento que pueden aprender los estudiantes dentro del proceso de entrenamiento. Éstos se dividen en objetos que definen conocimiento estructural y objetos que definen conocimiento procedimental.
- Ontología de monitorización del estudiante (*Student Monitoring Ontology*). Especifica las estrategias de seguimiento de ciertas variables durante el aprendizaje.

3.2.2. Método de diagnóstico pedagógico

Como ya se ha comentado anteriormente, este método es el encargado de inferir el conocimiento que posee actualmente el estudiante. Para esto, determina los objetivos alcanzados o no por el estudiante a partir de su comportamiento registrado en la ontología.

Para realizar este diagnóstico se utilizan tres módulos que interactúan entre sí como indica la figura 3.3, estos módulos se detallan a continuación:

- Módulo de diagnóstico pedagógico (MDP). Es el responsable de inferir qué objetivos de aprendizaje han sido alcanzados o no por el estudiante con un cierto nivel de fiabilidad. Para esto utiliza un conjunto de reglas de diagnóstico pedagógico y realiza consultas a la ontología antes mencionada.
- Sistema de mantenimiento de la verdad (ATMS). Controla la consistencia de las suposiciones usadas por el MDP y determina qué creencias se sostienen y cuáles no.
- Módulo de resolución de conflictos (RC). Se encarga de determinar las posibles causas de las contradicciones que aparezcan y de resolverlas buscando un nuevo

Figura 3.4 Laboratorio virtual de biotecnología.

Este laboratorio se desarrolló para acercar a los alumnos de grado de esta rama a una práctica de este tipo, puesto que este tipo de alumnos no suelen tener acceso a un laboratorio real con el material necesario para realizar esta práctica. Esto se debe a varias razones, una es la inexperiencia de estos en la materia y el consiguiente riesgo que asumen en un laboratorio real a tener accidentes. Otra razón es la larga duración de la práctica (alrededor de dos años) debido al tiempo que demoran los árboles en crecer, que supera con creces el tiempo del que disponen los alumnos para aprobar una asignatura. Además, una práctica como esta puede costar alrededor de unos 25000 euros por investigador (tomando en cuenta solo el costo del material fungible), precio que no es asequible ni para los padres de los estudiantes ni para la universidad. Este laboratorio evita estos dos problemas acelerando el experimento tanto como sea necesario, eliminando cualquier coste derivado del uso de materiales de laboratorio costosos y evitando los riesgos asociados con este tipo de experimentos.

Este laboratorio virtual de biotecnología toma prestadas algunas características de los videojuegos: la práctica se divide en fases o logros, es decir, el estudiante debe completar cada fase antes de que pueda pasar a la siguiente. Al final de cada fase, se le presenta al estudiante una lista de los errores que ha cometido, si los hubiere. Además, las acciones realizadas por los estudiantes en este laboratorio son registradas para su posterior análisis por parte del profesor o del equipo de desarrollo del laboratorio.

Figura 3.5 Esquema de protocolo para el laboratorio virtual de biotecnología.

3.3.1. Protocolo del laboratorio

La práctica está diseñada como un problema bien definido (*well-defined*), es decir, es un ejercicio con un proceso cerrado con un único resultado correcto, en el cual no existen muchas formas distintas de alcanzar dicho resultado correcto. Los estudiantes tienen que seguir un protocolo preestablecido, aunque, en algunas etapas de la práctica, son libres de realizar acciones en diferente orden (figura 3.5), por ejemplo, cuando tienen que agregar ciertos productos químicos a una mezcla. En el protocolo completo que tienen que realizar los alumnos existen 116 acciones correctas. Además de esas acciones, por motivos pedagógicos, se han considerado también otros tipos de acciones tales como acciones incorrectas y acciones intrascendentes. Sumando también estas acciones, se obtienen 176 acciones configuradas en el protocolo. La configuración de todas estas acciones se puede encontrar en el anexo A.

La figura 3.6 muestra, como ejemplo, la primera parte de la secuencia de pasos (definidas por el profesor) que deben realizar los alumnos en este laboratorio virtual.

El proceso de aprendizaje combina partes del proceso altamente estructuradas, cuidadosamente seleccionadas por el profesor que diseñó la práctica, con partes poco estructuradas. En las partes altamente estructuradas, se le informa al estudiante sobre qué acción debe tomar a continuación, y sólo se le permite cometer errores irrelevantes. Por otro lado, en las partes poco estructuradas de la práctica, al estudiante se le da

Figura 3.6 Protocolo para la preparación de medio de enraizamiento en el laboratorio virtual de biotecnología.

una indicación de bajo detalle en lugar de presentarle un mensaje específico sobre qué acción debe realizar a continuación. Por ejemplo, cuando el estudiante tiene que preparar una mezcla de componentes químicos, se le dice que prepare una mezcla, pero no qué químicos específicos se requieren.

3.3.2. Tutor automático

Este laboratorio virtual está equipado con un tutor automático que valida los intentos de acción del usuario y proporciona ayuda sobre la siguiente acción que se debe ejecutar o sobre un error si se ha cometido alguno.

Este tutor no se encarna en un avatar dentro del entorno virtual, como sí ocurre en otros entornos de entrenamiento. Este tutor está implementado como un *script* de un

objeto invisible, por tanto está completamente programado en el lenguaje de programación propio de OpenSimulator, llamado *Linden Scripting Language* (Heaton 2007). Este tutor ha sido rediseñado y reprogramado utilizando la arquitectura de *Mono framework* (Dotor Jara 2015), para ser utilizado en futuros laboratorios virtuales. Como en este caso se utilizó C# como lenguaje de programación, el tutor se podría integrar con entornos virtuales que no necesariamente estén implementados sobre OpenSimulator.

Figura 3.7 Ejemplos de mensajes de tutoría en el laboratorio virtual. Los globos grandes son imágenes ampliadas de las capturas de pantalla.

El usuario es asistido por medio de mensajes que indican las siguientes acciones que debe realizar en la práctica (como muestra la caja superior de la figura 3.7). Sin embargo, este tipo de mensajes también se utilizan para mostrar informes al final de cada fase describiendo los errores que el estudiante haya cometido desde el principio hasta el final de dicha fase (como indica la caja inferior de la figura 3.7).

Los mensajes de tutoría así como la definición de la secuencia de pasos que conforman el protocolo de la práctica se configuran en tiempo de diseño en un archivo con formato tabular cuyo contenido es fácilmente comprensible para un profesor sin habilidades de programación.

La secuencia configurada en este archivo describe el protocolo de la práctica en términos de dependencias e incompatibilidades entre acciones (véase el anexo A). Es decir, se definen las acciones que deben haber sido realizadas antes y las que no pueden ser ejecutadas después de una acción dada. La figura 3.8 muestra un diagrama de dependencias de una parte del protocolo de la práctica utilizado en este laboratorio virtual.

Figura 3.8 Ejemplo de acciones y sus interdependencias.

En esta parte de la práctica, el estudiante tiene que preparar una mezcla de tres productos químicos en un tubo, los cuales se deben mantener refrigerados el mayor tiempo posible. En primer lugar, el estudiante tiene que tomar una bandeja de hielo (acción *a*), posteriormente, debe obtener estas sustancias químicas del refrigerador (acciones *b*, *c* y *d*) y llevarlas en la bandeja de hielo al espacio de trabajo, también llamado poyata, (acción *e*) donde se preparará la mezcla. A continuación, el estudiante necesita usar una micropipeta con una punta (acciones *f* y *g*) para agregar cada producto químico a la mezcla (acciones *h.a*, *h.b* e *i*). Finalmente, el estudiante lleva el tubo con la mezcla (acción *j*) a otra parte del laboratorio.

Como puede deducirse de las dependencias descritas en la figura 3.8, hay más de una secuencia correcta posible. La figura 3.8 muestra dos posibles secuencias correctas,

pero hay, de hecho, 12 posibles secuencias correctas. Por motivos pedagógicos, se ha considerado oportuno que el sistema permita que el estudiante pueda realizar también acciones incorrectas tales como agregar el producto químico 4 a la mezcla (acción y).

Dependencias entre acciones

Como se puede apreciar en la figura 3.8, existen varios tipos de dependencias que se detallan a continuación:

Dependencias regulares: Una dependencia regular expresa que haber realizado una acción es un requisito para la correcta ejecución de otra. Por ejemplo, en la figura 3.8, si queremos tomar el químico 1 (acción b), es obligatorio antes tomar una bandeja de hielo (acción a).

Acciones mutex: Existe otro tipo de dependencia que representan acciones que se anulan mutuamente, por esta razón se las denomina como mutex, por ejemplo, “encender una lámpara” y “apagar una lámpara”. En la figura 3.8 el ejemplo de este tipo de acciones son “tomar bandeja de hielo” (acción a) y “soltar la bandeja de hielo en poyata” (acción e).

Acciones incompatibles: Una acción A es incompatible con una acción B si B no puede ejecutarse antes que A . Por ejemplo, “agregar químico 4” (acción y) y “tomar tubo con la mezcla” (acción j) en la figura 3.8 son incompatibles, ya que si la acción y ha sido realizada, la acción j no puede ejecutarse. Esto es así en este ejemplo porque el producto químico 4 no debería añadirse a la mezcla, por tanto si se añadiera, la acción que da por acaba la preparación de la mezcla (acción j) no puede ejecutarse correctamente. Este tipo de dependencias son útiles para detectar la ejecución de acciones incorrectas que no deberían haber sido ejecutadas antes de un determinado momento de la práctica.

Acciones sin orden: En algunos casos, el proceso de entrenamiento tiene grupos de acciones (o acciones compuestas) en los cuales las acciones se pueden ejecutar en cualquier orden. Un ejemplo de este tipo de grupos en la figura 3.8 es la acción compuesta h , formada por las acciones “agregar químico 1 a la mezcla” (acción $h.a$) y “agregar químico 2 a la mezcla” (acción $h.b$).

Acciones trampa: Una acción incorrecta es una acción cuya ejecución constituye un error en sí mismo. Sin embargo, la ejecución de una acción trampa no da lugar a un error,

sino que crea las condiciones necesarias para que un estudiante pueda cometer un error más adelante. En la figura 3.8, la acción x es una acción trampa porque es irrelevante para los objetivos de la práctica, pero es un requisito previo para una acción errónea posterior (acción y).

Acciones temporales: Las acciones temporales son aquellas que caducan al pasar un tiempo o necesitan de un tiempo determinado para completarse. De esta forma, pueden tener configurado un tiempo mínimo y/o un tiempo máximo. En las primeras se define el tiempo que necesita la acción para terminar su ejecución, por ejemplo, calentar un horno. Sin embargo, las segundas caducan si no se ha ejecutado la siguiente acción antes de un tiempo determinado, por ejemplo, la acción j , en la cual el tubo debe introducirse en una nevera antes de un tiempo en el cual la mezcla se echa a perder.

3.3.3. Comunicación entre objetos

En OpenSimulator, los objetos se comunican entre sí intercambiando mensajes. La figura 3.9 muestra la secuencia de mensajes que los objetos involucrados en el proceso de la figura 3.8 intercambian entre sí mientras el estudiante está realizando la práctica.

Cada una de las acciones ejecutadas por el usuario (avatar), son validadas por el objeto Tutor. En aras de la claridad en la figura 3.9, estas solicitudes (*Validar Acción(X)*) y la respuesta del tutor (VÁLIDO, FALLO O FALLO PERO SEGUIR) se presentan una sola vez en la parte inferior. Si la respuesta del tutor es de *fallo*, no se permite la ejecución de la acción, pero el tutor automático registra el error como un error irrelevante. Es considerado irrelevante porque no se ha permitido la ejecución de la acción, por tanto el error no tendrá consecuencias en el posterior desarrollo de la práctica. Sin embargo, si la respuesta es de *fallo pero seguir*, significa que aunque la acción haya sido validada como incorrecta, se permite al alumno continuar con la práctica y se registra dicho error. En este segundo caso, el error es considerado relevante de cara a la consecución con éxito de la práctica. De esta manera, los estudiantes descubrirán personalmente las consecuencias de sus errores más adelante, ya que son simulados en el mundo virtual (en nuestro ejemplo, el resultado final sería una planta muerta).

3.4. Discusión

Si bien el modelo del estudiante (Clemente 2011; Clemente et al. 2011) y el tutor automático del laboratorio virtual de biotecnología (Riofrío-Luzcando et al. 2012; Rico

Figura 3.9 Ejemplo de la secuencia de mensajes intercambiados entre el avatar y algunos de los objetos involucrados en la práctica, siguiendo las acciones y dependencias mostradas en la figura 3.8.

et al. 2012) han sido validados y publicados en congresos y revistas, adolecen de algunas deficiencias a la hora de proveer de una tutoría adaptada a las necesidades de cada estudiante.

El modelo del estudiante que se encuentra en MAEVIF tiene el inconveniente de necesitar bastante información previa de cada estudiante para proporcionar inferencias que sean confiables respecto al grado de adquisición de los objetivos de aprendizaje. Esto puede ser especialmente problemático si el alumno es nuevo en la asignatura, y por tanto no se dispone de suficientes datos de éste relativos al proceso de entrenamiento.

En relación al tutor que implementa el laboratorio de biotecnología, éste no entrega mensajes adaptados a las necesidades de cada alumno, es decir, su ayuda es la misma para todos. Además, no previene al alumno con antelación sobre errores comunes que muy probablemente pueda cometer en un paso posterior del proceso de entrenamiento.

CAPÍTULO 4

ESTADO DEL ARTE

En este capítulo se realizará un análisis de los trabajos existentes en la literatura que estén relacionados con los objetivos de esta tesis. De esta forma, este capítulo empieza explicando las aplicaciones de la minería de datos en la educación, especialmente aquellas que modelan los registros de actividad de estudiantes recogidos en un entorno de entrenamiento procedimental, con el fin de predecir eventos futuros que puedan realizar otros estudiantes en futuras sesiones de entrenamiento (OG_1). A continuación, se presentan investigaciones que estudian métodos de presentación gráfica de datos recogidos en entornos de aprendizaje (OG_2). Finalmente, se estudian los trabajos que utilizan técnicas de minería de datos para conseguir que un sistema inteligente de tutoría pueda generar una retroalimentación más adaptada al estudiante en un entorno de entrenamiento procedimental (OG_3).

4.1. Minería de Datos en la Educación

En la actualidad muchos de los sistemas computacionales dedicados a la educación poseen bases de datos o archivos en los que almacenan información relacionada con los estudiantes, que incluyen no solo sus datos personales, sino también sus calificaciones, documentos consultados o el registro de interacción del estudiante con el sistema.

Para descubrir patrones interesantes en los datos de los estudiantes, se han aplicado técnicas de minería de datos. De esta manera nace la disciplina conocida como *Educational Data Mining* (EDM) (Romero & Ventura 2007; 2010; Romero et al. 2010; Baker 2010). El EDM examina los repositorios de datos existentes en muchos entornos de aprendizaje por computador, para analizar y entender a los estudiantes y el aprendizaje en sí mismo, es decir, para detectar patrones (Scheuer & McLaren 2012). Como resultado de este proceso,

Figura 4.1 Principales disciplinas relacionadas con el EDM (Romero & Ventura 2013).

se pretende desarrollar soluciones computacionales que puedan mejorar el aprendizaje de los alumnos.

Como se puede apreciar en la figura 4.1, el EDM es un campo interdisciplinar, que surge de la sinergia de tres grandes disciplinas: la educación, la estadística y las ciencias de la computación. Cada una de las intersecciones entre dos de estas tres disciplinas crean sub-áreas estrechamente relacionadas con el EDM como son los *Computer based learning Environments* (CBLEs), la minería de datos y *learning analytics*.

Figura 4.2 Aplicación de la minería de datos en el diseño de sistemas educativos (Romero, Ventura, Pechenizkiy, & Baker 2010).

Como se puede ver en la figura 4.2, los usuarios pueden ser tanto estudiantes como profesores. Estos últimos diseñan el material que se incluye en los sistemas educativos para que los primeros accedan a ellos a través, por ejemplo, de un entorno computacional. Con la ayuda de las técnicas de minería de datos, se puede obtener conocimiento útil a partir de la información almacenada en estos entornos computacionales, no solo del material sino también de su uso. Este conocimiento puede ser entonces utilizado no solo por los profesores o administradores de cursos, sino también por los propios alumnos para mejorar su aprendizaje.

4.1.1. Descubrimiento de conocimiento en base de datos de entornos educativos

La figura 4.3 muestra que el EDM es un ciclo iterativo de formación de hipótesis, pruebas y refinamiento. El objetivo de este proceso no es solo obtener conocimiento de los datos, sino utilizarlo para modificar el entorno educacional y así mejorar el aprendizaje de los estudiantes.

Figura 4.3 KDD en EDM (Romero & Ventura 2013).

Las tareas de minería de datos más utilizadas en EDM son:

- **Predicción.** Es usada para prever el desempeño de los estudiantes y detectar comportamientos estudiantiles.
- **Agrupamiento o *clustering*.** Agrupa materiales similares de un curso o estudiantes teniendo en cuenta sus patrones de aprendizaje o de interacción (Vellido, Castro, & Nebot 2010).
- **Detección de valores atípicos.** Puede ser utilizada para identificar estudiantes con dificultades de aprendizaje, desviaciones en el comportamiento de profesores o alumnos, o para detectar procesos irregulares de aprendizaje.
- **Minería de relaciones.** Ésta abarca la generación de reglas de asociación, patrones secuenciales y correlaciones. Suelen ser usadas para identificar relaciones entre el

comportamiento de los profesores y las dificultades de aprendizaje de los alumnos, o para encontrar errores comunes que ocurren conjuntamente.

- **Análisis de redes sociales.** Sirve para interpretar y analizar la estructura y las relaciones que se hayan en las tareas colaborativas y en las interacciones con herramientas de comunicación (Rabbany, Takaffoli, & Zaiane 2011).
- **Minería de procesos.** Permite extraer conocimiento a partir de un proceso determinado por una secuencia de eventos, los cuales se registran en *logs*. Por ejemplo, este método se ha utilizado para reflejar el desempeño de los estudiantes a partir de sus registros de exámenes Trcka, Pechenizkiy, y van der Aalst (2010).
- **Text Mining.** Ayuda en el análisis del contenido de foros, chats, documentos, etc. Algunos ejemplos de su uso en EDM serían: para agrupar documentos por temas similares; o para evaluar el progreso de los estudiantes por medio de un hilo de discusión en un foro (Tane, Schmitz, & Stumme 2004).
- **Destilación de datos para la interpretación por parte de un ser humano.** Su objetivo es apoyar en la toma de decisiones mediante la presentación de información importante de una forma resumida, interactiva y visual. Por ejemplo, puede permitir a los educadores identificar visualmente patrones en el aprendizaje de los alumnos (Baker 2010).
- **Descubrimiento mediante modelos.** Se apoya en un modelo creado usando predicción, clustering o ingeniería del conocimiento para realizar un análisis o una predicción. Por ejemplo, puede permitir identificar relaciones entre el comportamiento de los estudiantes y sus características (Bienkowski, Feng, & Means 2012).
- **Seguimiento del conocimiento.** Esta técnica muy utilizada en SITs, sirve para calcular el nivel de las destrezas de un estudiante (Corbett & Anderson 1994).

4.1.2. Trabajos relacionados

Como se explicó anteriormente, son varias las técnicas de minería de datos utilizadas en EDM (Romero & Ventura 2013), como, por ejemplo, la predicción o el agrupamiento. Algunas aplicaciones generales del EDM (Romero et al. 2010) son:

- Comunicar las actividades de los estudiantes y el uso de los cursos en línea a los actores educativos.

- Ayudar con el mantenimiento y mejorar el curso analizando los datos de uso.
- Analizar qué tan bien está estructurado el dominio, en base a la predicción del desempeño estudiantil.
- Generar recomendaciones para los estudiantes.
- Predecir las calificaciones de los estudiantes y los resultados académicos.
- Modelar a los estudiantes.

Dado el alcance de esta investigación, la revisión de la literatura se centrará en las tres últimas aplicaciones de EDM.

Algunos investigadores (Perera, Kay, Koprinska, Yacef, & Zaïane 2009; Tang & McCalla 2005; Godoy & Amandi 2010) usan la minería de datos para proporcionar sugerencias, comentarios o recomendaciones sobre cuál es la mejor retroalimentación para cada estudiante, de los cuales, algunos (Fournier-Viger, Nkambou, & MephuNguifo 2010; Fournier-Viger, Nkambou, Nguifo, Mayers, & Faghihi 2013; Barnes & Stamper 2010; Jin et al. 2012) utilizan estos modelos dentro de un SIT. Las técnicas de minería de datos más utilizadas para este propósito son la asociación, análisis de secuencias, clasificación y clusterización.

Otros (Dekker, Pechenizkiy, & Vleeshouwers 2009; Romero, Espejo, Zafra, Romero, & Ventura 2013; J. A. Lara, Lizcano, Martínez, Pazos, & Riera 2014) tratan de predecir diferentes tipos de resultados académicos de los estudiantes, tales como las calificaciones finales o los abandonos. Las técnicas de minería utilizadas con más frecuencia en este grupo son la asociación, clasificación y el agrupamiento.

El modelado de estudiantes (Antunes 2008; HersHKovitz & Nachmias 2009; Arroyo et al. 2009; Barnes & Stamper 2010; Mavrikis 2010; Porayska-Pomsta, Mavrikis, D'Mello, Conati, & Baker 2013) tiene varias aplicaciones, como el análisis del comportamiento que han tenido los estudiantes dentro del entorno o la detección de problemas de aprendizaje. Este grupo utiliza con mayor frecuencia las mismas técnicas de minería de datos que las anteriores, y, además, el análisis estadístico, redes de Bayes, modelos psicométricos y aprendizaje por refuerzo.

Un trabajo interesante publicado por Bogarín, Romero, Cerezo, y Sánchez-Santillán (2014), dentro del grupo anterior, procesa los registros de Moodle para descubrir un modelo de comportamiento estudiantil específico. En éste, se dividen los registros de actividad en grupos de estudiantes con características similares usando un método de agrupamiento, y luego se aplica la minería de procesos a los *logs* de cada grupo para crear un modelo, el cual es representado por un grafo acíclico dirigido. Este modelo muestra

las secuencias de acciones más frecuentes que han realizado los estudiantes dentro de Moodle.

Cabe destacar que la mayoría de las investigaciones en EDM citadas anteriormente se han centrado en el estudio de datos o *logs* registrados por sistemas web de *e-learning*, como Moodle o MOOCs, o datos recopilados en los currículums estudiantiles (Romero & Ventura 2007; 2010; Peña-Ayala 2014; Sukhija, Jindal, & Aggarwal 2015).

4.2. Visualización de información de datos de educación

En esta sección se describen brevemente los principales trabajos que han investigado la forma de representar gráficamente los datos relacionados con el aprendizaje de los alumnos. Existen varios tipos de visualización de este tipo de datos (Mazza 2010). En esta tesis se van a abordar los siguientes tipos: visualización de modelos de usuario, visualización de comunicaciones en cursos online y visualización de datos de seguimiento de los alumnos.

Las representaciones gráficas de los modelos de estudiantes intentan simbolizar en qué medida cada estudiante ha comprendido el material enseñado, ya sea utilizando métodos jerárquicos o árboles (Bull & Kay 2007), redes Bayesianas (Zapata-Rivera & Greer 2001) o barras de progreso (Suraweera & Mitrovic 2002).

Varios trabajos intentan visualizar gráficamente las interacciones entre los estudiantes y el instructor que se han producido en foros de discusión, e-mail, o chats. Algunos los representan como grafos (Reffay & Chanier 2003) donde los estados representan a los individuos o grupos de individuos, mientras que las transiciones muestran las interacciones que han existido entre estos. Otro ejemplo es PeopleGarden (Donath, Dragulescu, Zinman, Viégas, & Xiong 2010), el cual utiliza una metáfora de flores y jardines para visualizar la participación de los estudiantes en un tablero de mensajes. Cada flor simboliza a un individuo, mientras que el tamaño de su tallo representa la cantidad de tiempo que dicho usuario ha estado presente en el tablero, y sus pétalos, el número de publicaciones.

De los tres grupos mencionados al comienzo de este apartado, el que más se acerca a la problemática tratada en esta tesis es el tercero, es decir, la visualización de datos de seguimiento de estudiantes. Este tipo de visualización nació gracias al auge de los sistemas de educación por medio de la web, mediante los cuales se distribuye información a los estudiantes, se preparan exámenes, se crean discusiones o se gestionan clases a distancia.

Uno de los problemas asociados con estos sistemas es la dificultad que tienen los profesores para verificar la correcta ejecución de estas tareas (Mazza 2004), es decir, verificar si los estudiantes están asistiendo a las clases a distancia, leyendo el material, ejecutando los ejercicios, etc. La solución radica en presentar gráficamente los *logs* de actividad, que, si bien suelen estar accesibles para el profesor dentro de estos sistemas, su formato textual los hace de difícil comprensión.

Un ejemplo de este tipo de sistemas es CourseVis (Mazza & Dimitrova 2007), el cual es una herramienta que transforma datos de seguimiento en representaciones gráficas como globos, barras o puntos. Por ejemplo, la cantidad de discusiones de los estudiantes en un foro, las dibuja con globos por estudiante y fecha, los cuales son tan grandes como *posts* hayan escrito. Otro gráfico en CourseVis representa con un grafo de puntos, por estudiante y fecha, el número de veces que ha ingresado cada alumno a un curso y, en el mismo gráfico, pero en la parte inferior, presenta con barras los accesos globales en el día (sintetizado para todos los alumnos). Además, CourseVis puede generar otro grafo que permite visualizar los accesos de los estudiantes a los recursos.

Las ideas de CourseVis fueron implementadas como un *plugin* para Moodle con el nombre de GISMO (*Graphical Interactive Student Monitoring System*) (Mazza & Botturi 2007).

4.3. Sistemas inteligentes de tutoría para entrenamiento procedimental integrados con minería de datos

En este apartado se van a abordar los trabajos más relacionados con el primer objetivo general y el tercer objetivo general de esta tesis. Por esta razón, se realizó una búsqueda exhaustiva para comprobar que el doctorando ya conocía todos los trabajos más relacionados con dichos objetivos.

4.3.1. Metodología de búsqueda

La revisión bibliográfica de los trabajos relacionados con esta tesis consistió en una búsqueda exhaustiva en Scopus. Scopus es una base de datos bibliográfica que indexa más de 97000 conferencias, más de 34000 libros, más de 22000 revistas de más de 5000

editores, entre estos Cambridge University Press, IEEE, Springer-Nature, Wiley-Blackwell, Elsevier, ACM¹.

La cadena de búsqueda se definió en base a la problemática principal que se desea solucionar en esta tesis. Es decir, modelar el comportamiento colectivo de estudiantes, utilizando sus registros de actividad, para otorgar una tutoría adaptada, en cierto modo, a cada nuevo estudiante, dentro de un entorno de entrenamiento procedimental.

De esta forma, los conceptos básicos que abarca la cadena de búsqueda son:

- Entrenamiento procedimental,
- minería de datos para la educación, y,
- sistemas inteligentes de tutoría.

Tabla 4.1 Palabras clave por concepto para búsqueda de trabajos relacionados.

Concepto	Palabras clave
Entrenamiento procedimental	<i>procedural learning OR procedural training OR procedural task OR procedural knowledge OR sequence of actions OR action sequence OR problem-solving</i>
Minería de datos para la educación	<i>(education OR training OR educational) AND data mining</i>
Sistemas inteligentes de tutoría	<i>(intelligent tutoring AND (system OR systems)) OR ITS</i>

Se utilizaron varias palabras claves, definidas en la tabla 4.1, para cada concepto. Adicionalmente, se agregaron filtros a esta cadena de búsqueda para descartar los trabajos que sean revisiones de literatura científica o *surveys*. Además, se limitó la búsqueda a las áreas relacionadas con esta tesis, es decir, educación, matemática, ciencias de la computación e ingeniería.

La figura 4.4 presenta el número de trabajos obtenidos para cada uno de los conceptos antes mencionados y para las intersecciones entre estos. En esta figura se puede visualizar que para cada una de estas temáticas o conceptos existe un alto número de trabajos relacionados. Sin embargo, en las intersecciones entre las temáticas el número se reduce considerablemente, llegando a ser solamente 21 artículos los relacionados con los tres conceptos a la vez.

De estos 21 artículos, se ha eliminado uno por ser un artículo publicado sobre esta misma tesis. Además, se filtraron otros 10 trabajos debido a que abarcan temas que no

¹Véase en (acceso el 24/04/2017): <https://www.elsevier.com/solutions/scopus/content>

Figura 4.4 Número de trabajos relacionados por concepto afín a esta tesis.

son de relevancia, como indica la tabla 4.2. Posteriormente, se analizaron minuciosamente cada uno de los 10 artículos restantes, y se eliminaron seis debido a que no utilizan la minería de datos para obtener el modelo o debido a que no utilizan el modelo para mejorar la tutoría inteligente. Finalmente, quedaron cuatro trabajos, de los cuales tres versan sobre la misma investigación (Fournier-Viger et al. 2013; Fournier-Viger, Faghihi, Nkamboun, & Nguifo 2010; Fournier-Viger, Nkambou, & Nguifo 2008) y el otro es el presentado por Jin et al. (2012), los cuales se detallan en el siguiente apartado.

Además de examinar los 21 trabajos de la triple intersección, se han revisado superficialmente los restantes 118 de la intersección entre los conceptos “Sistemas Inteligentes de Tutoría” y “Minería de datos de la educación”, ya que se sospechaba que algunos de estos trabajos podrían estar abordando la enseñanza de conocimiento procedimental sin referirse a él con ninguna de las palabras claves que habíamos previsto. En esta segunda revisión, se descubrieron los siguientes trabajos, Assistment (Razzaq et al. 2005) y Cognitive Tutor Algebra I (Ritter, Anderson, Koedinger, & Corbett 2007; Ritter, Carlson, Sandbothe, & Fancsali 2015), que serán presentados en el siguiente apartado.

Tabla 4.2 Número de artículos por temática no relevante de los 21 encontrados.

Tema no relevante	Artículos eliminados
Factores no cognitivos	7
Modelado del conocimiento del estudiante	3

4.3.2. Trabajos relacionados

Dos conocidos SITs que emplean EDM son Assistment (Razzaq et al. 2005) y Cognitive Tutor Algebra I (Ritter et al. 2007; Ritter et al. 2015), presentes entre los 139 trabajos como indica la figura 4.4. Ambos son herramientas web que guían a los estudiantes a través del proceso para resolver ejercicios de matemáticas. Existen varios estudios de minería de datos usando datos recolectados en estos dos entornos, pero no se informa de si los resultados de estos estudios han sido utilizados o no para mejorar los servicios de tutoría inteligente. Por ejemplo, los datos de Assistment se utilizaron para crear un modelo que predice cuándo un estudiante está a punto de pedir una pista (Vicente, Adjei, Colombo, & Heffernan 2015), pero este modelo no se ha integrado con el SIT. Por otro lado, en Cognitive Tutor Algebra I, utilizan técnicas de EDM para crear un modelo que detecte las actitudes/sentimientos de los estudiantes, como el compromiso, la concentración, la confusión, la frustración y el aburrimiento, únicamente a partir de las interacciones del estudiante con el tutor inteligente (Baker et al. 2012).

Después de una búsqueda exhaustiva de trabajos relacionados (detallada en la sección 4.3.1), sólo se han encontrado dos entornos de entrenamiento que analizan mediante EDM los datos de seguimiento de estudiantes en entornos de entrenamiento procedimental para mejorar la tutoría dada por un SIT.

El primer entorno, denominado CanadarmTutor (Fournier-Viger et al. 2013; Fournier-Viger, Nkambou, & MephuNguifo 2010; Fournier-Viger, Faghihi, et al. 2010; Fournier-Viger et al. 2008), simula el brazo robótico Canadarm2 utilizado en la Estación Espacial Internacional. El SIT integrado en este entorno proporciona asistencia a los usuarios sobre cómo realizar una secuencia correcta de operaciones con dicho brazo mecánico para alcanzar una meta. Para ello, integra un modelo cognitivo que evalúa las habilidades y el razonamiento espacial, además de un sistema experto que genera automáticamente rutas de solución.

Debido a que este entorno trata de entrenar a los alumnos en un procedimiento con una solución mal-definida (*ill-defined*), el SIT utiliza técnicas de minería de datos para extraer un modelo de tareas parciales a partir de las soluciones de usuarios que han utilizado el entorno anteriormente. Mediante la utilización de este modelo, el SIT puede reconocer el plan de aprendizaje y proporcionar asistencia basada en la predicción de la próxima acción correcta que podría realizar el usuario.

Esta ayuda mostrada al alumno puede ser de dos formas, una en la cual se le sugiere cuál es el siguiente paso correcto que puede ejecutar cuando este lo solicita, o mediante una demostración de la tarea parcial gracias a la generación de rutas de solución.

El segundo trabajo encontrado por medio de la búsqueda exhaustiva fue publicado por Barnes y Stamper (2007), quienes crean un proceso de decisiones de Markov a partir de las acciones que han realizado los estudiantes. En un primer estudio utilizaron este modelo en un tutor automático creado para la enseñanza de pruebas lógicas (Barnes & Stamper 2010), y posteriormente comprobaron su utilidad en un tutor para la enseñanza de la programación de ordenadores (Jin et al. 2011; Jin et al. 2012). Estos tutores extraen las soluciones que han realizado los estudiantes pasados y definen una cadena de Markov, para luego generar pistas automáticas para la siguiente prueba lógica que debe realizar un nuevo estudiante, en el caso del primer tutor, o sobre la siguiente instrucción de programación que debe escribir, para el caso del otro tutor. De igual forma que con el CanadarmTutor, estas pistas se generan solamente si el estudiante solicita ayuda al entorno.

A pesar de la investigación que ya se ha llevado a cabo en esta área, la comunidad no tiene resultados que sean aplicables de forma genérica (Romero & Ventura 2013), por ejemplo, modelos predictivos que pueden ser utilizados en más de un contexto diferente. Asimismo, como se ha podido constatar, hay una notable escasez de sistemas educativos inteligentes que aprovechen los modelos desarrollados por EDM (Baker 2014).

4.4. Discusión

En virtud de la revisión de la literatura del área, delimitada por los objetivos de esta tesis, que se ha llevado a cabo, se puede concluir que muchos de los trabajos de EDM intentan predecir datos académicos de los estudiantes, tales como sus calificaciones o si van a abandonar o no el curso. Otras investigaciones intentan analizar la conducta de los alumnos en sistemas de aprendizaje web como Moodle. Ahora bien, son muy pocos los que analizan los *logs* de los SITs para intentar predecir el comportamiento o las emociones de los estudiantes cuando utilizan dichos entornos, y son solamente CanadarmTutor y el trabajo propuesto por Barnes y Stamper (2007), los únicos que utilizan estas predicciones para mejorar la tutoría automática en entornos de entrenamiento procedimental.

Tanto el CanadarmTutor como el proceso de decisión de Markov propuesto por Barnes y Stamper (2007), comparten algunos de los objetivos de esta tesis. Sin embargo, al estar estos SITs diseñados para dominios mal definidos (*ill-defined*), tienen que trabajar con un número muy elevado de soluciones correctas y un número incluso más elevado de soluciones incorrectas. Esto se opone a la problemática que se desea solucionar con esta investigación, en la que los dominios a enseñar están bien definidos (*well-defined*), es decir, los pasos a seguir están determinados por un protocolo o procedimiento que el

estudiante debe aprender. Asimismo, cabe destacar que estos dos trabajos se limitan a presentar ayuda al estudiante sobre cuál es la siguiente tarea que puede ejecutar, pero no son capaces de predecir posibles errores en los que pueda caer el estudiante. Por esta razón, no pueden proporcionar ayuda para evitar que los cometan. Esto es debido a la propia naturaleza de los dominios mal definidos, puesto que en estos no es posible identificar fácilmente los errores en la secuencia de acciones, ya que no existe un protocolo de referencia con el que se puedan comparar las secuencias generadas por los alumnos.

El trabajo publicado por Bogarín et al. (2014) no presenta un modelo genérico, ya que está diseñado atendiendo al tipo de interacción que se puede realizar en Moodle y no toma en cuenta otro tipo de entornos virtuales educativos, como los entornos virtuales 3D, que tienen más formas de interacción que un sistema web. Además, no se utiliza el modelo para generar pistas para ayudar a los estudiantes.

Además, la forma en la que Bogarín et al. (2014) construyen las redes heurísticas no podría ser extrapolada al modelo que se desea implementar en esta tesis, ya que se pretende que este modelo se actualice en tiempo de ejecución con cada nuevo evento que realice cada nuevo estudiante. Por esta razón, el modelo debe ser construido con un método exhaustivo y no heurístico. No obstante, la idea de dividir a los alumnos en grupos teniendo en cuenta sus *logs* de actividad, sí que será aprovechada en esta tesis.

Una conclusión interesante del trabajo de Bogarín et al. (2014), relevante para esta tesis, es que cuando la información sobre los *logs* de actividad está sintetizada en forma de gráficos, modelos o representaciones visuales, los profesores y los estudiantes encuentran esta información más accesible. Por lo tanto, estos datos podrían ser muy útiles para el profesor para poder monitorizar el proceso de aprendizaje y adaptar su docencia consecuentemente. Esta conclusión, que nosotros compartimos, nos llevó a realizar un estudio del estado del arte sobre sistemas de visualización de *logs* de actividad en entornos educativos. Al analizar estos trabajos, se concluyó que la mayoría presentan gráficamente las acciones que se ejecutan en el tiempo en forma de puntos o estados de grafos dirigidos, como en el grafo de accesos a recursos de GISMO (Mazza & Botturi 2007). Asimismo, se concluyó que ninguna de las visualizaciones halladas resulta apropiada para visualizar un modelo como el que se pretende diseñar en esta tesis.

CAPÍTULO 5

DEFINICIÓN DEL PROBLEMA

En este capítulo se determinan los objetivos específicos que se pretenden alcanzar con esta tesis. Éstos se derivan de un análisis exhaustivo de la literatura existente sobre el tema, que ha revelado ciertas carencias en otros trabajos previos realizados en el campo de los Sistemas Inteligentes de Tutoría (SIT) y la minería de datos aplicada a la educación.

Esta investigación toma como punto de partida nuestras experiencias con el laboratorio virtual de biotecnología. Como ya explicó en el apartado 3.3, este laboratorio virtual dispone de un tutor automático capaz de supervisar las acciones de los estudiantes y proporcionarles pistas. Ahora bien, este tutor no es capaz de proporcionar una tutoría adaptada a las necesidades de cada alumno. Esto implica entonces que todos los alumnos reciben la misma ayuda en un mismo paso de la práctica o cuando cometen un error independientemente de cuál haya sido su rendimiento en la práctica o su historial académico. Aunque la personalización de la tutoría automática es un requisito señalado por numerosos trabajos previos en la literatura, nuestras experiencias con el laboratorio virtual de biotecnología nos sirvieron para constatar la importancia de dicha personalización.

Por otro lado, el tutor automático va registrando las acciones que realizan los estudiantes y los errores que cometen mientras realizan la práctica. Eso significa que el tutor dispone de una gran cantidad de información que podría ser utilizada o bien para inferir las necesidades de aprendizaje de cada alumno durante la práctica, o bien para predecir el comportamiento de alumnos con características similares en la misma situación.

Al observar a los alumnos utilizar este laboratorio virtual, nos dimos cuenta de que había algunos alumnos que cometían errores similares, formulaban preguntas parecidas, tenían dificultades para encontrar las mismas herramientas, etc. Por otra parte, existían otros alumnos que apenas cometían errores y progresaban rápidamente en la práctica. Es decir, nos apercebimos de que existían grupos de alumnos con comportamientos similares.

Por esta razón, vimos interesante la idea de agrupar los *logs* de los alumnos de acuerdo al rendimiento que habían demostrado en la práctica, con el fin de identificar grupos de alumnos que sean susceptibles de recibir la misma tutoría.

Otra cuestión que se planteó al utilizar el laboratorio virtual fue la necesidad por parte del profesor de poder visualizar el comportamiento de cada alumno por separado o de todo un grupo tras completar la práctica.

Teniendo en cuenta lo antes descrito, se consideró la posibilidad de utilizar el modelo del estudiante propuesto por Clemente (2011); Clemente et al. (2011). Pero, como ya se ha explicado en el capítulo anterior, las predicciones de este modelo solamente tienen una confianza alta, cuando se dispone de una información previa del estudiante que no siempre está disponible. Por lo tanto, teníamos que resolver el problema utilizando los logs de alumnos con una formación similar que ya habían realizado la práctica. Para ello, se decidió recurrir a las técnicas de minería de datos.

5.1. Objetivos de la Tesis

Tomando como base los tres objetivos generales de este trabajo, presentados en la introducción (sección 1.2), se enumeran a continuación los objetivos específicos derivados de cada uno de estos objetivos generales.

5.1.1. Propuesta de un modelo de comportamiento colectivo de estudiantes

Para proponer un modelo que, utilizando los registros de actividad de alumnos, pueda predecir, con un cierto nivel de certeza, los eventos futuros que pueda provocar un estudiante al realizar una práctica en un entorno de entrenamiento procedimental, se plantean los siguientes objetivos específicos:

- O_1 Diseñar e implementar un modelo que sintetice todas las secuencias de eventos generadas por los alumnos que han realizado una práctica en un entorno de entrenamiento procedimental.
- O_2 Definir una estrategia de minería de datos para crear el modelo del objetivo O_1 a partir de los registros de actividad de los alumnos en un entorno de entrenamiento procedimental.
- O_3 Dividir los registros de eventos de los alumnos dentro del modelo en grupos o clústeres atendiendo a la eficiencia con la que han realizado la práctica.

- O_4 Diferenciar los eventos por tipos para poder otorgar una tutoría diferente a cada uno de ellos.
- O_5 Dotar al modelo de datos estadísticos que permitan obtener predicciones sobre los eventos más probables que podría generar un estudiante en un momento dado de la práctica.
- O_6 Definir un algoritmo que permita actualizar el modelo con nuevos registros de actividad de alumnos incluso durante la realización de una práctica.
- O_7 Definir un algoritmo que permita clasificar a un estudiante como miembro de uno de los grupos o clústeres incluidos en el modelo durante la realización de una práctica.
- O_8 Evaluar la precisión de las predicciones del modelo con datos reales registrados en un entorno de entrenamiento procedimental.
- O_9 Evaluar distintos métodos de agrupamiento para crear los grupos de alumnos (registros de eventos) atendiendo a la precisión de las predicciones logradas.

5.1.2. Diseño de una visualización del modelo de comportamiento colectivo de estudiantes

Para diseñar una visualización interactiva del modelo de comportamiento colectivo de estudiantes, se plantean los siguientes objetivos específicos:

- O_{10} Diseñar e implementar una visualización gráfica del modelo, que permita descubrir, lo más fácilmente posible, patrones de comportamiento de interés pedagógico en las secuencias de eventos generadas por los estudiantes.
- O_{11} Diseñar una forma de navegación e interacción con el modelo que, para ser efectiva y fácil de usar, siga las recomendaciones de usabilidad propuestas por expertos de reconocido prestigio en la presentación de la información.
- O_{12} Demostrar la utilidad de la visualización para mejorar la docencia de un profesor mediante una serie de casos prácticos.
- O_{13} Validar la utilidad de la visualización del modelo para mejorar la tutoría otorgada por un Sistema Inteligente de Tutoría.

5.1.3. Diseño de una estrategia de tutoría basada en el modelo de comportamiento colectivo de estudiantes

Se han planteado los siguientes objetivos específicos para poder diseñar una estrategia de tutoría que utilice el modelo individual del estudiante (Clemente 2011; Clemente et al. 2011) y el modelo de comportamiento colectivo, propuesto en esta tesis, para generar una retroalimentación de tutoría adaptada a las necesidades de cada estudiante.

- O_{14} Diseñar una estrategia de tutoría que se base en la estrategia pedagógica del andamiaje.
- O_{15} Definir el proceso que seleccione o bien el modelo individual o bien el modelo colectivo a la hora de especificar la tutoría más adecuada para el alumno.
- O_{16} Detallar los procedimientos necesarios para escoger el andamiaje más apropiado a partir de la información disponible en el modelo seleccionado (individual o colectivo).

5.2. Supuestos de la Tesis

De acuerdo a los objetivos planteados, se presentan a continuación las premisas que se van a asumir en este trabajo:

- S_1 El entorno de aprendizaje procedimental que va a ser objeto de esta tesis debe reunir las siguientes características:
 - El problema que se desea que los estudiantes resuelvan en su entrenamiento será un problema bien definido (*well-defined*), es decir, dispondrá de un protocolo de acciones bien definidas con la solución correcta.
 - El entorno dispondrá de una interfaz gráfica de usuario en 2D o 3D.
 - Los alumnos trabajarán de forma individual en su práctica, si bien cabe la posibilidad de que puedan interaccionar entre ellos durante la sesión de aprendizaje en un entorno virtual.
 - Existirán partes del procedimiento de la práctica en las que no importará el orden relativo con el que se ejecuten ciertas acciones.
 - La corrección de una acción no solo dependerá de la acción en sí, sino también del momento en el que se realice en la práctica.

- Además de las acciones correctas, el entorno debe permitir a los alumnos realizar acciones incorrectas que no formen parte del procedimiento válido.
- Si se considera pedagógicamente apropiado, el Sistema Inteligente de Tutoría podría impedir que un alumno realice una acción incorrecta. En ese caso, se debería registrar el intento en el *log* del alumno, pero no se verán en el entorno las consecuencias de ejecutar dicha acción.
- Se contemplarán errores por omisión de acciones, es decir, se producirán errores cuando los alumnos no ejecuten ciertas acciones antes de un determinado momento de la práctica.
- Se distinguirán dos tipos de errores, errores irrelevantes y relevantes para los objetivos de la práctica. Si un alumno comete un error relevante, será imposible que acabe la práctica con éxito salvo que antes ejecute una acción reparadora del error, si existe.

- S_2 El comportamiento de los estudiantes que realizan una práctica en un entorno de aprendizaje procedimental es similar al comportamiento de otros estudiantes que en el pasado realizaron la misma práctica, poseían una formación previa equiparable y perseguían los mismos objetivos de aprendizaje.
- S_3 Un SIT que sea capaz de adaptarse al nivel de conocimientos de un estudiante aplicando una estrategia de tutoría basada en andamiaje, permitirá al estudiante aprender de una forma más efectiva que un SIT sin esa capacidad.
- S_4 En una estrategia de tutoría no siempre es lo más apropiado dejar que los alumnos cometan errores para que así aprendan de ellos, sino que a veces es preciso intervenir para prevenirlos mediante pistas.
- S_5 En el proceso de selección de la retroalimentación del SIT, si el modelo individual del estudiante no contiene suficiente información acerca del estudiante al que se está proporcionando la tutoría, como para predecir su rendimiento con suficiente fiabilidad, es una buena idea recurrir a un modelo colectivo de los estudiantes que sea capaz de proporcionar predicciones para los estudiantes teniendo en cuenta el grupo o clúster al que pertenecen.

Parte II

SOLUCIÓN ADOPTADA

CAPÍTULO 6

MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

El principal aporte del presente trabajo es el Modelo de Comportamiento Colectivo de Estudiantes (MCCE), el cual, como se explica en el apartado 6.1, se ha integrado en la arquitectura de un Sistema Inteligente de Tutoría (SIT) y en la arquitectura de MAEVIF descrita en la sección 3.1.

La principal aplicación de este modelo consiste en proporcionar predicciones de eventos que permitan al SIT dar el apoyo necesario al estudiante para que no cometa un posible error o para evitar que éste se estanque en el proceso de entrenamiento.

La principal premisa para la creación de este modelo es que el comportamiento de los estudiantes que ya completaron una práctica de entrenamiento será similar al de estudiantes con una formación parecida que vayan a realizar la misma práctica. Por esto, este modelo contiene información, en forma de sumario, de eventos ejecutados por alumnos en sesiones anteriores y registrados en el modelo del estudiante (Clemente 2011; Clemente et al. 2014; 2011), descrito en la sección 3.2.

Los eventos se refieren a las acciones realizadas por los estudiantes dentro del entorno de entrenamiento, así como también a los intentos de acción, es decir, a acciones que los estudiantes intentaron hacer, pero que el SIT no permitió que fueran completadas por motivos pedagógicos. Finalmente, un evento también puede ser un error provocado por una acción o un intento de acción. Estos eventos son generados de forma secuencial para cada alumno.

El MCCE propuesto consiste en varios clústeres de estudiantes (sección 6.2), donde cada uno de ellos contiene un autómata extendido (sección 6.3), el cual es un digrafo

adaptado a los propósitos de tutoría ante mencionados. Como se detalla posteriormente, el agrupamiento de los estudiantes en clústeres ayudará a proveer de una tutoría adaptada a cada tipo de estudiante.

Este capítulo, en primer lugar, explica cómo este modelo se integra en la arquitectura de un SIT y en la arquitectura de MAEVIF. Posteriormente, se define formalmente la estructura del modelo y se muestran dos extensos ejemplos de su aplicación. Finalmente, se especifican los algoritmos para construirlo desde cero a partir de los logs de los alumnos, y para actualizarlo dinámicamente con los nuevos eventos que se produzcan durante una sesión de aprendizaje.

6.1. Integración del Modelo propuesta en un Sistema Inteligente de Tutoría

Como se ha mencionado anteriormente, esta tesis propone la integración del MCCE en la arquitectura de un SIT. Partiendo de la arquitectura clásica de este tipo de sistemas propuesto por Wenger (1987), la figura 6.1 incluye al MCCE como parte de un nuevo **Módulo de Comportamiento Colectivo de Estudiantes**, el cual interactúa con los módulos del estudiante, experto y de tutoría. Si bien es cierto que la arquitectura clásica de un SIT incluye también al módulo de comunicación, no lo trataremos aquí ya que no es de relevancia para los objetivos de este trabajo y sus responsabilidades seguirán siendo las mismas que en la arquitectura original.

La interacción del MCCE con el módulo del estudiante es necesaria, ya que éste es el que almacena los datos de interacción del cada alumno con el entorno, es decir, las acciones realizadas o los intentos de ejecución de acciones. Cada interacción del estudiante se almacena dentro de una ontología formando la traza de su comportamiento (Clemente 2011; Clemente et al. 2014; 2011). El MCCE toma las trazas de alumnos que hayan realizado la misma práctica o actividad de entrenamiento en el pasado como datos de entrada para el algoritmo de creación del mismo.

Además de los datos históricos de las trazas de antiguos estudiantes, el MCCE utiliza como entrada de su construcción la información proporcionada por el módulo experto para la actividad pedagógica deseada, es decir, la definición de cada acción, sus pre y post condiciones, así como también la descripción de los posibles errores que puede cometer un alumno.

Las predicciones generadas por el MCCE junto con las inferencias del módulo del estudiante sobre el estado de los conocimientos de cada estudiante, y el conocimiento acerca de la actividad otorgado por el módulo experto, permitirán al módulo de tutoría

Figura 6.1 Arquitectura del SIT extendido con el MCCE.

tomar decisiones sobre qué tipo de tutoría es la más apropiada para un estudiante en cada momento. Para ello, el módulo de tutoría propuesto contiene cuatro componentes, un conmutador y tres componentes generadores de tutoría: tutoría basada en el modelo del estudiante (ME), tutoría basada en el modelo colectivo de comportamiento de estudiantes y tutoría genérica.

El conmutador se encarga de decidir qué tipo de tutoría es la más apropiada para cada estudiante en cada momento. Para tomar esta decisión, el conmutador tiene en cuenta si los resultados otorgados por el módulo del estudiante tienen una baja fiabilidad. Esto puede deberse a que no exista mucha información pedagógica sobre el alumno para esa actividad. En ese caso, será mejor utilizar la predicción calculada por el MCCE, pero si ésta de igual manera tiene una baja fiabilidad, se deberá dar una tutoría genérica.

Cada uno de los componentes generadores de tutoría se caracteriza por el tipo de tutoría que otorgan. Para ello, cada uno de ellos posee una serie de mecanismos que le permiten decidir qué tutoría proporcionar al alumno.

El componente de tutoría basada en el ME utiliza los datos del estado de los conocimientos del alumno para otorgar una tutoría adaptada a lo que el ME cree que el alumno sabe o no sabe. Este tipo de tutoría es la que logra un mayor nivel de personalización, por lo tanto, será la opción preferida del conmutador, siempre y cuando se disponga de suficiente información sobre el alumno.

El componente de tutoría basada en el MCCE utiliza las predicciones del MCCE, y proporciona una tutoría adaptada al tipo de estudiante, gracias al agrupamiento en clústeres, que tiene en cuenta los futuros eventos más probables, gracias al autómata de cada clúster. El proceso de tutoría para este componente se detalla en la sección 8.5.

La tutoría genérica se refiere a un tipo de ayuda más rígida y menos adaptada al estudiante, es decir, se otorgan mensajes dependiendo de la siguiente acción que se debe realizar en la actividad (según esté configurada ésta en el módulo experto) o del último error cometido por el alumno. Un ejemplo de este tipo de tutoría se encuentra detallada en el apartado de tutoría del laboratorio de biotecnología desarrollado como preámbulo a esta tesis (sección 3.3).

Figura 6.2 Arquitectura de MAEVIF con Agente de Comportamiento Colectivo de Estudiantes.

6.1.1. Integración del modelo de comportamiento colectivo de estudiantes en MAEVIF

Este modelo ha sido integrado dentro de la arquitectura de MAEVIF (de Antonio et al. 2005; Imbert et al. 2007; Méndez Pozo 2008) como parte de un nuevo agente llamado **Agente de Comportamiento Colectivo de Estudiantes**, como muestra la figura 6.2.

Este nuevo agente se ha agregado a la arquitectura de MAEVIF considerando la modificación a la arquitectura clásica de un SIT descrita anteriormente, es decir, respetando las relaciones del MCCE con cada uno de los otros módulos explicadas anteriormente. Asimismo, se agrega la relación con el agente de comunicación global, el cual le hará llegar las acciones realizadas por cada estudiante para mantener dicho modelo actualizado. De esta manera, en la figura 6.2 se puede apreciar que los agentes que interaccionan con el agente de comportamiento colectivo son el agente de modelado del estudiante, el agente experto, el agente de tutoría y el agente de comunicación global.

Los módulos mencionados en el apartado anterior se encuentran englobados en los agentes de MAEVIF con el mismo nombre.

6.2. Descripción del Modelo de Comportamiento Colectivo de Estudiantes

La idea de construir este modelo nació de nuestra experiencias durante la evaluación del laboratorio virtual de biotecnología (Rico et al. 2012; Riofrío-Luzcando et al. 2012), detallado en la sección 3.3. Durante el diseño del modelo fue necesario observar el comportamiento de los estudiantes en este entorno virtual siguiendo un método etnográfico. Posteriormente, y como recomienda Mostow y Beck (2006), se estudiaron a mano los *logs* de los estudiantes registrados por el tutor automático de dicho laboratorio virtual para identificar fenómenos que no se habían detectado mediante la observación sistemática.

De este análisis se derivaron varias conclusiones interesantes, una de ellos fue que los estudiantes pueden ser categorizados por grupos dependiendo del desempeño que han tenido durante el proceso de entrenamiento. Además, resultaba evidente que cada uno de estos grupos de estudiantes requerían un diferente nivel de ayuda para completar su entrenamiento. Gracias a esto, germinó la idea de utilizar técnicas minería de datos de agrupamiento sobre los *logs* de los estudiantes.

Otra conclusión extraída de este estudio fue que los estudiantes cometían errores leves desde el punto de vista pedagógico muchas veces seguidas, como, por ejemplo, tocar un

objeto que ya habían tocado anteriormente debido a que no encontraban el objeto que debían tocar en la siguiente acción correcta. Este descubrimiento dio paso a la necesidad de apoyar al estudiante a salir de este tipo de bucles.

Los estudiantes realizaron dentro de este laboratorio un entrenamiento de tipo procedimental, es decir, estos debían realizar una serie de pasos secuenciales para cumplir un objetivo. Por esto, como se mencionó en el apartado 2.3.4, en minería de datos los modelos secuenciales pueden ser representados mediante modelos basados en grafos (Mannila & Meek 2000), y pueden ser descritos como digrafos para descubrir cuáles son los eventos más frecuentes (Mannila et al. 1997).

Tomando en cuenta las consideraciones anteriores, el modelo propuesto (figura 6.3) consiste en varios clústeres de alumnos por cada actividad o dominio de aprendizaje, donde cada grupo contiene, a su vez, un autómata extendido (descrito en la sección 6.3), el cual, básicamente, es un grafo dirigido adaptado a los propósitos pedagógicos de un proceso de entrenamiento procedimental.

Figura 6.3 Modelo de Comportamiento Colectivo de Estudiantes.

Teniendo en cuenta la adaptación del KDD al EDM (sección 4.1.1) formulada por Romero y Ventura (2013), este modelo puede ser considerado como el resultado de la fase de Modelos/Patrones de este ampliamente conocido proceso para el descubrimiento de conocimiento en base de datos.

El proceso de creación de este modelo comparte las mismas fases principales con el modelo propuesto por Bogarín et al. (2014), y se ejecuta en tiempo de arranque del tutor automático. El primer paso es obtener los *logs* de los estudiantes de la ontología de trazas de estudiantes del modelo del estudiante (Clemente 2011; Clemente et al. 2014; 2011).

Con estos registros de actividad, los alumnos son divididos en grupos utilizando diferentes métodos de agrupamiento (detallados en la sección 6.8). Posteriormente, en

cada clúster se construye un autómata a partir de los eventos registrados en el *log* de cada estudiante.

Como se puede apreciar, las técnicas de minería de datos utilizadas para la creación de este modelo son, en la primera fase, técnicas de agrupamiento (*clustering*); y en la segunda fase, para la construcción de cada autómata, se emplea una técnica exhaustiva de minería de procesos, específicamente de datos secuenciales, cuyo algoritmo forma parte de las contribuciones de esta tesis.

6.3. Definición del Autómata Extendido

El autómata propuesto define las secuencias de eventos generados por varios alumnos en la actividad de entrenamiento. La figura 6.4 muestra un ejemplo de este autómata, donde los estados son representados por círculos y las transiciones por flechas.

Las transiciones denotan eventos provocados por la ejecución o el intento de ejecución de una acción por parte de estudiantes que ya han realizado el proceso de entrenamiento. Estos mismos eventos podrían ser generados también por estudiantes que en el futuro llevaran a cabo la misma práctica.

Figura 6.4 Ejemplo de un autómata extendido del MCCE.

Los estados del autómata representan diferentes situaciones derivadas de los eventos existentes en los logs de los alumnos. Como se puede apreciar en la figura 6.4, existen varios eventos relacionados con acciones correctas que producen estados correctos. Por ejemplo, el evento de “hacer 1” dirige al “estado 1” y “hacer 3” lleva al “estado 3”.

Las posibles situaciones que pueden ser representadas por eventos dentro del autómata son:

- Una acción válida del ejercicio (“hacer 1” en la figura 6.4).
- Un intento de acción bloqueada por el tutor automático (“intentar 3” en la figura 6.4). Este evento puede darse a causa de que la acción fue validada como incorrecta por el tutor debido a su configuración de pre y post condiciones, y el error está configurado dentro de la estrategia de tutoría para que el estudiante no pueda ejecutarla en el momento en el que lo está intentando. Estos errores son denominados errores irrelevantes, porque no impiden que el alumno pueda completar la práctica con éxito.
- Un error detectado por el tutor automático al validar una acción incorrecta, que no ha sido configurado para que bloquee la secuencia del proceso (“fallar 3” en la figura 6.4). Una acción incorrecta puede producir más de un error, por tanto cada uno de estos es representado como un evento diferente. A estos errores se los denomina errores relevantes, porque impiden que el alumno pueda completar la práctica con éxito.
- Terminar el ejercicio sin errores (“finalizar” en la figura 6.4)

El soporte de un estado se define como la división del número de estudiantes que han pasado por ese estado entre el número total de alumnos de los cuales se ha tomado los *logs* para generar el autómata, es decir, el número total de estudiantes en el clúster. Por ejemplo (de la figura 6.4) el estado 2 tiene un soporte del 100 %, esto quiere decir que todos los alumnos han llegado a este estado, mientras que al estado 3 ha llegado un 80 % de éstos.

La confianza de una transición o evento se define como la división del número de estudiantes que han generado dicho evento entre el número de estudiantes que alcanzaron el estado desde el que parte dicha transición. Utilizando el mismo ejemplo anterior del 100 % de alumnos que han llegado al estado 2 un 20 % han ejecutado el evento “hacer 5”. Por esta razón al estado 3 solo ha llegado el 80 %.

Las siguientes ecuaciones definen el cálculo del soporte y la confianza de cada estado y evento respectivamente, basándose en la definición de estos conceptos descritos en la sección 2.3.4 (Hernández Orallo et al. 2004; Hasher et al. 2005):

$$soporte(S) = \frac{f_S}{N_C} \quad (6.1)$$

$$confianza(E) = \frac{f_E}{f_{S_1}} | E : S_1 \Rightarrow S_2 \quad (6.2)$$

La ecuación 6.1 define al soporte de un estado S , donde f_S representa la frecuencia de estudiantes en S y N_C el número total de alumnos dentro del clúster C . Mientras que la ecuación 6.2 define la confianza de una transición o evento E que va desde el estado S_1 al S_2 . En esta ecuación, f_E es la frecuencia del evento y f_{S_1} la del estado inicial.

Además de los estados y los eventos, en la figura 6.4 se pueden distinguir tres zonas que agrupan a los estados: zona del flujo correcto, zona de errores irrelevantes y zona de errores relevantes.

6.3.1. Zona de flujo correcto

Esta zona representa las secuencias correctas de acciones, determinadas por el módulo experto, que se han encontrado en los *logs* de los estudiantes que ya han realizado la práctica.

En esta área se encuentran los estados que representan acciones que han sido validadas como correctas por el tutor automático, es decir, acciones que han realizado los estudiantes cuyas precondiciones se han cumplido. Estos estados están representados gráficamente por círculos blancos en la figura 6.4. Por ejemplo, los estados 1, 2 y 3 que son el resultado de los eventos “hacer 1”, “hacer 2” y “hacer 3” respectivamente.

Asimismo, en esta área se pueden encontrar acciones que forman parte del proceso correcto de la práctica, pero que se han ejecutado prematuramente y al ser validadas dan lugar a eventos de error. Eso ocurre, por ejemplo, con el evento “hacer 5” que sale del estado 2 en la figura 6.4. Esta acción debería ejecutarse después de la 3 y la 4, pero algún estudiante la ha ejecutado antes de tiempo o no sabía que tenía que ejecutar la 3 y la 4 antes. Esto se representa en el autómata mediante la transición que va desde el estado 2 al estado 5. Como se explicará más tarde, como resultado de la validación de la acción “hacer 5”, se generan los eventos “fallar 3” y “fallar 4”, ya que el tutor automático considera que el alumno ha olvidado realizarlas.

Por otro lado, en esta zona también pueden aparecer acciones que en el momento en el que fueron ejecutadas fueron dadas por válidas, pero posteriormente resultaron incompatibles con otras acciones que ejecutó el estudiante. Un ejemplo de este tipo de acciones sería la acción “hacer AC”, que se puede encontrar en el autómata de la figura 6.8.

Existen otro tipo de acciones que pueden estar presentes en los *logs* de los estudiantes llamadas **acciones intrascendentes**, las cuales no influyen en el desarrollo de la práctica. Este tipo de acciones por defecto no son consideradas en la creación del modelo, a menos que en su configuración esté detallado que dicha acción es de importancia pedagógica, en cuyo caso se las tratará como acciones correctas y por lo tanto se encontrarán en esta

zona. Un ejemplo de este tipo de acciones intrascendentes es el tomar un recipiente que luego no es necesario en la práctica.

En esta zona se presenta el estado final, simbolizado en la figura 6.4 por dos círculos concéntricos. Como ya se ha explicado anteriormente, este estado valida la secuencia como correcta, por consiguiente, la frecuencia de alumnos presente en este estado corresponde a la cantidad de ellos que han concluido todo el proceso de entrenamiento correctamente.

6.3.2. Zona de errores irrelevantes

Esta zona agrupa estados derivados de eventos de error que no influyen en el resultado final. Estos errores están asociados con intentos de acción bloqueados por el tutor. Los estados de esta zona están representados gráficamente en la figura 6.4 por círculos amarillos.

Los eventos que conducen a un estado de esta zona son transiciones normales, es decir, transiciones que poseen únicamente la frecuencia de estudiantes que lo han provocado, mientras que las transiciones que salen de estos contienen un vector de frecuencias, por ello se denominan **transiciones vectoriales**. El vector en una transición de este tipo contiene el porcentaje de estudiantes que han caído en el error un número i veces. En otras palabras, cada posición del vector contiene una frecuencia de alumnos que especifica cuántos de ellos han ejecutado la acción errónea el número de veces que señala el índice de dicha posición. De esta manera, la primera posición del vector (posición 0) contiene la frecuencia de estudiantes que han ejecutado una sola vez la acción incorrecta (han repetido 0 veces el error), mientras que las demás posiciones contienen las frecuencias de estudiantes que han repetido dicho error al menos una vez (han repetido i veces el error). Esto se debe a que al ser considerados este tipo de errores como irrelevantes, pueden ser provocados varias veces consecutivas por un mismo estudiante. En otras palabras, si el estudiante realiza varias veces seguidas la misma acción, el evento de error se reproducirá en cada ejecución. Por esto, los estados de esta zona contienen transiciones hacia sí mismos para representar el efecto de bucle.

Un ejemplo representado en la figura 6.4 es el estado amarillo 2, al cual han llegado el 40 % de estudiantes al intentar realizar la acción 3 antes de la 2. De este 40 %, 10 % no han repetido dicho error, el 30 % han repetido “intentar 3” una vez, el 45 % dos veces y el 15 % tres veces.

Los eventos de error que se presentan en esta zona son aquellos cuya ejecución ha sido **bloqueada** por el tutor automático. Es decir, cuando el alumno realiza una acción y ésta es validada como errónea, el tutor puede permitir que el estudiante continúe con el entrenamiento como si “aprobara” dicha acción; o por el contrario, puede informar del

error y “bloquear” al estudiante en su entrenamiento hasta que éste ejecute una acción correcta. El tutor automático toma esta decisión en base a la configuración que haya dado el profesor al error, la cual depende de razones pedagógicas.

Los tipos de errores registrados por el tutor automático que se pueden encontrar en esta zona son:

- Errores de dependencia bloqueantes: errores que se producen por acciones previas no realizadas.
- Errores de incompatibilidades bloqueantes: errores que se producen por acciones posteriores que se han ejecutado antes de lo debido.
- Errores de tiempo bloqueantes (máximo de tiempo caducado y mínimo de tiempo necesario): errores que se producen si se incumplen los temporizadores para cuando una acción tiene configurado un tiempo mínimo para terminar su ejecución antes de que el alumno realice el siguiente paso, o un tiempo máximo para que se ejecute la siguiente acción.
- Errores de mundo: errores que se producen cuando el estudiante interactúa de forma incorrecta con el entorno de entrenamiento, por ejemplo, colocar un objeto en otro lugar.
- Errores de acción ya realizada o no encontrada.: errores que se producen cuando un estudiante intenta ejecutar una acción que ha realizado anteriormente, o una que no es parte del proceso de entrenamiento seleccionado al inicio.

6.3.3. Zona de errores relevantes

Esta zona, a diferencia de la anterior, agrupa estados asociados a errores que sí influyen en el resultado final del proceso de entrenamiento. Es decir, los errores que no son evitados por el tutor automático y por lo tanto permite que estudiante los cometa y prosiga con el proceso de entrenamiento. A consecuencia de estos errores el estado final estará asociado a un resultado erróneo.

Los estados derivados de eventos de errores de esta zona están representados gráficamente en la figura 6.4 con círculos rojo (tendrán diferentes tonalidades de rojo dependiendo del tipo de error) y los estados derivados de acciones correctas que se ejecutaron desde estados erróneos con círculos anaranjados. Eso significa que un estado anaranjado es un estado erróneo, no porque se derive directamente de un evento de error, sino porque arrastra un error cometido con anterioridad por el alumno.

Un ejemplo de flujo en esta zona representado en la figura 6.4 comienza con el evento de error “fallar 5”, el cual se produce cuando el estudiante realiza la acción 6 (“hacer 6”) después de la 4, es decir, omite la acción 5 (“hacer 5”). Este evento conduce a un estado rojo 5 desde el cual un 80 % de los estudiantes que cometieron este error (un 60 % del total) ejecutan correctamente la acción 7, que conduce al estado anaranjado 7.

Los estados de error se derivan de la última acción correcta, ya que al momento de ser validada ésta por el tutor es cuando se que encuentra los errores en sus pre y post condiciones. Por ejemplo, en la figura 6.4, el estado 5 rojo se encuentra después del estado 6 blanco ya que en la configuración de la acción 6 se encuentra como pre condición haber hecho la acción 5 y al no haberla encontrado se genera el evento “fallar 5” que lleva al estado de error 5. De igual manera del estado 5 blanco se generan los eventos “fallar 3” y “fallar 4”.

Además, en la figura 6.4 se puede apreciar que existen dos estados 7: uno blanco que representa un estado óptimo al que se ha llegado sin ningún error; y otro anaranjado que representa un estado erróneo al que se ha llegado tras ejecutar una acción correcta (“hacer 7”) que se había visto precedida por al menos un error (“fallar 5”). Además, existe un tercer estado final (6 rojo), el cual se produce al momento de validar los errores cometidos en la acción 7 para los alumnos que no han realizado la acción 6.

Un error relevante puede ser corregido mediante una **acción reparadora**, las cuales redirigen al flujo correcto, siempre que se hayan corregido todos los errores relevantes cometidos por el estudiante hasta el momento. Ahora bien, si una acción reparadora no corrige todos los fallos, se la considera un estado erróneo, porque todavía arrastra algunos errores previos, y por consiguiente se mantendrá en esta zona errónea. Como ejemplo de evento de reparación en la zona de errores relevantes de la figura 6.4 se encuentra el evento “reparar 3”, que produce el estado Z. Este estado se encuentra en esta zona y es anaranjado debido a que no ha corregido todos los errores, es decir, solo ha corregido el error 3 y no el 4 que han sido detectados al momento de validar la acción 5 (estado 5 blanco). En cambio, el evento “reparar 5” sí que repara todos los errores encontrados en la validación de la acción “hacer 6”, y por consiguiente produce un estado correcto (Y) en la zona del flujo correcto.

Los eventos de error que se presentan en esta zona son aquellos cuya ejecución “no ha sido bloqueada” por el tutor automático (como se explicó anteriormente). Los tipos de errores registrados por el tutor automático que se pueden encontrar en esta zona son:

- Errores de dependencia no bloqueantes.
- Errores de incompatibilidades no bloqueantes.

- Errores de tiempo no bloqueantes (máximo de tiempo caducado y mínimo de tiempo necesario).

6.3.4. Definición formal del autómata

Basándose en la definición de un autómata finito determinista (sección 2.3.4) el autómata extendido del modelo de comportamiento colectivo del estudiante se puede definir matemáticamente de la siguiente forma:

$$EA = (S, S_C, S_{IE}, S_{RE}, S_{CE}, Z_C, Z_{IE}, Z_{RE}, s_0, s_f, T_N, T_V, \delta, \gamma, \phi_N, \phi_V) \quad (6.3)$$

En la ecuación 6.3, S representa el conjunto formado por todos los estados. Los conjuntos de estados de las tres zonas detalladas en las secciones anteriores están definidos por Z_C , Z_{IE} y Z_{RE} . s_0 es el estado inicial del autómata. s_f corresponde al estado final del autómata. T_N y T_V representan los conjuntos de transiciones, es decir, T_N comprende las transiciones normales, mientras que T_V las transiciones vectoriales. δ es la función de transición de un estado a otro, γ la función que asocia a un estado con sus frecuencias, mientras que ϕ_N y ϕ_V las funciones que definen las frecuencias de las transiciones, la primera para las transiciones normales y la segunda para las vectoriales.

$$\begin{aligned} Z_C &= S_C \\ Z_{IE} &= S_{IE} \\ Z_{RE} &= S_{RE} \cup S_{CE} \\ S &= S_C \cup S_{IE} \cup S_{RE} \cup S_{CE} \\ S_C \cap S_{IE} \cap S_{RE} \cap S_{CE} &= \emptyset \end{aligned} \quad (6.4)$$

La ecuación 6.4 define en detalle cada zona del autómata definido en la ecuación 6.3, donde:

- Z_C define la zona de flujo correcto. Como se detalló anteriormente esta zona contiene estados pertenecientes al conjunto de estados correctos representados por el conjunto S_C (estados blancos en figura 6.4).
- Z_{IE} define la zona de errores irrelevantes. La cual contiene estados de error del mismo nombre, definidos por el conjunto S_{IE} (estados amarillos en la figura 6.4).
- Z_{RE} es la zona de errores relevantes. La cual a diferencia de las otras contiene dos tipos de estados S_{RE} (estados rojos en la figura 6.4) y S_{CE} (estados anaranjados

en la figura 6.4). S_{RE} es el conjunto de estados que se derivan directamente de errores relevantes, mientras que S_{CE} es el conjunto de estados que son el resultado de ejecutar una acción correcta después de un estado ya perteneciente a esta zona Z_{RE} , si la acción es reparadora pertenece a este grupo siempre y cuando no haya corregido todos los errores del estudiante.

La función δ es definida por la ecuación 6.5, donde s_i es el estado inicial del que parte la transición y s_f el estado final (al que llega la transición).

$$\begin{aligned}\delta : T &\rightarrow SxS \\ \delta(t) &= (s_i, s_f)\end{aligned}\tag{6.5}$$

La función γ está definida para cada estado en S estableciendo de esta manera la frecuencia de cada estado, de esta manera agregamos pesos a los estados del autómata, extendiendo así la definición clásica de los autómatas finitos deterministas.

$$\begin{aligned}\gamma : S &\rightarrow \mathbb{N} \\ \gamma(s) &= f\end{aligned}\tag{6.6}$$

De forma similar a la función γ , las funciones ϕ_N y ϕ_V proveen de frecuencias pero en este caso a cada transición t , extendiendo de esta forma la definición de una transición en un autómata finito determinista.

$$\begin{aligned}\phi_N : T_N &\rightarrow \mathbb{N} \\ \phi_N(t) &= f\end{aligned}\tag{6.7}$$

$$\begin{aligned}\phi_V : T_V &\rightarrow \mathbb{N}^n \\ \phi_V(t) &= f^n\end{aligned}\tag{6.8}$$

6.4. Ejemplos de aplicabilidad del autómata extendido

A continuación, se presentan dos ejemplos prácticos de procesos en los que se puede aplicar el autómata para representar su secuencia de pasos, junto con algunos posibles errores que se pueden cometer en ellos. El primero es un proceso cotidiano de mantenimiento de bujías en un motor de combustión interna, y el segundo es un fragmento del protocolo del laboratorio de biotecnología descrito anteriormente.

6.4.1. Autómata para el mantenimiento de bujías

El objetivo de este proceso es el mantenimiento de bujías en un motor de combustión interna, el cual es conocido y muy utilizado en la actualidad.

El proceso detallado se presenta en la figura 6.5, en el cual se pueden distinguir varias acciones que son indispensables antes de iniciar la limpieza como verificar que el motor esté apagado y frío. Acciones que pueden ser omitidas por error, pero que luego impedirán alcanzar el objetivo final del proceso.

Figura 6.5 Ejemplo de proceso de limpieza y cambio de bujías.

En la figura también se puede apreciar que existen tres posibles soluciones correctas, una cuando la bujía tiene grietas, ya que si las tiene, no se puede reutilizar la bujía y hay que reemplazarla. Las otras dos posibles secuencias correctas se producen si la bujía vieja no tiene grietas, en cuyo caso se puede proceder con la limpieza o sustituirla por una nueva.

Por ejemplo, si se plantea como ejercicio que la bujía del motor que se desea limpiar tiene grietas, y el alumno decide no sustituirla y procede con su limpieza, el resultado del proceso de entrenamiento será entonces erróneo. Para el presente ejemplo se ha decidido tomar la solución de este ejercicio como flujo correcto, ya que favorece la creación de un ejemplo de autómata con los diferentes tipos de errores que se desean ilustrar.

La figura 6.6 presenta un ejemplo de autómata del MCCE creado en base a la secuencia de pasos de la figura 6.5. Este autómata se ha creado pensando en el uso que le han dado los estudiantes a un supuesto entorno de entrenamiento en el que se enseña cómo proceder con el mantenimiento de las bujías de un motor con inyección interna.

Figura 6.6 Ejemplo de autómata del proceso de la figura 6.5.

El resultado de ejecutar cada acción del proceso de la figura 6.7 se encuentra representado en el autómata con un estado con su numeral. Es decir, la acción “verificar que el motor esté apagado” que en la imagen del proceso posee el numeral 1, en el autómata se ha representado con el estado con el mismo número.

Cabe señalar que al ser éste un autómata extendido, tanto los estados como las transiciones poseen frecuencias del número de estudiantes que han llegado a un estado y han realizado un evento representado por la transición, respectivamente. Sin embargo, se muestran solamente las frecuencias de los estados y transiciones relevantes para el ejemplo.

Se puede apreciar en la figura 6.6 como los eventos asociados con acciones validadas como correctas por el tutor automático producen estados correctos. Por ejemplo, el evento “hacer 2” (verificar que el motor esté frío) conduce al estado 1 de color blanco con un 80 % de frecuencia. Esto quiere decir que la mayoría de los alumnos, que iniciaron el proceso de creación de la mezcla, verificaron que el motor estuviera frío. Por otro lado, los eventos que llevan a estados anaranjados (por ejemplo “hacer 4” desde el estado rojo 2) también son validados como correctos por el tutor. Sin embargo, se encuentran en la zona inferior (zona de errores relevantes) debido a que arrastran un error relevante (como se explica en la sección 6.3.2) realizado con anterioridad. En este sentido, en la figura 6.6 se muestra una secuencia de estados anaranjados que se corresponde con una secuencia de acciones correctas, que empieza en el estado anaranjado 4 y acaba en el estado anaranjado 12.

A diferencia de los eventos correctos, los eventos de error dirigen a estados en las zonas de errores irrelevantes o relevantes dependiendo del tipo de fallo. Por ejemplo, el evento “intentar 2” (error irrelevante) ejecutado por el 20 % de estudiantes desde el estado inicial, conduce a un estado 1 de color amarillo. Esto quiere decir que este porcentaje de alumnos han intentado verificar que el motor estuviera frío sin antes apagarlo. Ahora bien, al ser un error poco común y fácil de descubrir, la mayoría de estos estudiantes solo lo comenten una vez (80 % en la posición cero de la transición vectorial).

Otro ejemplo de error irrelevante es el estado “AR”, que se refiere a un error de una acción ya realizada (Riofrío-Luzcando et al. 2012; Rico et al. 2012), es decir, cuando un alumno intenta ejecutar una tarea previamente completada por él/ella mismo. En el ejemplo de la figura, un 5 % de alumnos intentan realizar de nuevo la acción 1 después de haber ejecutado la 2 (*hacer 1* $\rightarrow$ *hacer 2* $\rightarrow$ *intentar 1*), produciendo de esta manera el error.

Los errores relevantes están representados en la figura 6.6 por medio de estados de color rojo. Este es el caso del estado rojo 2, que es el resultado del evento de error “fallar 2”

producido al momento de validar la correcta ejecución de la acción 3. Si en esa validación se detecta que el alumno ha ejecutado la secuencia *hacer 1* $\rightarrow$ *hacer 3*, se genera dicho error, porque se detecta que previamente no se ha ejecutado la acción 2. En este ejemplo, se supone que la configuración del tutor automático permite la ejecución de la acción 3 pese al error, pero el tutor registra el evento de error “fallar 2”, que posteriormente será mostrado al estudiante.

Otros ejemplos de este tipo de errores son los producidos al momento de validar la acción 12, en donde vamos a suponer que el tutor automático verifica que se ha seguido la secuencia correcta (*hacer 8.2* $\rightarrow$ *hacer 11*), es decir, que se ha detectado que la bujía tiene grietas y posteriormente se ha sustituido. Como se puede apreciar en la figura 6.6, del estado 12 se deriva una secuencia de fallos *fallar 8.2* $\rightarrow$ *fallar 11* producidos porque varios alumnos no realizaron las acciones correspondientes a dichos fallos.

Además de la secuencia de errores antes mencionada, del estado 12 se deriva otra secuencia adicional de fallos *fallar 8.1* $\rightarrow$ *fallar 9* $\rightarrow$ *fallar 10*, ya que, según el ejemplo, varios alumnos, además de no ejecutar la secuencia correcta (*hacer 8.2* $\rightarrow$ *hacer 11*), realizaron en su lugar una secuencia de acciones incorrectas: decidieron que la bujía no tenía grietas (“fallar 8.1”) y procedieron a limpiarla (“fallar 9” y “fallar 10”). Estas tres acciones son incompatibles con el resultado correcto del ejercicio.

En la figura 6.6 se pueden ver los soportes del estado rojo de error 2 y el estado rojo de error 3. Los soportes de estos estados indican respectivamente que: un pequeño grupo de estudiantes se han olvidado de verificar que el motor esté frío (10 %), porque han ejecutado la secuencia *hacer 1* $\rightarrow$ *hacer 3*; y que la mayoría no desconectaron la batería antes de desconectar el cable y retirar la bujía (80 %), porque han ejecutado una de las siguientes secuencias *hacer 1* $\rightarrow$ *hacer 4* o *hacer 1* $\rightarrow$ *hacer 2* $\rightarrow$ *hacer 4*.

Los siguientes ejemplos de errores son más importantes pedagógicamente en el proceso de entrenamiento que los anteriores (estados rojos 2 y 3), puesto que en este entorno se intenta que el estudiante aprenda a identificar una bujía con grietas y que además la reemplace, ya que, de lo contrario, se podría producir un fallo del motor. Se puede apreciar que un 70 % no ha encontrado la grieta de la bujía (estado rojo oscuro 8.2). A pesar de esto, un 10 % de estos alumnos sustituyó la bujía, ya que los estudiantes que ejecutaron la acción 11 son un 40 %, esto es, un 10 % más de los que sí encontraron grietas en la bujía (8.2 blanco). Esto último quiere decir que, del 70 % que cometieron el error anterior un 60 % continuaron con la secuencia errónea, mientras que un 10 % no. Por esta razón el estado 12 posee una frecuencia del 40 % de estudiantes en el clúster, pero los estudiantes que han completado exitosamente el ejercicio son solamente un 30 % (soporte del estado final).

Como se puede apreciar, estas frecuencias ayudan a detectar no solo posibles caminos más comunes localizando las transiciones más probables desde un estado, sino también estados frecuentes debido al alto número de estudiantes que han llegado a él. En otras palabras, no solamente es importante la confianza de un estado sino también su soporte.

Viendo estos ejemplos, tanto un tutor automático como uno humano podrían decidir dar una mejor ayuda al estudiante. Por ejemplo, se puede recordar a los alumnos que es importante tomar medidas de seguridad, para intentar que verifiquen que el motor esté frío y que desconecten la batería. Así mismo, se podría dar un mensaje posterior a realizar la acción 7 que diga que una bujía con grietas puede dañar al motor, y de esta forma animar a los estudiantes a verificar que la bujía no tenga grietas.

Como se explicó anteriormente, para cada clúster se construye un autómata diferente, por tanto, las frecuencias del ejemplo presentado en la figura 6.6 podrían ser diferentes a las de otro autómata de otro clúster. Por ejemplo, el estado de error 3 podría no tener una frecuencia tan alta, o de hecho podría no existir. Esto justifica que se pueda proveer de diferente tutoría a los estudiantes dependiendo del clúster al que pertenezcan.

6.4.2. Autómata para el laboratorio virtual de biotecnología

Como segundo ejemplo para explicar cómo se puede aplicar el autómata, se ha tomado un fragmento del protocolo del laboratorio virtual de biotecnología, es decir, una sub-

Figura 6.7 Ejemplo del proceso del laboratorio virtual de biotecnología.

secuencia de pasos del proceso de entrenamiento (Riofrío-Luzcando et al. 2012; Rico et al. 2012) que tienen que llevar a cabo los alumnos.

La figura 6.7 representa una versión simplificada del proceso de creación de la mezcla en la cual deben crecer los árboles de chopo, el cual, a su vez, forma parte del proceso de preparación del medio de enraizamiento.

Como se explica en la sección 3.3.1, y como se muestra en la figura 6.7, la mayoría de las acciones deben ser realizadas en un orden específico, pero existen algunos grupos de acciones en los que no importa en qué orden sean ejecutadas las acciones siempre y cuando sean ejecutadas todas. En la figura todas las acciones deben seguir un orden estricto a excepción de las acciones 3 y 4 que pueden ser realizadas también en orden inverso (4 y 3), siempre y cuando sean ejecutadas después de la acción 2 y antes de la 5.

Figura 6.8 Ejemplo de autómata del proceso de la figura 6.7.

La figura 6.8 muestra un posible autómata derivado del proceso de la figura 6.7. No se muestran todas las frecuencias en aras de la claridad de la imagen, sino solamente aquellas necesarias para explicar el ejemplo.

De igual manera que en el ejemplo anterior, se puede apreciar en la figura 6.8 como los eventos asociados con acciones validadas como correctas por el tutor automático producen estados correctos. Por ejemplo, el evento “hacer 1” (soltar vaso en el agitador) conduce al estado 1 de color blanco con un 100 % de frecuencia, lo cual quiere decir que todos los alumnos que iniciaron el proceso de creación de la mezcla, soltaron el vaso en el agitador. Otro ejemplo es el evento “hacer 4” (agregar sacarosa) desde el estado 3 con una frecuencia del 20 %, lo cual indica que ese porcentaje de alumnos del total que han llegado hasta el estado 3 agregaron sacarosa a la mezcla correctamente.

En este caso, el estado “AR” aunque también se refiere a un error de una acción ya realizada como en el ejemplo anterior (Riofrío-Luzcando et al. 2012; Rico et al. 2012), la secuencia en la que se produce el error es más compleja, debido a que las acciones 3 y 4 pueden ser realizadas en cualquier orden (*hacer 3* $\rightarrow$ *hacer 4* o *hacer 4* $\rightarrow$ *hacer 3*). De esta manera, en el autómata se aprecia que pocos alumnos desean realizar de nuevo la acción 4 (“intentar 4”) habiéndola realizado anteriormente justo después de la 2 (*hacer 2* $\rightarrow$ *hacer 4* $\rightarrow$ *hacer 3*). En otras palabras, se olvidan de que ya han agregado la sacarosa anteriormente.

La acción “agregar caseína” (AC), que no está incluida en la figura 6.7, representa una acción no presente en el proceso antes mencionado, que se encuentra disponible en el entorno virtual para dotarlo de realismo, ya que en un laboratorio real pueden existir otros compuestos químicos aparte de los necesarios para el cumplimiento del proceso deseado. Además, también así se podrá comprobar si el estudiante sabe qué debe agregar a la mezcla y no se tendrá que limitar a agregar todo lo que se le facilita, lo cual es interesante desde un punto de vista pedagógico. Como esta acción existen otras más y todas ellas son consideradas como “incompatibles” (Riofrío-Luzcando et al. 2012; Rico et al. 2012) con alguna acción correcta del proceso de entrenamiento. Concretamente en este ejemplo la acción AC es incompatible con la acción 3. En general, este tipo de acciones se le permiten realizar o no al alumno dependiendo de la configuración de la estrategia de tutoría.

Un ejemplo de nodo producido por un error relevante en la figura 6.8 es el estado rojo AC que es el resultado del evento de error “fallar AC” producido al momento de validar la correcta ejecución de la acción 3. Como se explicó en la definición del autómata, las siguientes acciones correctas ya no lo son del todo, pues arrastran un error cometido con anterioridad. Este es el caso de los estados anaranjados 4, 5, 6 y 7 que se derivan de acciones ejecutadas después del error AC. Asimismo, se puede apreciar que varios estudiantes terminan el proceso cometiendo solamente ese error.

Otro ejemplo de error relevante es el evento “fallar 5” que produce el estado rojo 5, y que se detecta al validar la acción 6 o la 7. Este evento es generado por no realizar la acción 5 del proceso, es decir, debido a que algunos alumnos agregaron agarosa o apagaron el agitador después de haber agregado la sacarosa (acción 6 o 7 después de la 4). Para el primer caso se produce solamente el “fallar 5” (*hacer 6* $\rightarrow$ *fallar 5*). En cambio, para el caso de los alumnos que han realizado después de la acción 4 directamente la 7 (olvidando dos acciones correctas) no solo se produce el “fallar 5” sino también el “fallar 6” consecutivamente, produciendo el camino *hacer 7* $\rightarrow$ *fallar 5* $\rightarrow$ *fallar 6*.

Las confianzas en la figura 6.8 muestran que varios alumnos han olvidado encender el agitador antes de intentar agregar sustancias a la mezcla, es decir, el 40 % intentaron realizar la acción 3 (“intentar 3”) antes que la 2. Otro ejemplo se presenta en las transiciones que salen del estado 3, las cuales indican que: un 20 % de los estudiantes realizan la acción 4 después de la 3; un 10 % intenta la 4 cuando ya la ha hecho anteriormente; otro 10 % no cae en este error y ejecuta la 5; y una gran cantidad de alumnos (60 %) han cometido el error de agregar caseína.

Para el caso de los soportes en los estados, la figura 6.8 presenta tres ejemplos: estado de error AC, estado de error 5 y estado de error 6. En el primer y segundo caso significa que un alto número de estudiantes han llegado a esos estados, es decir, que un 60 % del total de alumnos en el clúster han agregado caseína a mezcla y un 70 % no ajustaron el pH, mientras que en el tercer error denota que un 40 % de estos no agregaron agarosa a la mezcla. Como existe un camino que conecta los tres errores, eso implica que varios alumnos han cometido dos o tres de los errores.

6.5. Proceso de construcción del autómata

El proceso de construcción del autómata de cada clúster está detallado en el algoritmo 1. Este algoritmo recibe como entradas los *logs* de estudiantes pertenecientes a un clúster y el identificador de dicho clúster. El proceso de creación obtiene cada evento registrado en cada *log* en orden cronológico, posteriormente busca si existe ya un estado en el autómata y una transición desde el estado anterior que concuerde con el evento. Si el estado y/o la transición no existen, se crean con una frecuencia inicial de 1, y en caso contrario, simplemente se aumentan sus frecuencias.

Para poder decidir si un estado o una transición existente en el autómata se corresponde con el evento actual en el proceso, estos contienen información acerca del evento al que representan. Ahora bien, con el fin de facilitar la comprensión del algoritmo 1, se obviarán este tipo de detalles de implementación en la especificación del mismo.

Algoritmo 1 Proceso de construcción del autómata

Entradas

Entrada *logs de estudiantes*

Entrada *cluster*

Salidas

Salida *EA* (véase la ecuación 6.3)

for each *estudiante* $\in$ *cluster* **do**

$s_i = s_0$

Leer *log* de *estudiante* desde *logs de estudiantes*

for each *evento* $\in$ *log* **do**

AGREGAREVENTOLOG(*evento*, s_i)

end for

end for

procedure AGREGAREVENTOLOG(*evento*, s_i)

if $\exists s_f \in S$ | s_f es resultado de *evento* **then**

$\gamma(s_f) = \gamma(s_f) + 1$ (véase la ecuación 6.6)

if $\exists t \in model$ | $\delta(t) = (s_i, s_f) \wedge t$ concuerda con *evento* **then**

if $t \in T_V$ **then**

$count$ = número de eventos consecutivos igual a *evento* en *log*

$\phi_V(t)_{[count-1]} = \phi_V(t)_{[count-1]} + 1$

(véase la ecuación 6.8)

Quitar eventos consecutivos iguales a *evento* desde *log*

else

$\phi_N(t) = \phi_N(t) + 1$ (véase la ecuación 6.7)

end if

else

AGREGARTRANSICIÓN(s_i, s_f)

end if

$s_i = s_f$

else

Agregar estado nuevo s a S (S es elemento de EA).

AGREGARTRANSICIÓN(s_i, s)

$s_i = s$

end if

end procedure

procedure AGREGARTRANSICIÓN(s_i, s_f)

if $s_i \in S_{IR}$ **then** (véase la ecuación 6.5)

Agregar una nueva t_V a T_V desde s_i hasta s_f usando $\delta(t)$

(T_V es elemento de EA)

else

Agregar una nueva t_N a T_N desde s_i hasta s_f usando $\delta(t)$

(T_N is element of EA)

end if

end procedure

6.6. Proceso de actualización del modelo

El MCCE se actualiza con cada nuevo evento producido por cada estudiante que se encuentra utilizando el entorno de entrenamiento. De esta manera, durante una sesión de aprendizaje, el modelo se va adaptando mejor a los estudiantes que están utilizando el entorno de entrenamiento, lo cual permite mejorar la precisión de las predicciones del modelo. En otras palabras, el modelo es capaz de aprender para adaptarse mejor a los cambios que se vayan produciendo en los estudiantes de los sucesivos cursos.

Para esto, este proceso primero busca el estado que resulta del último evento ejecutado por el alumno. Esta búsqueda se realiza en el autómata del clúster al que pertenece el dicho estudiante. Posteriormente, desde este estado, se crea una nueva transición o se incrementa la frecuencia de una ya existente de una forma similar que a como se realiza en el proceso de creación del autómata. Es decir, esta actualización se procesa de la siguiente manera:

1. Buscar el estado al que ha llegado el estudiante s_i que se deriva del último evento generado por éste. Si se trata del inicio del proceso de entrenamiento, el estado actual se define como el “estado 0”.
2. Ejecutar el procedimiento *AgregarEventoLog* del algoritmo de 1, donde *evento* es el evento con el que se quiere actualizar el modelo y s_i es el estado actual.

6.7. Proceso de reclasificación de un estudiante

Cuando un alumno inicia la actividad de entrenamiento, se le asigna a un clúster por defecto. Este clúster es aquel que tiene los valores promedio en los atributos utilizados para el cálculo de los grupos en el proceso de creación del MCCE, es decir, el **clúster promedio**. Sin embargo, a medida que el estudiante avance en el proceso de dicho entrenamiento, es posible que esta clasificación por defecto no se corresponda con su comportamiento. Por esta razón podría ser necesario reclasificar al alumno en diferentes momentos de su adiestramiento. Los momentos en los que se comprobará si es necesario o no reclasificar a un alumno recibirán el nombre de **puntos de control**.

Los puntos de control pueden ser configurados inicialmente por un profesor, o pueden ser especificados con diferentes porcentajes del promedio del tiempo que les ha tomado concluir la actividad a los estudiantes pertenecientes al mismo clúster.

Cuando el alumno alcance un punto de control, será reclasificado teniendo en cuenta su registro de acciones realizadas hasta ese momento. Este proceso no implica simplemente

trasladar al estudiante de un grupo a otro. Conlleva también modificar el autómata del clúster que abandona y el autómata del clúster al que llega, ya que sus acciones han ido actualizando el autómata de un clúster al que va a dejar de pertenecer. Para esto seguimos el siguiente proceso:

1. Obtener el clúster actual al que pertenece el estudiante.
2. Seleccionar el clúster al que mejor se ajusta el registro de acciones realizadas hasta el momento por el estudiante.
3. Si el clúster que mejor encaja es el actual, no hacer nada.
4. En caso contrario
 - a) Deshacer en el autómata del clúster actual las modificaciones que se han derivado de cada acción que ha realizado el estudiante, ejecutando el proceso opuesto al que realiza el algoritmo de creación 1, es decir, disminuir frecuencias y/o eliminar estados y eventos.
 - b) Aplicar las acciones realizadas por el estudiante al autómata del nuevo clúster de acuerdo al proceso de creación (algoritmo 1).

Para obtener el clúster al que mejor se ajusta el comportamiento del estudiante hasta el punto de control, utilizamos los mismos parámetros que se utilizaron al inicio para crear los grupos. Es decir, obtenemos los *logs* de los estudiantes que pertenecen a un grupo y verificamos si el comportamiento del alumno actual se ajusta a los parámetros definidos para ese clúster. Ahora bien, para este proceso únicamente consideramos los fragmentos de los *logs* de los estudiantes que se corresponden con la parte del proceso que el alumno actual ha realizado hasta el punto de control.

Por ejemplo, si se utilizó el agrupamiento por errores para la generación de los clústeres (sección 6.8), se verifica si el número de errores cometidos por el estudiante actual bajo supervisión se encuentra entre el valor máximo y mínimo de errores realizados por los estudiantes de cada clúster hasta el punto de control. Si dos grupos concuerdan con esta premisa, se toma aquel en el que la media del número de errores se acerque más al del actual alumno. Este proceso es similar al utilizado en las técnicas de *Incremental Clustering* (Rokach & Maimon 2005).

6.8. Funciones de Agrupamiento

En el presente trabajo hemos considerado la utilización de dos métodos para crear los clústeres de estudiantes. El primer método se basa en características o parámetros calculados a partir de los *logs* de los estudiantes en el proceso de entrenamiento; y el segundo método se basa directamente en las secuencias de eventos generadas por cada estudiante y guardadas en los *logs*.

Esto no significa que el MCCE deba estrictamente ser creado con uno de estos métodos, como se ha visto anteriormente, su diseño permite que se pueda utilizar cualquier método de agrupamiento, lo único importante es que los alumnos estén divididos en grupos de similares características.

Con el método de agrupamiento según las características calculadas del *log* de un alumno se utilizaron tres funciones alternativas:

- Agrupamiento por errores.
- Agrupamiento por errores y tiempo.
- Agrupamiento por eventos en cada zona del autómata.

En las dos primeras el error asociado con cada *log* es calculado como la suma ponderada de los errores registrados en el *log* utilizando diferentes pesos dependiendo de la importancia pedagógica de cada error. Es decir, si el error es pedagógicamente importante sumará más que uno que no influye, por ejemplo, el error de “agregar caseína” de la figura 6.7 pesará más que el error de “intentar 3”.

La segunda función agrupa en base a dos parámetros, el primero es el error calculado de la misma forma que en la primera función, y el segundo es el tiempo que ha tardado cada estudiante en completar el proceso de entrenamiento. Si un alumno no ha completado la actividad se le penaliza con 24 horas, de esta forma los tratamos como casos especiales.

La última función agrupa atendiendo al contenido de una tripleta de parámetros en la que cada uno de ellos representa el número de eventos de cada tipo (zona del autómata) existentes en el *log* de cada estudiante. Para esto se usa el mismo criterio de clasificación de eventos que se utiliza en la creación del autómata.

Con el método de agrupamiento de secuencias, simplemente se pretende utilizar un algoritmo que agrupe a los alumnos dependiendo de las orden de acciones que han realizado, es decir, como se explica en la sección 2.3.4, crear grupos de alumnos cuyas secuencias de acciones tengan una alta cohesión.

La razón para utilizar estos dos métodos de agrupamiento y varias funciones alternativas para el primer método, es porque se desea demostrar que la agrupación de los

estudiantes, junto con la aplicación de los autómatas, nos permite obtener un mecanismo de predicción de eventos lo suficientemente preciso como para proporcionar una tutoría adecuada para cada tipo de estudiante. Esta cuestión será evaluada en el apartado de validación (capítulo 9), en el que compararemos los resultados obtenidos con estos métodos de clusterización.

CAPÍTULO 7

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

Los patrones creados por la fase de minería de datos del KDD no son útiles por sí solos, puesto que deben ser interpretados por un humano o por otro software, para de esta forma poder sacar partido de los resultados que presenten estos patrones. La forma más utilizada de interpretación de patrones es mediante la presentación de un gráfico a un humano experto en el dominio. A esta técnica de minería de datos se la conoce como destilación de datos para la interpretación por parte de un ser humano (Baker 2010). Básicamente, esta técnica consiste en la presentación de información relevante de una forma resumida, interactiva y visual, con el fin de dar soporte a la toma de decisiones.

Por esta razón, como parte del presente trabajo, se ha diseñado una herramienta para visualizar gráficamente el autómata de cada clúster del MCCE por medio de un grafo. Se ha elegido un grafo, porque los grafos no son solamente útiles como un modelo de patrones secuenciales de minería de datos, sino que también pueden ser adecuados para representar cualquier clase de datos (Tan et al. 2013) ya que pueden mostrar las relaciones que existen entre los objetos.

El objetivo principal de esta herramienta es permitir la identificación de patrones relevantes en el comportamiento de los estudiantes, como, por ejemplo: errores comunes; tareas que les es más difícil realizar que otras; tareas que no saben cómo realizar por lo que se atascan en el procedimiento; etc. Esto puede ser de gran ayuda para el profesor, en la medida en que le puede permitir adaptar su docencia para solventar las deficiencias reveladas por los patrones de comportamiento. Asimismo, estos patrones de comportamiento también pueden ser tenidos en cuenta a la hora de diseñar o cambiar

Figura 7.1 Captura de pantalla del visualizador del MCCE.

la estrategia de tutoría aplicada por un Sistema Inteligente de Tutoría en un entorno virtual de entrenamiento. La utilización de estos patrones en la estrategia de tutoría se explica en detalle en el apartado 9.3.

Por otro lado, desde el punto de vista del diseño del MCCE esta herramienta nos ha ayudado en dos tareas importantes del proceso de desarrollo:

- Agregar mejoras al modelo.
- Depuración de errores.

Al haberse aplicado un desarrollo iterativo para el diseño e implementación del modelo, se vio la necesidad de presentarlo gráficamente para visualizar los resultados de cada iteración y de esta forma detectar características que falten por agregar al MCCE. Además, la identificación de deficiencias en el MCCE mediante métodos tradicionales de depuración fue muy complicada debido a la complejidad de dicho modelo. Para lo cual, la presentación gráfica brindó una gran ayuda, ya que se podían distinguir fácilmente errores tales como estados sin conexiones, estados duplicados, estados que no deberían existir, etc.

Por esto, pensando en la aproximación al usuario final y utilizando técnicas de visualización de información, se concibió esta herramienta para facilitar, en primera instancia, las dos tareas asociadas al desarrollo mencionadas anteriormente, así como la labor docente del profesor y el diseño pedagógico de un entorno virtual de entrenamiento.

Los posibles usuarios de esta herramienta son de dos tipos: para las tareas de implementación somos los propios diseñadores del MCCE; y para las tareas pedagógicas son el profesor de la asignatura o un experto pedagogo del dominio de aprendizaje acompañado del diseñador del Sistema Inteligente de Tutoría.

Los detalles de la implementación de este visualizador se encuentran descritos en una tesis de máster (Huang 2015) y en un trabajo de fin de grado (de Jesús Sen 2016).

Al momento de diseñar e implementar el visualizador, tuvimos que abordar varios problemas que se presentan al visualizar un grafo (Mazza 2009), especialmente, si este grafo es muy grande:

- Posicionamiento y representación de los estados.
- Representación de las aristas.
- Dimensionamiento del grafo.
- Interacción con el grafo.

7.1. Posicionamiento y Representación de los Estados

“En una buena visualización, la posición de los nodos ayudará a agregar información a las aristas, y viceversa” Card, Mackinlay, y Shneiderman (1999).

Comúnmente los grafos representan datos abstractos, es decir, que no tienen una localización espacial natural. Por esta razón, es importante escoger un criterio para distribuir los nodos en el espacio de dibujo dependiendo de su dominio de aplicación (Mazza 2009).

Una posible solución a esta problemática consiste en utilizar variables “espaciales” asociadas con los nodos (Card et al. 1999). Teniendo en cuenta que los estados del MCCE se ubican de una forma abstracta en diferentes zonas, se ha tomado esta propiedad como característica “espacial” de cada nodo. Así, se ha decidido dividir el área de dibujo en las tres zonas presentes en el MCCE (zona de errores irrelevantes, zona de flujo correcto y zona de errores relevantes), en las cuales se posicionan los estados pertenecientes a cada una de ellas (como se muestra en la figura 7.1). Esto es, los estados que pertenezcan a la zona de errores irrelevantes, se dibujan en la zona superior, los del flujo correcto en la de en medio y los que pertenezcan a la zona de errores relevantes en la parte de inferior.

Ya que todos los estudiantes empiezan su entrenamiento desde un estado ubicado en el flujo correcto, y posteriormente la secuencia se puede dirigir a otras zonas dependiendo los errores, se ha posicionado la zona de flujo correcto en la parte central. Esta decisión ayuda también a distinguir mejor los errores irrelevantes de los relevantes.

Además, en nuestro caso, al ser los estados secuenciales, la solución más obvia al problema del posicionamiento de los estados es dibujarlos consecutivamente según el orden cronológico de las acciones en el proceso de entrenamiento. Esta solución funciona únicamente para los estados correctos, es decir, los que se encuentran en la zona de flujo correcto. Esto se debe a que estos estados siempre van a estar en una secuencia estricta. Sin embargo, al incluir los estados de error, esta secuencia ya no se puede dibujar en una sola línea.

Por otro lado, se puede distinguir en la figura 7.1 que los estados correctos (verdes) no se encuentran perfectamente alineados en la horizontal, así como los estados de error (no verdes) tampoco están perfectamente alineados en las verticales. Esta distribución de los estados se implementó para evitar el solapamiento de las aristas que conectan estados que se encuentran en distintas posiciones de una secuencia horizontal. Además, si la disposición por defecto de los estados y las aristas no es suficientemente legible para

el usuario, el usuario puede mover los estados a la posición que éste desee. Para ello, se puede pinchar y arrastrar el estado con el ratón (*drag and drop*).

Otras características a tomar en cuenta al momento de graficar los estados son su forma, color y tamaño. Al asemejarse este modelo a un mapa conceptual (Mazza 2009), es necesario incluir dentro de cada estado el código de la acción que lo provoca. Por lo tanto, se ha adoptado una forma rectangular para los nodos (figura 7.2), porque es la que mejor se adapta a la forma de un texto. La anchura de los nodos se establece de automáticamente dependiendo del tamaño del texto que se debe incluir.

Figura 7.2 Visualización de un estado. flt1 representa al código de la acción de obtener un vaso de precipitados (Riofrío-Luzcando, Ramírez, & Berrocal-Lobo 2012).

En cuanto a su color, distinguimos a los estados por dos características, la zona a la que pertenecen (color del contorno) y su tipo (color del relleno). De esta forma, la zona a la que corresponde un estado, además de reflejarse por la ubicación de éste, se distingue también por el color del contorno de cada nodo de acuerdo a la siguiente gama de colores:

- Zona de flujo correcto (verde).
- Zona de errores irrelevantes (amarillo).
- Zona de errores relevantes (rojo).

Con respecto a la distinción de cada estado por su tipo la gama de colores es la siguiente:

- Estados correctos (verde).
- Estados de errores de dependencia simple (anaranjado oscuro).
- Estados de errores de dependencia compleja (anaranjado).
- Estados de errores de incompatibilidades (rojo anaranjado).
- Estados de errores de tiempo (amarillo).
- Estados de errores de mundo (amarillo oscuro).

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

- Estados de errores de acción ya realizada o no encontrada (lila).

Como se explicó en el capítulo anterior (sección 6.3.3), pueden existir estados correctos dentro de la zona de errores relevantes, ya que las acciones que producen los estados han sido ejecutadas después de haberse producido un evento de error pedagógicamente importante (no bloqueante). Tomando en cuenta el código de colores antes dispuesto, este tipo de estados tienen un relleno de color verde pero su contorno es de color rojo, a diferencia de los estados ubicados en el flujo correcto que son totalmente verdes.

Para facilitar la lectura de los eventos de errores relevantes, aparte de posicionarse en la zona inferior del área de dibujo (zona de errores relevantes), se disponen de forma vertical desde el estado correcto del que se originan dichos eventos de error (véase la figura 7.3); es decir, desde el estado derivado de la acción correcta en la cual el tutor automático, al validar como incorrectas algunas o todas las precondiciones de dicha acción, ha generado errores en los *logs* de los estudiantes. Los estados correctos desde los que salen las transiciones hacia estados de error pueden estar ubicados tanto en la zona de flujo correcto como en la de errores relevantes.

Como se muestra en la figura 7.3, los estados de error por dependencias (anaranjado con contorno rojo) penden del estado `f1t16`. Esto se debe a que la acción que produce este evento posee precondiciones de dependencias en la configuración del tutor automático, que han sido evaluadas como incorrectas para algunos estudiantes, porque previamente no habían ejecutado las acciones `f1t12`, `f1t13`, `f1t14` (`f1t16_f1t12`, `f1t16_f1t13`, `f1t16_f1t14`).

También se puede apreciar en esta figura, que los estados de error se posicionan siguiendo el orden que presentan en la configuración del tutor, que, a su vez, coincide con el orden en el que aparecen los eventos de error en los *logs* de los estudiantes.

Otro ejemplo de estado de error relevante presente en la figura 7.3, es el de un error de incompatibilidad (rojo con contorno rojo), el cual a diferencia de los estados de dependencia, pende de un estado correcto de la zona de errores relevantes. Este estado correcto se deriva de la misma acción `f1t16` del ejemplo anterior, pero arrastra un error cometido con anterioridad, de ahí que se encuentre en la zona de errores relevantes. Esta incompatibilidad, al igual que las dependencias del ejemplo anterior, se encuentra definida en la configuración de la acción `f1t16`.

Si al momento de validar una acción, el tutor encontrara errores de los dos tipos, se dibujan desde la acción correcta primero los estados de errores por dependencias y posteriormente los de incompatibilidades, respetando la secuencia de estados dentro de cada tipo antes mencionada.

Por último, para facilitar la comprensión del grafo en la zona de errores irrelevantes, se decidió agrupar todos los estados de error de acción ya realizada existentes en un

Figura 7.3 Disposición de los estados de errores relevantes desde un estado correcto.

tramo del grafo en un único **superestado**. Es decir, los estados del tipo de error de acción ya realizada que gráficamente aparecerían en un determinado tramo del grafo se agrupan en un único nodo con el nombre *ActionAlreadyPerformed* (figura 7.4) seguido de un número que distingue a este nodo de otros nodos del mismo tipo existentes en otros tramos del grafo. De igual forma se actuó con los estados asociados a errores de acción ya realizada (*ActionNotFound#* figura 7.5).

Figura 7.4 Estado de acciones ya realizadas.

Figura 7.5 Estado de acciones no encontradas.

Esta simplificación se pudo adoptar sin temor a perder información relevante para el profesor debido a que este tipo de errores por separado no tienen importancia pedagógica. Ahora bien, si se consideran en conjunto y tienen una alta frecuencia, podrían indicar que los alumnos están desorientados o que están teniendo dificultades al interactuar con ciertos objetos del entorno virtual.

Aunque estos estados se encuentren agrupados gráficamente, en el MCCE se mantendrán separados para el cálculo de predicciones de comportamiento.

7.2. Representación de las Aristas

Como ya se ha explicado en la sección 2.3.4, las relaciones entre dos nodos de un grafo se representan por medio de aristas, las cuales tienen asociados pesos. Para el MCCE, como ya se explicó en el apartado 6.3, estas aristas representan los eventos provocados tras ejecutar o intentar realizar una acción en el entorno de entrenamiento. De igual forma, como ya se detalló en el capítulo anterior, estos eventos tienen pesos asociados a ellos que representan la frecuencia con la que los estudiantes han producido dicho evento.

En el visualizador los eventos se representan como flechas que tienen como inicio el ultimo estado alcanzado antes de que el evento se produzca, y como final el estado que resulta de dicho evento (figura 7.6).

Figura 7.6 Visualización de un evento (arista).

Los pesos de cada arista se muestran al pasar por encima el puntero del ratón (figura 7.7), lo cual provoca también que cambie de color la flecha (solamente mientras se mantiene el puntero sobre ésta).

Figura 7.7 Visualización del peso de un evento.

Al no presentar por defecto todos los pesos de los eventos, se aligera la cantidad de información que se presenta en la zona de dibujo, y se mejora la legibilidad del grafo. Sin embargo, para distinguir a primera vista los pesos de mayor o menor valor, éstos se pintan siguiendo una escala de grises en función de su frecuencia. De esta manera, cuanto más alta sea esta frecuencia, más oscura será la línea de la arista.

7.3. Dimensionamiento del Grafo

Algunos conjuntos de datos pueden tener una gran cantidad de registros, lo cual complica la representación de todos estos datos a la vez en la pantalla por medio de un grafo (Mazza 2009). La solución que se adoptó para el grafo de nuestro modelo fue la de visualizar solo una parte del grafo en lugar del grafo en su totalidad. El tamaño de esta parte del grafo vendrá determinado por las dimensiones de la ventana del visualizador y el nivel de acercamiento. Es decir, mientras más grande sea la ventana y más alejado sea el nivel de *zoom* más nodos se visualizarán (figuras 7.8a y 7.8b).

De esta forma, la visión preliminar del autómata (Shneiderman 1996) en un inicio dependerá del tamaño de la ventana del navegador, pudiendo llegar a tener una visión global si se disminuye lo suficiente el nivel de *zoom*. Esto permite al usuario definir interactivamente (como se detalla en la siguiente sección) la cantidad de nodos que desea ver dependiendo de sus necesidades. Además, se le permite navegar vertical y horizontalmente (*panning*) por el grafo mediante los botones en forma de flechas existentes a los costados de la zona de dibujo del grafo.

7.4. Interacción con el Grafo

La posibilidad de interaccionar con el grafo es especialmente útil cuando se necesita trabajar con grafos con un gran número de nodos y aristas. En este sentido, funciones como el acercamiento y el filtrado ayudan al usuario a analizar el grafo más fácilmente (Mazza 2009).

La taxonomía TTT de Shneiderman (1996) define varias tareas que los usuarios deben poder realizar en un entorno de visualización de información sin importar el tipo de datos que se presenten. De estas tareas, el visualizador del MCCE implementa las siguientes:

7.4.1. Acercamiento

Esta operación permite ampliar la visualización de un área del grafo para que el usuario pueda centrarse en uno o varios nodos de interés (figura 7.8a), o reducir la visualización de un área del grafo para obtener una visión más global (figura 7.8b), como se indica en la sección anterior.

Esto se efectúa de dos formas, una mediante los botones con forma de lupa que se encuentran al costado izquierdo de la zona de dibujo, y otra utilizando la rueda del ratón. Mediante la primera opción se amplía o se reduce la vista centrando la vista en el centro

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

(a) Ventana pequeña en *Zoom In*

(b) Ventana grande en *Zoom out*

Figura 7.8 Ejemplos de disposiciones del autómata.

de la pantalla, mientras que en la segunda opción se centra la vista ampliada/reducida en la posición en la que se encuentra el puntero del ratón.

Shneiderman (1996) trata este concepto de una forma muy abstracta. Es decir, manifiesta que los usuarios típicamente tienen interés en uno o varios objetos de la colección, por lo tanto, propone que es necesario implementar una herramienta que permita al usuario centrarse en estos objetos de interés. Tomando en cuenta esta definición, podemos afirmar que el visualizador dispone de otra forma de enfoque que consiste en una caja de búsqueda ubicada en la parte superior derecha. Al ingresar el código de una acción del proceso de entrenamiento, seguido del nombre de la región, el visualizador busca el estado asociado a la acción en el grafo. A continuación, mueve la vista del grafo hasta centrar el estado asociado a la acción en el área de dibujo; si la vista se encuentra con un *zoom* muy alejado, la acerca hasta un nivel adecuado para poder leer y distinguir el estado.

(a) Sin filtros

(b) Eventos con frecuencias de más del 20 %

(c) Eventos y estados con frecuencias de más del 20 %

Figura 7.9 Ejemplos de filtros de frecuencias en el autómata.

7.4.2. Filtrado

Como menciona Shneiderman (1996), el filtrado sirve para deshacernos de ítems no interesantes. En este sentido, el visualizador permite filtrar los nodos y/o aristas con frecuencias por dejado de dos umbrales dados. Para ello, se utilizan dos barras de desplazamiento y/o dos cajas de texto situadas en el costado izquierdo.

Como muestran los ejemplos de las figuras 7.9a, 7.9b y 7.9c, al aumentar el valor de los umbrales, el autómatas se vuelve más legible y por consiguiente el usuario puede identificar con más facilidad rutas y estados significativos desde el punto de vista pedagógico. Por ejemplo, en la figura 7.9b se pueden observar únicamente los eventos que han sido generados por al menos un 20 % de los alumnos que han llegado a cada estado. Otro ejemplo es la figura 7.9c, en la cual se pueden encontrar únicamente los estados a los que han llegado al menos un 20 % de los alumnos que han realizado la práctica.

El visualizador, además de la vista global del autómatas, posee dos vistas que filtran la información del modelo que se utiliza para construir el grafo: vista por fechas y vista por estudiante. A estos modos se acceden por medio del menú superior.

Mediante la **vista por fechas** se presenta el modelo con datos registrados dentro de un rango de fechas especificadas por el usuario, es decir, se visualiza el MCCE construido únicamente con registros de alumnos guardados entre dichas fechas. Esta funcionalidad se concibió para poder comparar el comportamiento de alumnos de diferentes cursos académicos.

La **vista por estudiante** presenta la secuencia de eventos realizada por un estudiante específico (figura 7.10). Esta vista ha sido de gran ayuda para la corrección de errores en la fase de desarrollo del MCCE. Además, permite al profesor analizar el comportamiento individual de cada estudiante por separado.

7.4.3. Detalles bajo demanda

Shneiderman (1996) recomienda que los detalles de los objetos sean presentados cuando el usuario lo requiera, es decir, que la información de los objetos que no se presente a primera vista, sea mostrada como respuesta a algún evento generado por el usuario.

El visualizador permite al usuario obtener detalles de cada estado y evento mediante eventos de ratón. En el caso de los estados, cuando un usuario selecciona uno de estos haciendo clic con el ratón sobre el nodo deseado, se presentan en la parte superior del visualizador los siguientes datos de dicho estado:

- Identificador

Figura 7.10 Vista por estudiante del visualizador del MCCE.

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

- Frecuencia de estudiantes
- Área a la que pertenece
- Descripción del estado

La descripción del estado varía dependiendo de su tipo de la siguiente forma, a excepción de los errores de acción no encontrada y ya realizada, los cuales al no ser de importancia pedagógica no presentan ninguna descripción:

- Estado correcto. Se presenta el nombre de la acción que lo provoca.
- Error de mundo. Mensaje de error.
- Error bloqueante o no bloqueante. Tipo de error (dependencia, incompatibilidad) y el mensaje de error.

Las aristas pueden responder a dos eventos de usuario: al pasar el ratón sobre la flecha que representa a una arista, cambia de color y se muestra la frecuencia de estudiantes como indica la figura 7.7; el segundo evento consiste en hacer clic sobre la flecha del evento deseado, lo cual presenta en la parte superior del visualizador la siguiente información de dicho evento:

- Identificador y tipo de evento
- Frecuencia de estudiantes
- Identificador del nodo de salida
- Identificador del nodo de llegada

7.4.4. Relación

Es importante destacar las relaciones entre objetos en el gráfico (Shneiderman 1996). Por esto, el visualizador del modelo permite al usuario identificar la relación que existe entre los eventos y cada estado de una forma sencilla. Para esto, cuando se selecciona un nodo, se pintan de colores sus aristas (como muestra la figura 7.11), diferenciando las aristas que salen del nodo (en púrpura) de los que entran a éste (en azul claro).

Esta funcionalidad puede servir de ayuda al usuario a la hora de distinguir los eventos de un estado, ya que, al existir muchas aristas dentro del grafo, a veces es complicado identificar cuáles son las que se relacionan con el estado de interés.

Figura 7.11 Ejemplos de relaciones (colores) de eventos por estado.

Por ejemplo, en la figura 7.11a no es fácil identificar cuáles son las fechas que se relacionan con cada estado, debido a los cruces que existen entre líneas (por ejemplo la arista que va desde flt8 a flt10). Sin embargo, al comparar las figuras 7.11b y 7.11c se puede diferenciar claramente qué eventos se relacionan con uno y otro estado, así como el sentido de cada evento. En estos casos, como ya se mencionó anteriormente, el visualizar también permite mover los estados con el fin de conseguir una disposición más legible de una parte del grafo.

7.5. Ejemplos de Aplicaciones del Visualizador

Como ya se ha mencionado anteriormente, el visualizador del MCCE es una herramienta en la que los profesores pueden apoyarse para adaptar su docencia teniendo en cuenta el comportamiento de los estudiantes en el entorno virtual de aprendizaje procedimental. Además, puede servir para identificar los pasos del proceso de entrenamiento, en los que se podría mejorar la estrategia de tutoría inteligente.

A continuación, se presentan tres ejemplos en los cuales se explica detalladamente cómo utilizar este visualizador en tareas que podría realizar un profesor en su asignatura.

7.5.1. Calificar a los alumnos

Este ejemplo intenta demostrar la utilidad del visualizador como herramienta para asistir al profesor en el proceso de calificación de un alumno.

Supongamos que en una asignatura el profesor incluye un ejercicio que se debe desarrollar en un entorno virtual de entrenamiento procedimental, como los presentados en el capítulo anterior (véanse las secciones 6.4.1 y 6.4.2). Con ayuda del visualizador el profesor podría analizar no solo el rendimiento global de los alumnos, sino también el individual de cada alumno por separado.

Si tomamos la práctica del laboratorio de biotecnología (del que se disponen *logs* de los alumnos, véase la sección 3.3), observaremos que solamente un estudiante ha terminado el proceso sin errores, para obtener este resultado se realiza el siguiente proceso en el visualizador:

1. Abrir la vista general del MCCE.
2. Sabiendo que la última acción del proceso es la f3t61, escribir este código en el buscador.
3. Seleccionar la que pertenece a la zona del flujo correcto.
4. Finalmente se presiona el botón de búsqueda.

Como se ha explicado ya, el visualizador cambia el foco al estado requerido y presenta su información detallada, en este caso una frecuencia de 1 correspondiente al 1.47% (teniendo en cuenta que el total de alumnos que realizaron el laboratorio es de 68).

Cuando el profesor esté analizando el rendimiento de cada alumno (utilizando la vista por estudiante), puede realizar el mismo procedimiento anterior en esta vista para saber que estudiantes han completado correctamente todo el ejercicio y por esta razón obtengan la mayor calificación.

Realizando este procedimiento con los estudiantes que completaron el laboratorio de biotecnología se encontró que el alumno que ha terminado correctamente el ejercicio es el número 22. Ya que al realizar el procedimiento anterior por cada estudiante, en éste es en el único en el que se encontró el estado f3t61 en flujo correcto, así como también toda la secuencia de estados correctos que preceden a dicho estado (figura 7.12).

Figura 7.12 Secuencia de estados correctos del estudiante 22 del laboratorio de biotecnología.

Para los demás estudiantes se puede analizar cuantos y que tipo de errores han cometido para otorgar una calificación. Es decir, un error de la zona de errores irrelevantes no puede influir en la nota de la misma forma que un error en la zona de errores relevantes.

Para realizar este análisis, el proceso es sencillo: se tiene que navegar por el autómata utilizando los botones situados a los extremos del área de dibujo (o desplazando autómata con el ratón), y en el estado de error que se encuentre hacer clic para leer la descripción del error. Con la posición y el detalle del error el profesor puede deducir si el estudiante ha comprendido o no los conceptos, lo que influirá en su calificación.

Por ejemplo, el estudiante del ejemplo anterior cometió varios errores irrelevantes, pero, como se puede ver en la figura, éstos no influyen en el resultado final. Al contrario que los dos siguientes ejemplos, los cuales, si cometen errores importantes, pero no tendrían la misma calificación.

La secuencia de estados del siguiente estudiante (figura 7.13) es correcta hasta la acción f1t14 (ajustar pH) la cual valida como incorrectas dos dependencias anteriores, es decir, el no haber realizado esas dos acciones (primera circunferencia rojo). La acción f1t20 encuentra además de estos dos errores otro más, de igual forma por una acción no ejecutada (segunda circunferencia rojo). Este estudiante comete un error más (circunferencia amarilla) al no encender la cabina de flujo, lo cual no es un error muy importante como los anteriores que son por no agregar sustancias a la mezcla.

Figura 7.13 Secuencia de estados del estudiante 13 del laboratorio de biotecnología.

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

Lo importante en este ejemplo es que este estudiante comete solamente estos errores en todo el proceso, errores que pertenecen a la primera fase del laboratorio. De esto se puede intuir que el alumno, al visualizar el mensaje final de la fase con la lista de errores no bloqueantes que ha cometido, ha aprendido de sus fallos y ha estudiado mejor las acciones de las siguientes fases del ejercicio. De esta conclusión, el profesor podría otorgarle una buena calificación, aunque no sea la máxima.

Al contrario que el ejemplo anterior, el siguiente estudiante (figura 7.14) no aprende de sus errores y comete varios en todas las fases del laboratorio. Los errores que comete son en los componentes que debe agregar en las mezclas necesarias para cada fase del laboratorio, en las cuales no agrega varios, o ninguno de ellos, como es el caso de la fase 2.

Figura 7.14 Secuencias de estados del estudiante 27 del laboratorio de biotecnología.

Teniendo abierta en otra pestaña del navegador la vista global del MCCE, se puede buscar los errores que ha cometido el estudiante y analizar si son errores frecuentes entre todos los estudiantes. Para este estudiante, solo dos errores superan el 60 % de frecuencia, pero los demás no superan el 30 %.

De este análisis se puede intuir que a este estudiante no le ha interesado estudiar los conceptos otorgados en el proceso del laboratorio, y que solo ha ejecutado las acciones que el tutor automático le ha indicado como hacerlas. Conclusión que ayudaría al profesor para dar una mala calificación a este alumno.

7.5.2. Comparar el comportamiento de los alumnos de diferentes clústeres

Otro ejemplo de aplicación del visualizador del MCCE es para comparar las secuencias de acciones realizadas por cada tipo de estudiante, y con este análisis cambiar el método como se imparte la asignatura y/o la estrategia de tutoría inteligente.

Para esto se abre en varias pestañas del navegador con la vista global del modelo (tantas como clústeres haya en éste), y se selecciona en cada una de ellas un clúster diferente. Las operaciones más utilizadas para este análisis es la navegación por el autómatas o paneo, la búsqueda de estados específicos, el *zoom* y la presentación del detalle de estados.

Por ejemplo, en los datos obtenidos con los alumnos que utilizaron el laboratorio virtual de biotecnología, el MCCE obtuvo dos grupos utilizando la función de Eventos por Zona y el algoritmo XMeans (explicado en la sección 6.8).

Uno de los primeros pasos en esta comparación, puede ser visualizar lo más alejado posible la secuencia de estados de cada clúster, para así obtener una primera idea del tipo de alumnos de cada grupo (figura 7.15).

(a) Primera parte del autómata del clúster 0.

(b) Primera parte del autómata del clúster 1.

Figura 7.15 Visualización global del autómata de cada clúster.

En la figura 7.15 fácilmente se pueden visualizar las diferencias entre los autómatas, el del clúster 0 tiene muchos estados en la zona de errores relevantes, mientras que el del clúster 1 no. De lo que se puede concluir que los alumnos del clúster 1 tienen mejor rendimiento que los otros.

Ampliando la vista en el primer error del primer autómata, se puede apreciar que muy pocos estudiantes (casi un 8 %) olvidan tomar guantes de goma antes de obtener un vaso de precipitados (error `flt1_f0t8` en la figura 7.16). La frecuencia de este estado de

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

error es muy baja, además, este error no existe en el autómata del otro grupo, por estas razones, este error no es significativo para sacar conclusiones importantes.

Figura 7.16 Primer error en el autómata del clúster 0.

Por el contrario, en la primera secuencia de fallos del autómata del mismo clúster (figura 7.17), los estados de error poseen frecuencias de más del 60 %. Sin embargo, al no existir este tipo de errores en el otro grupo de estudiantes, se puede decir que, no necesariamente son errores asociados al método utilizado para impartir la asignatura. Para evitar este tipo de errores en futuros alumnos, se puede mejorar la tutoría automática para ese clúster en las acciones del proceso representadas por los estados ubicados cerca de los errores.

Continuando con la navegación por el autómata del clúster 1, el primer error encontrado es al finalizar la mezcla de la fase 2 (f2t52), en el cual la mitad de los alumnos olvidan agregar dos compuestos. Si se busca este estado en el otro clúster se puede ver, que además de estos mismos errores, existen otros igualmente por olvidar agregar compuestos a la mezcla (figura 7.18), todos con una frecuencia alta.

Para evitar este tipo de errores en un futuro, aparte con una mejor tutoría automática, el profesor podría modificar la metodología en la que dicta esos conceptos, para que los alumnos los asimilen de mejor manera.

Por último, sabiendo que la última acción es la f3t61, se puede buscar ésta en ambos autómatas. Los resultados, presentados en la figura 7.19, muestran que dentro del primer grupo de estudiantes ninguno finaliza correctamente la práctica de laboratorio, al contrario que en el clúster 1.

Figura 7.17 Primera secuencia de errores en el autómata del clúster 0.

7.5.3. Encontrar errores comunes

Una de las mayores utilidades del visualizador es para poder encontrar eventos más comunes que se produzcan en el proceso de entrenamiento, como errores, ya sean estos importantes o no pedagógicamente. Y, partiendo del estado deseado, analizar la secuencia de acciones que han realizado los estudiantes para llegar a éste.

Usando la vista global del modelo se pueden filtrar los estados con frecuencias bajas para que se presenten solamente los estados más comunes. Posteriormente, utilizando para eso la función de paneo, se puede navegar por el autómata para encontrar errores interesantes que hayan cometido los estudiantes. Finalmente, utilizando el *zoom* y/o

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

Figura 7.18 Comparación de errores al finalizar la mezcla de la fase 2 en los autómatas de los dos grupos de estudiantes.

seleccionando el estado se descubrirá los detalles del error, lo cual llevará al profesor a sacar conclusiones con respecto a si los estudiantes han asimilado o no los conceptos.

Siguiendo con el caso del laboratorio de biotecnología, un incidente de error es el presentado por la figura 7.20, el cual indica que un alto número de estudiantes comenten el error de volver a intentar ejecutar una acción (en este caso esterilizar el material), ya sea desde un estado correcto o después de haber cometido otro tipo de errores.

VISUALIZACIÓN DEL MODELO DE COMPORTAMIENTO COLECTIVO DE ESTUDIANTES

Figura 7.21 Error irrelevante frecuente importante para el profesor.

En el último ejemplo, presentado por la figura 7.22, se encontraron dos errores relevantes con una frecuencia mayor al 60 %, los cuales se producen desde la acción “apagar agitador” (f1t20). Uno se genera por no agregar bactoagar a la mezcla, mientras que el otro se da por no ajustar el pH de la misma. Analizando la secuencia de acciones se puede ver que efectivamente faltan estos pasos en la secuencia de estados correctos, ya que se han filtrado por tener frecuencias bajas, es decir, son ejecutados por los estudiantes que no han cometido los errores (menos del 40 %).

Figura 7.22 Errores relevantes frecuentes.

Este tipo de errores son importantes para el profesor, ya que así puede darse cuenta de los pasos del proceso que los alumnos asimilan de forma menos eficaz, y profundizar en los conceptos relacionados con estos.

Asimismo, se podría modificar la estrategia de tutoría automática para otorgar a los alumnos un mensaje que no enumere cuáles son los componentes que se deben agregar. Por ejemplo, pedirles que recuerden revisar en sus apuntes cuáles son los componentes de la mezcla. Este tipo de mensajes genéricos son necesarios, ya que al profesor le interesa

que los alumnos aprendan qué es lo que deben agregar por ellos mismos, no simplemente siguiendo paso a paso lo que indique el tutor, como si fuera un tutorial.

CAPÍTULO 8

ESTRATEGIA DE TUTORÍA

Además de la interpretación que da un experto humano a los patrones generados por el proceso de minería por medio de una representación gráfica de éstos, existe la posibilidad de que un sistema inteligente los analice y saque provecho de dichos patrones. En nuestro caso este sistema inteligente es un SIT, el cual intenta utilizar los resultados generados por el modelo de comportamiento colectivo de estudiantes para otorgar una tutoría automática adaptada a cada estudiante teniendo en cuenta el comportamiento demostrado por los estudiantes de su mismo clúster.

En algunos casos es una buena idea dejar que los estudiantes se equivoquen para mejorar su propio aprendizaje, es decir, que aprendan de sus errores. Sin embargo, en otros casos es mejor prevenir los errores dando el mínimo soporte necesario para que puedan progresar en la solución de un problema por sí mismos, y así superen su zona de desarrollo próximo (ZDP)(Vygotsky 1978).

Como ya se explicó en el apartado 2.1.6, el andamiaje (*scaffolding*) de Wood et al. (1976) puede ayudar al alumno a superar su ZDP. De esta manera, cada estudiante puede aprender con una pequeña ayuda y no necesariamente de sus errores. En términos de las dimensiones de tutoría contingente (instruccional, de dominio y temporal) (Wood 2003), la estrategia de tutoría propuesta abarcará la contingencia instruccional al establecer el nivel de detalle de los mensajes (aumento o desvanecimiento de la tutoría) según el grado de certeza de que el alumno vaya a ser capaz de realizar una acción correcta o cometer un error.

La contingencia de dominio es un concepto que está intrínseco en el entrenamiento procedimental, puesto que este tipo de adiestramiento se basa en una secuencia establecida de pasos, por lo cual, el tutor debe enfocarse en calcular la tutoría para el siguiente paso.

La contingencia temporal será considerada al decidir si hay que dar una pista al estudiante y en qué momento dependiendo de los resultados otorgados tanto por el

modelo individual del estudiante como por el de comportamiento colectivo de estudiantes, según se explica en la sección 8.3.

Además, al utilizarse el andamiaje como herramienta principal para la tutoría, deben estar presentes sus tres características dentro del proceso: contingencia, desvanecimiento (*fading*) y transferencia de responsabilidad Pol et al. (2010). Como ya se explicó en la sección 2.1.6, se denomina contingencia a la adaptación del soporte o ayuda al ZDP de cada estudiante. El desvanecimiento es la reducción gradual de esta ayuda debido a que el alumno está aprendiendo. Y la transferencia de responsabilidad se da gracias a que la asistencia va desvaneciéndose, el estudiante puede ir asumiendo un mayor rol dentro de su aprendizaje o, en otras palabras, éste puede responsabilizarse más de su tarea.

De esta manera, la estrategia de tutoría propuesta con cada error cometido por el alumno incrementará gradualmente el soporte (*scaffold*), o gradualmente disipará la ayuda si el estudiante muestra un progreso favorable (Olney 2014; Holden & Sinatra 2014). Por otra parte, la transferencia de responsabilidad no es una competencia a implementar en el tutor automático, ya que ésta no es un proceso computacional, sino el hecho de que el estudiante se responsabilice más de su aprendizaje como resultado del desvanecimiento de su tutoría.

8.1. Propuesta de Estrategia de Tutoría

La presente propuesta de estrategia de tutoría se basa en los conceptos antes mencionados, es decir, intenta ajustar la tutoría de acuerdo al comportamiento que tenga el estudiante dentro del entorno de entrenamiento. De esta forma, la figura 8.1 presenta un proceso iterativo que adapta la tutoría a cada interacción o falta de interacción del estudiante con el entorno de entrenamiento. En este proceso, se selecciona la tutoría atendiendo al modelo individual del estudiante y al modelo de comportamiento colectivo, los cuales han sido previamente actualizados con todos los eventos que ha generado el alumno hasta el momento.

Debido a los diferentes tipos de entornos de entrenamiento procedimental existentes y a la gran cantidad de tipos procesos (dominios de aprendizaje) que pueden abarcar cada uno de estos entornos, la estrategia de tutoría detallada en la figura 8.1 presenta un enfoque de tutoría genérica o independiente del dominio.

En la secuencia de la figura 8.1 existen dos tipos de pasos, los que son realizados por el estudiante (gris) y los que son procesados por el SIT (azul).

Figura 8.1 Estrategia de tutoría.

Como primer paso el estudiante selecciona el proceso de entrenamiento o práctica que desea realizar entre los que ofrece el entorno. De esta forma, se inicializa el proceso iterativo para cada evento que genere el estudiante en el entorno.

Antes de percibir la ejecución de una acción, el tutor inicializa un temporizador con el tiempo estimado que se necesita para ejecutar la siguiente acción correcta. Si este temporizador termina o si el estudiante interactúa con el entorno generando un evento, se actualizan tanto el modelo del estudiante (ME) como el modelo colectivo de comportamiento de estudiantes (MCCE) con dicho evento. La forma en la que se actualiza el MCCE con cada nuevo evento, se encuentra detallada en la sección 6.6.

Por otro lado, el ME se modifica mediante la inclusión de una nueva *Action_Trace* a la ontología *Student_Trace* con el evento que se ha producido. En el ME (Clemente et al. 2011; Clemente 2011) cada tipo de acción está definida mediante una subclase de la clase *Punctual_Action*, las cuales, a su vez, son tratadas también como objetos de aprendizaje o *learning object* –LO– (McGreal 2004; Polsani 2006).

Después de actualizar los modelos, se selecciona la realimentación de tutoría más adecuada para cada estudiante y para el instante en el que se encuentra éste dentro del proceso de entrenamiento (sección 8.3). Esta ayuda debe ser definida con arreglo a la ZDP del estudiante calculada por el ME y el MCCE (secciones 8.4.2 y 8.5.1). Posteriormente, esta ayuda es enviada al entorno de entrenamiento, para que éste la presente al estudiante.

Finalmente, se actualiza la ontología de trazas de estudiantes del ME con la tutoría escogida. Y se vuelve al inicio del proceso, es decir, se vuelve a inicializar el temporizador para la siguiente acción correcta que se debe ejecutar y se espera a que el estudiante interactúe con el entorno.

Cada iteración de esta estrategia de tutoría concuerda con el concepto de andamio o *scaffold* de la teoría propuesta por Wood et al. (1976), ya que, en cada iteración, el nivel de ayuda se va aumentando o disminuyendo dependiendo del desempeño del estudiante en el proceso de aprendizaje. Esto se consigue gracias a la continua actualización de los modelos (ME y MCCE), con lo cual se posee una visión completamente actualizada del estudiante para poder decidir qué tutoría es la más adecuada en cada momento.

En este sentido, mientras el alumno vaya realizando acciones correctas, los modelos ME y MCCE predecirán que sus siguientes acciones serán correctas con una alta probabilidad, por lo cual la tutoría irá desvaneciéndose.

Por el contrario, al cometer un error, los mismos modelos predecirán que el estudiante no posee unos ciertos conocimientos, por lo tanto la probabilidad de que el estudiante cometa un nuevo error será mayor y entonces el tutor le apoyará más para evitar futuros errores.

8.2. Selección de Tutoría para el Estudiante

La selección de una tutoría adecuada para el estudiante es muy importante para que éste, además de aprender, no se frustre en el proceso. Para concretar el proceso de tutoría, es importante explicar primero qué tipos de tutoría existen en la presente propuesta, así como los tipos y niveles de pistas que pueden dar cada uno de estos tipos de tutoría.

8.2.1. Tipos de tutoría

Para analizar los tipos de tutoría es importante recordar los componentes del módulo de tutoría dentro de la arquitectura del MCCE presentados en la figura 8.2 (tomada de la figura 6.1 de la sección 6.1).

Figura 8.2 Componentes del módulo de tutoría.

Como ya se explicó en la sección 6.1, el conmutador se encarga de decidir qué tipo de tutoría es la más apropiada para cada estudiante en cada momento. Esta inferencia se produce dentro de un proceso de selección de tutoría (sección 8.3), basándose en el estado de los conocimientos del estudiante (calculado por el ME), el resultado de la predicción realizada por el MCCE, y la definición de la acción que se encuentra en el módulo experto.

Además del conmutador, la figura 8.2 posee tres componentes más, los cuales generan la ayuda para cada tipo de tutoría:

- Tutoría basada en el modelo del estudiante.
- Tutoría basada en el modelo de comportamiento colectivo de estudiantes.
- Tutoría genérica.

La tutoría basada en el ME utiliza los datos del estado de los conocimientos del alumno para otorgar una ayuda adaptada a lo que no conoce, es decir, por medio de las inferencias de este módulo se puede discernir si el estudiante no conoce o ha olvidado un concepto asociado con una acción futura. El proceso para la obtención de este tipo de ayuda se detalla en la sección 8.4.

El segundo tipo de tutoría utiliza el MCCE para proporcionar una ayuda adaptada al tipo de estudiante (gracias al agrupamiento) teniendo en cuenta los futuros eventos

generados por el estudiante más probables (gracias al autómata). El proceso se detalla en la sección 8.5.

Como se explicará con detalle a continuación, una de las ventajas de la tutoría adaptada a las características individuales del estudiante (ME) es que ésta puede afinar la pista para cada objetivo de aprendizaje de cada acción del proceso de aprendizaje. En cambio, la tutoría apoyada por el MCCE es de grano más grueso, ya que proporciona pistas para facilitar o evitar la realización de acciones completas.

La tutoría genérica se refiere a un tipo de ayuda más rígida, que no está adaptada a cada estudiante. Es decir, se otorgan mensajes teniendo solo en cuenta la siguiente acción que se debe realizar (según esté configurada ésta en el módulo experto) o el último error cometido. Este tipo de tutoría se encuentra detallada en el apartado de tutoría del laboratorio de biotecnología desarrollado como preámbulo a esta tesis (Riofrío-Luzcando et al. 2012; Rico et al. 2012). Esta tutoría genérica se basa en lo que cree el profesor que es el mensaje más adecuado para ayudar a un estudiante cualquiera a ejecutar la siguiente acción.

8.2.2. Tipos y niveles de pistas

Como ya se ha explicado anteriormente, la pista que se otorgue al estudiante será adaptada a éste dependiendo de sus conocimientos y/o su comportamiento en el entorno. Para esto, es necesario que estas pistas estén configuradas en el módulo experto con varios niveles de detalle para así poder elegir una ayuda más o menos específica dependiendo de cuál sea la más adecuada en cada caso.

Figura 8.3 Actividades, acciones y objetivos de aprendizaje.

Como indica la figura 8.3, en la configuración presente en el módulo experto, el proceso de entrenamiento se encuentra conformado por actividades, las cuales se definen por medio

de un grupo de acciones o tareas con sus respectivas precondiciones y postcondiciones. Y, cada acción, a su vez, tiene asociados uno o más objetivos de aprendizaje (Clemente et al. 2011; Clemente 2011).

Las pistas deben estar vinculadas a cada una de estas categorías configuradas para el proceso de entrenamiento. De esta forma, sumando a éstas los errores que pueden producirse durante un proceso de entrenamiento, los tipos de pistas que se pueden otorgar al estudiante son:

- Pistas por actividad.
- Pistas por acción.
- Pistas por objetivo de aprendizaje.
- Pistas por error.

Así, se deben configurar pistas con los distintos niveles de detalle que desee el profesor para cada actividad, acción, objetivo y posible error. Además, cabe también la posibilidad de que el profesor no desee especificar ningún tipo de pista para una cierta actividad, acción u objetivo por razones pedagógicas.

Pistas por actividad

Si está configurada la tutoría para una actividad, ésta al menos debería ser del tipo genérica, es decir, un mensaje que, sin especificar claramente lo que se debe realizar en la acción, ayude al alumno a darse cuenta de lo que se debe realizar en ella.

Por ejemplo, si la actividad corresponde a la obtención de una mezcla de varios químicos, la pista genérica podría ser un mensaje como: “No olvides consultar en el protocolo cuáles son los componentes de la mezcla”.

Además de la pista genérica, podría existir para la actividad otro nivel de tutoría más detallado, como, por ejemplo, un vídeo o una demostración por medio de un avatar como indican W. L. Johnson y Rickel (1997) o Guerrero y Ramírez (2015).

Pistas por acción

De igual forma, para las acciones pueden existir o no pistas de tutoría configuradas, pero, a diferencia de las pistas de las actividades, si está configurada una tutoría, ésta debería tener varios niveles de detalle dependiendo del grado de probabilidad o certeza que se tenga de que el estudiante no sabe cómo realizar la tarea.

Otra diferencia con la tutoría para las actividades es que los vídeos o demostraciones se pueden realizar solamente para un conjunto de acciones. Por esta razón, las pistas para cada acción serán mensajes que guíen al alumno a realizar únicamente una acción.

Por ejemplo, en la mezcla anterior de químicos, la primera acción podría ser llenar un vaso con agua. Para la cual, los mensajes de tutoría según su nivel de detalle podrían ser:

- Mensaje genérico: “Consulta en el protocolo cuál es la primera acción necesaria en el proceso de la mezcla”.
- Pista de bajo detalle: “No olvides que la solución necesita estar hidratada”.
- Pista de alto detalle: “Tienes que llenar el vaso con agua”.

Pistas por objetivo de aprendizaje

De igual forma que para las acciones, las pistas por objetivo pueden tener varios niveles de detalle. Éstas asistirán al alumno en la ejecución de la tarea, pero con la vista puesta en un cierto objetivo de aprendizaje asociado a la tarea.

Por ejemplo, distinguir pistas por objetivos de aprendizaje es útil cuando el alumno saber cómo realizar una acción, pero no conoce la herramienta necesaria para realizar dicha acción. Es decir, cuando el alumno solo posee algunos de los conocimientos necesarios para realizar una acción. Para este caso no se dará tutoría para ayudar a saber por qué o cuándo se debe realizar la acción, sino para identificar la herramienta necesaria.

Por ejemplo, supongamos que mientras se realiza la mezcla del ejemplo anterior, hay que medir su pH para verificar si la solución no es ácida. Los objetivos de aprendizaje para esta tarea serían:

1. Saber que hay que medir el pH de la solución.
2. Saber en qué momento hay que medirlo.
3. Reconocer el pH-metro.
4. Saber dónde está el pH-metro.

Si suponemos que el alumno no sabe reconocer el pH-metro (objetivo 3), los mensajes de tutoría según su nivel de detalle podrían ser:

- Mensaje genérico: “El pH-metro es un sensor”.
- Pista de bajo detalle: “El pH-metro es un medidor de color blanco con una vara al costado”.
- Pista de alto detalle: Presentar una imagen con el pH-metro.

Pistas por error

Los errores que se pueden presentar en un entorno de entrenamiento procedimental pueden ser de varios tipos:

- Errores por dependencias fallidas.
- Errores por acciones incompatibles.
- Errores del entorno.
- Otro tipo de errores.

Los errores por dependencias e incompatibilidades están vinculados a la secuencia de las acciones. Un error por dependencia se puede producir cuando se desea realizar una acción sin haber ejecutado otra acción previa a esta, es decir, en la secuencia de acciones $A \rightarrow B \rightarrow C$ ejecutar B sin haber realizado antes A .

Los errores por incompatibilidades se producen cuando el estudiante ha ejecutado una acción que no debería haber sido realizada antes que otra acción. Por ejemplo, supongamos que consideramos la secuencia del ejemplo anterior y el alumno realiza una acción incorrecta D antes que C . En ese caso, cuando se valide la acción C , se detectará una incompatibilidad con D .

Para prevenir errores de estos dos tipos anteriores, se pueden emplear pistas con varios niveles de detalle. Por ejemplo, los mensajes para prevenir el error de olvidar agregar agua del ejemplo anterior de la mezcla de químicos podrían ser:

- Pista de bajo detalle: “No olvides que la solución necesita estar hidratada”.
- Pista de alto detalle: “Debes agregar agua al vaso”.

Es importante diferenciar un mensaje de “prevención de error” de un mensaje de error, pues los últimos se generan después de que se haya producido el error. Así, para el ejemplo anterior, un posible mensaje de error sería “No has agregado agua al vaso”.

Los errores del entorno son aquellos que se ocasionan cuando el estado o la disposición de los objetos en el entorno virtual no es la necesaria como para que se pueda realizar una acción. Por ejemplo, soltar un vaso en el suelo en lugar de en un fregadero o intentar tocar un objeto que se encuentra demasiado lejos.

En general, los errores de entorno suelen tener poca importancia pedagógica, ya que normalmente suelen estar asociados a problemas en la interacción con el entorno virtual. Por tanto, los mensajes para éstos serán de dos niveles, uno genérico para prevenir el

error después de una acción correcta, y uno detallado para prevenir que el alumno se estanque en el error, es decir, siga cometiéndolo una y otra vez. Por ejemplo, para el error de soltar el vaso podrían ser:

- Pista de bajo detalle: “El vaso debe ser llenado de agua en un sitio adecuado para ello”.
- Pista de alto detalle: “El vaso debe ser soltado bajo el grifo de agua ubicado junto a la pared y cerca de la puerta de la sala”.

Esto es así porque no importa que el alumno cometa una vez un error de entorno, pero debemos evitar que se quede bloqueado y siga insistiendo en el mismo error.

El último tipo de errores se refiere a los fallos que no son pedagógicamente relevantes, por ejemplo, si un estudiante ejecuta una acción que ya ha sido realizada anteriormente. Al ser estos errores poco importantes, no tendrán mensaje de prevención, sino solamente mensajes de error, los cuales podrían estar formulados de forma genérica, por ejemplo, “Acción ya realizada”.

Tutoría autorregulada

Además de las pistas de ayuda para acciones, objetivos o errores, se puede utilizar tutoría para el aprendizaje autorregulado (Zimmerman 1990).

Una estrategia de tutoría autorregulada permite al estudiante monitorizar su progreso en el proceso de entrenamiento mediante una bitácora de aprendizaje (Goldberg & Spain 2014). Para esto se presentan dos soluciones a continuación.

La primera solución consiste en presentar la lista de las acciones que ha ejecutado el estudiante en modo de texto cuando éste lo solicite. En esta lista no se diferenciarán entre acciones correctas e incorrectas. Es decir, simplemente se presentan los eventos generados por el alumno, por ejemplo:

1. Vaso tomado de la vitrina.
2. Error al soltar el vaso.
3. Vaso ubicado debajo del grifo de agua.
4. Vaso lleno con agua.

La segunda solución consiste en mostrar el grafo de la secuencia de eventos ejecutados. Para esto se utilizará la vista de estudiante del visualizador del MCCE (figura 7.10). El

momento en el cual se presentará este grafo dependerá de la configuración de la práctica en el módulo experto, es decir, se debe especificar después de qué acción se debe presentar el grafo creado hasta ese instante.

Además del grafo de los eventos ejecutados por el alumno, se presentará uno con la ruta correcta de acciones que debería haber realizado para que pueda comparar con su solución.

Otra estrategia para el aprendizaje autorregulado es la auto-explicación (Goldberg & Spain 2014), la cual promueve que los estudiantes expliquen con sus propias palabras lo que han aprendido. Una de las formas para realizar una auto-explicación dentro de un entorno de aprendizaje es por medio de un menú que permita al estudiante seleccionar la explicación que demuestra mejor lo que se ha aprendido (Aleven & Koedinger 2002).

Esto se puede solucionar agregando un nuevo agente a la arquitectura de tutoría inteligente, presentada en la sección 6.1.1. Este agente se encargará de generar las preguntas necesarias para el test que se debe presentar al estudiante. Este test no solo servirá para que el estudiante se dé cuenta de lo que ha aprendido y lo que no, sino también para actualizar el nivel de fiabilidad que tiene el modelo del estudiante sobre la adquisición o no de los objetivos de aprendizaje evaluados por el test.

8.3. Proceso de Selección de Tutoría

De los niveles de tutoría antes mencionados, es importante proporcionar siempre al estudiante la tutoría por actividad (siempre que esté configurada) y la tutoría para prevenir errores. Sin embargo, la tutoría por acción y por objetivo, serán proporcionadas dependiendo del modelo seleccionado por el conmutador, es decir, se otorgará la una o la otra dependiendo de los resultados que generen el ME y el MCCE. Preferentemente, se tratará de dar la tutoría por objetivo basada en el ME, porque, además de proporcionar una tutoría de grano más fino, se apoyará en información específica de cada estudiante.

La figura 8.4 define el proceso en el que se decide cuándo dar cada uno de estos tipos de tutorías y en qué orden entre ellos. Esta figura, además, indica en qué modelo estará basada la selección de la pista para el alumno.

Como se puede observar en la figura 8.4, el proceso de tutoría se dispara cuando el estudiante interactúa con el entorno de aprendizaje ($acción_i$). El proceso empieza entonces con una secuencia de pasos para dar tutoría autorregulada, los cuales se ejecutan si la $acción_i$ es el final de un subproceso o fase.

Posteriormente, se selecciona la pista más adecuada para la $acción_{i+1}$, como se encuentra detallado en el algoritmo 2. Este proceso también se ejecuta si el temporizador,

Figura 8.4 Proceso general de tutoría.

inicializado con el tiempo máximo de espera para ejecutar la $acción_{i+1}$ después de la $acción_i$, ha finalizado. Si es éste el caso, se pregunta al estudiante si desea ayuda para continuar con el proceso de entrenamiento. Si la respuesta es afirmativa, se ejecuta de igual forma el algoritmo 2 para la $acción_{i+1}$

En este algoritmo, las funciones auxiliares (TUTORÍAME_ACTIVIDAD, TUTORÍAME_ZDP, TUTORÍAMCCE_ZDP y TUTORÍAMCCE_ERRORES) están asociadas a cada tipo de tutoría especificada anteriormente. Por ejemplo, el método TUTORÍAME_ACTIVIDAD define el proceso de tutoría para una actividad basándose en la pre-

Algoritmo 2 Proceso de selección de tutoría

Entradas

Entrada $acción_i$
Entrada $estudiante$
Entrada ME
Entrada $MCCE$

Salidas

Salida $pistas$

```
if  $acción_{i+1}$  tiene tutoría para actividad configurada then
     $pistas = \text{TUTORÍAME\_ACTIVIDAD}(\text{estudiante}, acción_i, ME)$ 
end if
if  $acción_{i+1}$  tiene tutoría para acción configurada then
     $pistas = pistas + \text{SELECCIÓN TUTORÍA ZDP}(\text{estudiante}, acción_i, ME, MCCE)$ 
end if
 $pistas = pistas + \text{TUTORÍAMCCE\_ERRORES}(\text{estudiante}, acción_i, MCCE)$ 

function SELECCIÓN TUTORÍA ZDP( $estudiante, acción_i, ME, MCCE$ )
    for each  $acción_{i+1}$  posible desde  $acción_i$  do
        Obtener objetivos de aprendizaje de la  $acción_{i+1}$  del ME
         $fiabilidadME = ALTA$ 
        for each  $objetivo \in acción_{i+1}$  do
            if  $objetivo.priority = HIGH \wedge (objetivo.acquired = unknown \vee 0 <$  then
                 $objetivo.levelCurrentReliability < objetivo.levelReliability)$ 
                 $fiabilidadME = BAJA$ 
                break
            end if
        end for
        if  $fiabilidadME = BAJA$  then
            break
        end if
    end for
    if  $fiabilidadME = ALTA$  then
         $pistas = \text{TUTORÍAME\_ZDP}(\text{estudiante}, acción_i, ME)$ 
    else
        Obtener soporte de la  $acción_i$  del MCCE (ver ecuación 6.1)
        if  $soporte \geq acción_i.umbra$  then
             $pistas = \text{TUTORÍAMCCE\_ZDP}(\text{estudiante}, acción_i, MCCE)$ 
        else
             $pistas = \text{tutoría genérica como se explica en la sección 3.3.2}$ 
        end if
    end if
    return  $pistas$ 
end function
```

dicción dada por el modelo del estudiante (ME). Estos métodos devuelven una o varias pistas, las cuales se asignan a una colección de *pistas*, las cuales serán proporcionadas al estudiante posteriormente.

Con respecto a la selección del tipo de tutoría para la acción, se puede apreciar que se toma como prioritaria la predicción que pueda dar el ME. Sin embargo, si la fiabilidad de dicha predicción es baja se toma la predicción dada por el MCCE. Ahora bien, si la fiabilidad de ambas predicciones es baja, se escoge la tutoría genérica.

La fiabilidad de la predicción dada por el ME se calcula en base a la ecuación 8.1. Por otro lado, para que la fiabilidad de la predicción otorgada por el MCCE sea alta, el soporte del estado derivado de la última acción ejecutada, tiene que ser mayor a un umbral configurado para dicha acción. Este soporte se calcula como indica la ecuación 6.1 de la sección 6.3.

$$\begin{aligned} & \text{objetivo.priority} = HIGH \wedge \\ & (\text{objetivo.acquired} = unknown \vee \\ & 0 < \text{objetivo.levelCurrentReliability} < \text{objetivo.levelReliability}) \end{aligned} \quad (8.1)$$

Como se puede ver en la ecuación 8.1, la fiabilidad se calcula para un solo *objetivo* de aprendizaje, ya que se ejecuta (dentro del algoritmo 2) iterativamente por todos los objetivos de acción_{i+1} hasta encontrar uno que cumpla con la condición, si existe. Esto es debido a que si el ME no tiene suficiente certeza sobre la adquisición o no de al menos un objetivo de alta prioridad asociado a la acción, no se puede saber con certeza si el alumno sabrá o no realizar dicha acción.

A continuación, se van a explicar las clases y propiedades de la ontología definida en el ME que aparecen en la ecuación 8.1.

La propiedad *acquired*, asociada a la clase *Objective_State* de la ontología *Learning_Objective* del ME, puede tomar uno de los siguientes tres valores (Clemente 2011; Clemente et al. 2011): *true* (el sistema sabe que el estudiante alcanzó el objetivo), *false* (el sistema sabe que el estudiante no alcanzó el objetivo) y *unknown* (el sistema no sabe nada acerca de la adquisición del objetivo por el estudiante).

La propiedad *priority*, perteneciente a la misma clase *Objective_State* de la ontología *Learning_Objective* del ME, representa la importancia que tiene el objetivo de aprendizaje dentro de la acción: *HIGH* o *LOW*. Por ejemplo, supongamos que la acción “medir pH” (antes mencionada) tiene asociados los siguientes objetivos de aprendizaje:

1. Saber que hay que medir el pH de la solución.
2. Saber en qué momento hay que medirlo.

3. Reconocer el pH-metro.
4. Saber dónde está el pH-metro.

De estos objetivos, los tres primeros tienen una importancia pedagógica mayor que el último, puesto que el desconocimiento de dónde se encuentra un objeto no significa necesariamente que el alumno desconozca el propósito de la acción. Además, si el estudiante ha realizado la tarea en otro entorno, es muy probable que no sepa dónde se encuentra la herramienta en un nuevo entorno de aprendizaje, pero es posible que posea los conocimientos asociados a los otros objetivos.

Por esta razón, no se descarta la tutoría basada en el ME para la acción, si ésta tiene objetivos de baja prioridad, para los cuales el ME no tiene información, o si la tiene, no posee un nivel de certeza mayor al exigido para dichos objetivos.

En la clase *Specific_Objective* de la ontología *Learning_Objective*, la propiedad *levelReliability* representa el umbral de fiabilidad para considerar a un objetivo como alcanzado o no. Este umbral se refiere al número mínimo de veces que el estudiante debería haber demostrado (o no) haber alcanzado un objetivo como para que el objetivo se considera adquirido (o no). Por otro lado, en la clase *Specific_Objective_State* de la ontología *Student_State*, la propiedad *levelCurrentReliability* establece el número de veces que el estudiante ha demostrado haber alcanzado ese objetivo. Esta propiedad se infiere por medio de un conjunto de reglas de diagnóstico pedagógico (Clemente et al. 2011; Clemente 2011), que tienen en cuenta el comportamiento del estudiante así como otras características personales.

Las funciones auxiliares TUTORÍAME_ACTIVIDAD, TUTORÍAME_ZDP, TUTORÍAMCCE_ZDP y TUTORÍAMCCE_ERRORES, que se utilizan en el algoritmo 2, son detalladas en los siguientes apartados de este capítulo.

8.4. Tutoría basada en el Modelo del Estudiante

Como se ha explicado anteriormente, la tutoría basada en el ME intenta adaptar las pistas al conocimiento que tiene el alumno acerca del tema que se está tratando en el entrenamiento. Este tipo de tutoría se ejecuta para obtener pistas para una actividad o para los objetivos de la siguiente acción que debería ejecutar el estudiante.

8.4.1. Tutoría para actividad

Se ofrece al alumno ayuda sobre cómo realizar una actividad, si la siguiente acción que debería ejecutar el estudiante tiene configurado este nivel de tutoría, como indi-

ca la figura 8.4. El algoritmo 2, además de detallar esta condición, llama al método `TUTORÍAME_ACTIVIDAD` definido en el algoritmo 3.

Para poder seleccionar qué tipo de pista se otorga al alumno para enseñarle cómo realizar una actividad, es necesario calcular la fiabilidad de las inferencias del ME para esa actividad. Para esto, se calculan tres tipos de fiabilidades dependiendo de la confianza de cada objetivo y de si el alumno ha adquirido o no los objetivos (*acquired*). La confianza de un objetivo, que se define en la ecuación 8.2, es una medida de la certeza que se tiene sobre el estado de adquisición de un objetivo. Si la confianza es mayor o igual que uno, esto implica que se tiene una certeza lo suficientemente alta sobre el estado de adquisición del objetivo. En cambio, si la confianza está por debajo de uno, esto implicará lo contrario.

$$confianza_objetivo = \frac{objetivo.levelCurrentReliability}{objetivo.levelReliability} \quad (8.2)$$

$$fiabilidad_estricta_false_actividad = \frac{\sum_{j=1}^{acciones\ de\ la\ actividad_i} \frac{\sum_{objetivos\ de\ acción_j} \frac{objetivo.acquired=false \wedge confianza_objetivo \geq 1}{número_objetivos_acción_j} 1}{número_acciones_actividad_i} \quad (8.3)$$

$$fiabilidad_false_actividad = \frac{\sum_{j=1}^{acciones\ de\ la\ actividad_i} \frac{\sum_{objetivos\ de\ acción_j} \frac{objetivo.acquired=false \wedge confianza_objetivo}{número_objetivos_acción_j}}{número_acciones_actividad_i} \quad (8.4)$$

$$fiabilidad_true_actividad = \frac{\sum_{j=1}^{acciones\ de\ la\ actividad_i} \frac{\sum_{objetivos\ de\ acción_j} \frac{objetivo.acquired=true \wedge confianza_objetivo}{número_objetivos_acción_j}}{número_acciones_actividad_i} \quad (8.5)$$

Dos de estas fiabilidades se calculan para los objetivos no adquiridos (*acquired* = *false*). La primera (ecuación 8.3) calcula la proporción de objetivos no adquiridos con una confianza mayor o igual a 1, es decir, se excluyen aquellos objetivos de los cuales el ME no está completamente seguro de que no se hayan adquirido (*objetivo.levelCurrentReliability* $\geq$ *objetivo.levelReliability*). Por esta razón, se dice que esta fiabilidad es estricta.

La segunda fiabilidad para los objetivos no adquiridos (ecuación 8.4) define una fiabilidad que no es estricta, puesto que calcula el promedio de las confianzas de los objetivos que no han sido alcanzados por el alumno.

Finalmente, se define una fiabilidad para cuando el alumno sí que ha alcanzado el objetivo (ecuación 8.5). Ésta se calcula como el promedio de las confianzas de los objetivos con *acquired* = *true*.

Estas tres fiabilidades son utilizadas en el algoritmo 3 para definir las condiciones que permiten escoger qué tipo de pista es más adecuada dar al estudiante con respecto a la siguiente actividad.

Algoritmo 3 Proceso de tutoría para una actividad

Entradas

Entrada *acción_i*
Entrada *estudiante*
Entrada *ME*

Salidas

Salida *pista*

```
function TUTORÍAME_ACTIVIDAD(estudiante, accióni, ME)
  Obtener actividadi de accióni
  if actividadi tiene configurada demostración then
    if fiabilidad_estricta_false_actividad ≥ umbral_estricto then
      (ver ecuaciones 8.4 y 8.5)
      pista = demostración con avatar
    else if fiabilidad_false_actividad ≥ umbral_false ∨ fiabilidad_true_actividad ≤ umbral_true then
      (fiabilidad_false_actividad y fiabilidad_true_actividad definidas en
      ecuaciones 8.4 y 8.5)
      Preguntar al alumno si desea una demostración
      if respuesta afirmativa then
        pista = demostración con avatar
      else
        pista = mensaje configurado para actividadi
      end if
    else
      pista = mensaje configurado para actividadi
      (Condición que se cumple si para todos los objetivos se cumple
      acquired = unknown o fiabilidad_true_actividad > umbral_true o
      fiabilidad_false_actividad < umbral_false)
    end if
  else
    pista = mensaje configurado para actividadi
  end if
  return pista
end function
```

El procedimiento TUTORÍAME_ACTIVIDAD recibe como parámetro el *estudiante*, además de la acción, ya que las ecuaciones 8.2, 8.3, 8.4 y 8.5, se calculan en base a los objetivos alcanzados y no alcanzados por este *estudiante*. Es decir, se obtiene el *levelCurrentReliability* de cada objetivo del *estudiante* en el ME que tenga la propiedad *acquired* en *false* y *true*.

Los umbrales utilizados en el algoritmo 3 (*umbral_estricto*, *umbral_false* y *umbral_true*) son constantes definidas por el profesor para cada actividad. Es decir, se deben definir los valores mínimos para los cuales las fiabilidades pueden darse por válidas.

Por ejemplo, el *umbral_estricto* para la *actividad_i* podría tener un valor alto (0.90) para asegurarnos de que se va a proporcionar una demostración con avatar al alumno si existe una certeza alta de que el alumno desconoce todos los objetivos de la actividad. Por otro lado, el *umbral_false* podría tener un valor menor (0.75), puesto que la demostración siempre se la dará, si el estudiante así lo desea.

8.4.2. Tutoría para la zona de desarrollo próximo del estudiante

Para otorgar pistas adaptadas a las necesidades individuales de cada estudiante, es importante primero definir su ZDP con respecto a las propiedades de su conocimiento definidas en el ME. Una vez determinada la ZDP de cada estudiante, se puede decidir qué nivel de detalle de pista se le presenta dependiendo del conocimiento adquirido sobre el dominio que abarca el proceso de entrenamiento.

Figura 8.5 Definición del cálculo del ZDP para el ME.

De esta forma, la ZDP, calculada en términos del ME, está definida por los objetivos de aprendizaje de cada acción consecutiva (*acción_{i+1}*) a la última ejecutada

($acción_i$). Estos objetivos son aquellos de los cuales el ME no ha inferido si el estudiante los conoce o no ($acquired = unknown$); de los que no tiene la suficiente información ($levelCurrentReliability < levelReliability$); o de los que el ME tiene una certeza alta que el estudiante no conoce ($acquired = false \wedge levelCurrentReliability \geq levelReliability$). Esta definición está representada por la figura 8.5.

El proceso que decide qué nivel de pista se otorga dependiendo de las características de cada objetivo de aprendizaje, es detallado por el algoritmo 4.

Algoritmo 4 Proceso de tutoría para el ZDP de un estudiante usando el ME

Entradas

Entrada $acción_i$
 Entrada $estudiante$
 Entrada ME

Salidas

Salida $pistas$

```

function TUTORÍAME_ZDP(estudiante,  $acción_i$ ,  $ME$ )
  for each  $acción_{i+1}$  posible desde  $acción_i$  do
    for each  $objetivo \in acción_{i+1}$  do
       $confianza\_objetivo = \frac{ME.estudiante.objetivo.levelCurrentReliability}{ME.objetivo.levelReliability}$ 
      if  $objetivo.acquired = unknown$  then
         $pistas = pistas +$  mensaje de ayuda de bajo detalle
      else if  $objetivo.acquired = true \wedge confianza\_objetivo < 1$  then
         $pistas = pistas +$  mensaje de ayuda genérico
      else if  $objetivo.acquired = false$  then
        if  $confianza\_objetivo < 1$  then
           $pistas = pistas +$  mensaje de ayuda de bajo detalle
        else
           $pistas = pistas +$  mensaje de ayuda de alto detalle
          (Condición para  $confianza\_objetivo \geq 1$ )
        end if
      end if
    end for
  end for
  return  $pistas$ 
end function

```

Como se puede apreciar en el algoritmo 4, cuando el objetivo ha sido adquirido pero se tiene una certeza baja, se otorga una pista genérica. Esta baja certeza significa que el ME no está seguro de que el alumno conozca el objetivo de aprendizaje.

Para los objetivos de los que no se tiene información, aquellos de los cuales no se ha inferido nada, se otorga una pista con bajo detalle, pero mayor que una pista genérica

(como en el caso anterior). Esto es debido a que es muy probable que el alumno no conozca el objetivo, ya que no lo ha demostrado hasta el momento.

En el caso de los objetivos que no han sido adquiridos, existen dos posibilidades: con fiabilidad alta y baja. Para los primeros, el ME tiene una alta certeza de que el alumno no los conoce, por ello se otorga una pista de alto detalle. Mientras que para los objetivos de fiabilidad baja ocurre una situación similar a la de los objetivos de los que no se dispone información, debido a que es probable que el alumno no los conozca, pero el ME no está seguro de ello. Por tanto, se proporciona una pista con bajo detalle.

8.5. Tutoría basada en el Modelo de Comportamiento Colectivo de Estudiantes

Como se ha mencionado ya, el MCCE permite predecir el siguiente evento más probable que puede generar un alumno, este evento puede ser una acción válida o un error. Para el caso de que el siguiente evento más probable sea una acción válida, se realiza una tutoría para ayudar al alumno a superar su ZDP. En cambio, para un evento de error, la tutoría adecuada es aquella en la que se intenta prevenir que el alumno cometa dicho fallo.

Estos tipos de tutoría fueron ya explicados anteriormente, así como también se ha detallado el orden en el que se ejecutan dentro del proceso de selección de tutoría (sección 8.3).

8.5.1. Tutoría para la zona de desarrollo próximo del estudiante

De igual forma que en la tutoría basada en el ME, la contingencia se abarca por medio del cálculo de la ZDP del estudiante. Por esta razón, el siguiente apartado describe el método en el que se calcula esta zona basado en el autómata definido para el MCCE. Posteriormente se detalla el proceso para seleccionar el nivel de pista adecuado para la confianza de predicción de esa ZDP.

Cálculo de la zona de desarrollo próximo

En base a los conceptos tratados anteriormente, la ZDP del estudiante en proceso de entrenamiento se calcula utilizando el autómata generado por la secuencia estados correctos que representan las acciones definidas en el módulo experto, como muestra la figura 8.6.

Figura 8.6 Definición del cálculo del ZDP para el MCCE.

La ZDP de un estudiante está representada por los estados correctos que puede alcanzar éste con un poco de ayuda del tutor automático desde el último estado. Se entiende por un “poco de ayuda” un mensaje o pista que proporcione la suficiente información al estudiante como para que sea capaz por sí mismo de realizar la siguiente acción correcta. Ahora bien, como lo que se pretende es ajustar el nivel de ayuda de manera que el estudiante reciba la ayuda justa y necesaria, se tendrá en cuenta la confianza de la transición que une el estado actual con el siguiente estado correcto. En este sentido, si esta confianza es alta, la pista será genérica o de poco de detalle, o incluso se podría prescindir de la pista. En cambio, si la confianza es baja, la pista tendrá que ser más detallada, porque eso indicará que pocos alumnos en este clúster han sido capaces de realizar la acción correctamente.

La actualización dinámica del MCCE mientras los alumnos de una cohorte realizan la práctica irá ajustado el nivel de detalle de las pistas según las confianzas vayan variando durante dicha práctica. Esto lógicamente será posible, siempre y cuando se hayan configurado pistas con distinto nivel de detalle para cada transición de interés pedagógico.

Llegados a este punto, cabe señalar que la frontera entre la ZDP y la siguiente zona es difusa. Por tanto, tendría sentido suponer que otros estados a los que no se pueda llegar directamente desde el estado actual, podrían estar también en la ZDP. Esto es así, porque también podrían ser alcanzados con un poco de ayuda tras dos pasos o N pasos. No obstante, dado que ya existe un tipo de tutoría para una actividad o subproceso completo, la tutoría basada en el MCCE que se va a determinar en cada paso, se va a definir con la vista puesta únicamente en la siguiente acción del procedimiento.

Suponiendo que existe una secuencia de acciones $A \rightarrow B \rightarrow C$, y que un estudiante ha ejecutado ya la acción A , la ZDP de dicho estudiante sería la acción B (figura 8.6), puesto que para ejecutar esta acción, podría necesitar asistencia al momento de haber realizado la acción A . De hecho, con una confianza del 30 %, será razonable dar una pista detallada, ya que parece que la mayoría de los estudiantes que alcanzaron el estado

A, no fueron capaces de llegar al B. Por otro lado, para realizar la acción C el alumno necesitará un poco más ayuda, sobre todo, teniendo en cuenta que solo el 12% (40×30) de los que llegaron al estado A, luego consiguieron llegar al C.

Por ejemplo, tomando la parte final del autómata para el proceso de entrenamiento en el mantenimiento de bujías en un motor de combustión interna (sección 6.4.1), se puede calcular la ZDP desde la acción 7 como muestra la figura 8.7.

Figura 8.7 Ejemplo de ZDP para autómata de mantenimiento de bujías.

Como se puede ver en la figura, la ZDP desde el estado 7 está representada por el estado 8.2, puesto que la acción hacer 8.2 es la única acción correcta posible que el alumno puede realizar inmediatamente después de hacer 7. Cuando este haya realizado dicha acción, la nueva ZDP será el estado 11.

Si se da el caso de que el alumno cometa un error, la ZDP estará determinado por las siguientes acciones correctas posibles, a pesar que de los siguientes estados arrastren el error anterior. Por ejemplo, si se ejecutara la acción hacer 8.1 en lugar de hacer 8.2 después de la acción hacer 7, la ZDP seguirá siendo el estado 11, de igual forma que en el ejemplo anterior.

Otro ejemplo de cálculo de la ZDP es el expuesto por la figura 8.8, tomado de la parte inicial del autómata del laboratorio de biotecnología (sección 6.4.2).

Figura 8.8 Ejemplo de ZDP para autómata del laboratorio de biotecnología.

En este caso, la ZDP no está formada solamente por un estado, sino por dos (3 y 4). Esto es debido a que desde el estado 2 es posible ejecutar dos acciones distintas, ya que pueden ser ejecutadas en diferente orden relativo entre ellas ($3 \rightarrow 4$ o $4 \rightarrow 3$). Sin embargo, una vez que haya sido ejecutada una de las dos, la siguiente ZDP estará

formado por el estado derivado de la otra; es decir, el estado 4 formaría la ZDP, si se ha ejecutado hacer 3 después de hacer 2; y la ZDP sería igual al estado 3, si se ha ejecutado hacer 4 luego de hacer 2.

Proceso de selección de pista para la zona de desarrollo próximo

Como se detalla en la sección anterior, la ZDP engloba todas las posibles acciones correctas que se pueden realizar desde la última ejecutada, por lo tanto, la contingencia de la tutoría basada en el MCCE se puede definir a partir de la probabilidad de realizar una acción correcta en el siguiente paso del entrenamiento.

Desde la última acción se obtiene la confianza de los eventos (transiciones) que producen estados presentes en la ZDP del estudiante. Si al menos una de estas confianzas está por encima de un umbral configurado por el profesor, no se otorga una pista al estudiante, pues es muy probable que ejecute este evento por sí solo. Es decir, no necesita ayuda para superar su ZDP.

Sin embargo, si no existe un evento cuya confianza supere este umbral, se puede generar una pista para ayudar al estudiante a superar su ZDP. Esta pista se genera en base a la máxima de las confianzas de los eventos desde la última acción, y el nivel de pista se obtiene de acuerdo a los umbrales configurados en el módulo experto para la acción relacionada con el evento con máxima confianza.

Este proceso se detalla en el algoritmo 5. Algoritmo que detalla el método TUTORÍAMCCE_ZDP llamado desde el proceso de selección de tutoría (algoritmo 2).

Algoritmo 5 Proceso de tutoría para el ZDP de un estudiante usando el MCCE

Entradas

Entrada $acción_i$
Entrada $estudiante$
Entrada $MCCE$

Salidas

Salida $pista$

function TUTORÍAMCCE_ZDP($estudiante$, $acción_i$, $MCCE$)

Obtener del $MCCE$ el $clúster$ al que pertenece $estudiante$

Obtener el $autómata$ del $clúster$

Obtener $autómata.s_i | s_i$ concuerda con $acción_i$

if $\nexists t \in T_N | \delta(t) = (s_i, s_{i+1}) \wedge t.confianza > umbral_no_tutoria$ **then**

(ver ecuaciones 6.2, 6.3 y 6.5 del capítulo 6)

$max_confianza = \max(t.confianza | \delta(t) = (s_i, s_{i+1}))$

```
if  $max\_confianza \geq umbral_A$  then
  pista = pista genérica
else if  $umbral_A > max\_confianza \geq umbral_B$  then
  pista = pista de bajo detalle
else if  $max\_confianza < umbral_B$  then
  pista = pista de alto detalle
end if
end if
return pista
end function
```

El *umbral_no_tutoria* es un valor constante definido por el profesor para cada proceso de entrenamiento. Así como también, son especificados los umbrales para escoger el nivel de detalle de la pista para cada acción. Estos umbrales se configuran en el módulo experto en forma de rango de valores numéricos ($umbral_A = 0.75$ y $umbral_B = 0.5$). Por ejemplo:

- $\geq 0.75 \rightarrow$ Pista genérica.
- $< 0.75 \wedge \geq 0.5 \rightarrow$ Pista de bajo detalle.
- $< 0.5 \rightarrow$ Pista de alto detalle.

8.5.2. Tutoría para la prevención de errores

La prevención de errores es una de las aplicaciones principales del MCCE. Sin embargo, no es recomendable prevenir siempre a los alumnos si pueden cometer un error, por lo tanto, sólo se les ayudará si la probabilidad de que caigan en el fallo es alta (como indica el algoritmo 6).

Esta probabilidad se calcula no solo para los errores que son inminentes, sino también para los errores más probables que pueden ser ejecutados unos pasos más tarde.

Sea la secuencia de estados ($\bar{S}$) desde s_1 a s_N definida por la ecuación 8.6, donde el último estado al que ha llegado el estudiante s_i es igual al estado inicial de la ruta (s_1) y el estado de error s_{er} es igual al estado final (s_N).

$$\begin{aligned}\bar{S} &= (s_1, s_2, \dots, s_N) \\ \forall k \in \{1, 2, \dots, N\}, \exists t \in T | \delta(t) &= (s_k, s_{k+1}) \\ s_i &= s_1 \wedge s_{er} = s_N\end{aligned}\tag{8.6}$$

La probabilidad de caer en un error inminente se calcula simplemente con la confianza de la transición que lleva al estado que corresponde al error (ecuación 6.2). Ahora bien, cuando se consideran también los errores que se pueden producir tras un cierto número de pasos (secuencia $\bar{S}$), la confianza de un estado de error s_{er} se calcula mediante el producto de las confianzas de las transiciones que llegan a este desde el estado actual s_i (ecuación 8.7). De esta forma, en general, conforme más alejado se encuentre el estado de error S_{er} desde el último estado alcanzado por el alumno S_i , menor será esta confianza.

$$confianza_{\bar{S}} = \prod_{k=1}^{N-1} t_k.confianza | \delta(t_k) = (s_k, s_{k+1}) \wedge s_k = \bar{S}_k \quad (8.7)$$

Además de prevenir los errores pedagógicamente importantes o relevantes, el algoritmo 6 provee pistas al estudiante para que no cometa o no vuelva a cometer (estancamiento) un error irrelevante. La decisión de cuándo proporcionar y cuándo no la pista a un estudiante que ha caído en un error de este tipo, se basa en el cálculo del número de estudiantes que en anteriores sesiones de entrenamiento han caído en el error la misma cantidad de veces que lo ha hecho el alumno que está realizando la práctica, dividido por el soporte del estado de error (ecuación 8.8). Si hay pocos alumnos respecto al total de los que cometieron al menos una vez el error (soporte del estado de error) que consiguen salir en el siguiente intento, se proporciona una pista detallada. Si, por el contrario, hay muchos, entonces la pista no se considera necesaria.

$$\frac{\phi_V(t)_{[n-1]}}{\gamma(s_i)} \leq umbral_repetir_error \text{ donde } n \rightarrow \begin{matrix} \text{número de veces consecuti-} \\ \text{vas que el } estudiante \text{ ha} \\ \text{cometido el } error \end{matrix} \quad (8.8)$$

Algoritmo 6 Proceso de tutoría para la prevención de errores usando el MCCE

Entradas

Entrada $acción_i$
Entrada $estudiante$
Entrada $MCCE$

Salidas

Salida $pistas$

function TUTORÍAMCCE_ERRORES($estudiante$, $acción_i$, $MCCE$)

Obtener del $MCCE$ el $clúster$ al que pertenece $estudiante$

Obtener el $autómata$ del $clúster$

Obtener $autómata.s_i | s_i$ concuerda con $acción_i$

▷ **Prevención de errores relevantes**

```
for each  $s_{er} \in S_{RE} | s_{er}.soporte \geq umbral\_errores\_relevantes \wedge$  do  
  el error de  $s_{er}$  tiene tutoría configurada  $\wedge \exists \bar{S} | \bar{S} = (s_i, \dots, s_{er})$   
  (ver ecuaciones 6.3 y 6.5)  
  Calcular  $confianza\_S$  según ecuación 8.7  
  if  $confianza\_S \geq umbral_A$  then  
     $pistas = pistas +$  pista genérica  
  else if  $umbral_A > confianza\_S \geq umbral_B$  then  
     $pistas = pistas +$  pista de bajo detalle  
  else if  $confianza\_S < umbral_B$  then  
     $pistas = pistas +$  pista de alto detalle  
  end if  
end for
```

▷ **Prevención de errores irrelevantes**

```
for each  $t_I \in T_N | \exists s_{i+1} \in S_{IE} \wedge \exists \delta(t_I) = (s_i, s_{i+1})$  do  
  if  $t_I.confianza > umbral\_errores\_irrelevantes \wedge$  then  
     $\exists$  mensaje genérico para el error de  $s_{i+1}$   
     $pistas = pistas +$  mensaje genérico de error  
  end if  
end for  
if  $s_i \in S_{IE} \wedge \exists$  mensaje detallado para el error de  $s_i$  then  
  Obtener  $t_V \in T_V | \delta(t_V) = (s_i, s_{i+1}) \wedge s_{i+1} \in S_C$   
  Obtener  $n$  = número de veces consecutivas que el estudiante  
    ha cometido el error  
  if  $\frac{\phi_V(t)_{[n-1]}}{\gamma(s_i)} \leq umbral\_repetir\_error$  then (ver ecuación 6.8)  
     $pistas = pistas +$  mensaje detallado para error  
  end if  
end if  
return  $pistas$   
end function
```

La función $\phi_V(t)$ de la ecuación 8.8 está definida en la ecuación 6.8 del capítulo 6.

Los umbrales que se utilizan en el algoritmo para decidir si se da o no la pista son constantes definidas por el profesor. Los umbrales *umbral_errores_relevantes* y *umbral_errores_irrelevantes* tienen que ser valores relativamente grandes, puesto que la tutoría se da si la probabilidad de caer en el error es alta. Por otra parte, el *umbral_repetir_error* es un valor relativamente pequeño debido a que la pista se da al estudiante para que evite el estancamiento, cuando la probabilidad de que salga en el próximo intento es baja.

Los valores de estos umbrales y todos los demás necesarios para los procesos detallados en este capítulo pueden ser el resultado de un análisis de las soluciones adoptadas por

los alumnos durante el proceso de entrenamiento y de los errores que han cometido. Por tanto, este sería otro ejemplo de aplicación del visualizador del MCCE, detallado en el capítulo 7.

Parte III

**RESULTADOS Y
APORTACIONES**

CAPÍTULO 9

VALIDACIÓN

La validación que se presenta en este capítulo tiene tres objetivos:

1. Verificar la precisión del modelo de comportamiento colectivo de estudiantes (MCCE) a la hora de predecir los eventos generados por los alumnos.
2. Validar la clasificación de los estudiantes realizada por el MCCE mediante las funciones de agrupamiento.
3. Validar la utilidad del visualizador del MCCE a la hora de mejorar la tutoría que proporciona un tutor automático.

Con el primer objetivo de la validación se pretende demostrar que el error promedio asociado a las predicciones del modelo es aceptable para un tutor automático. Este error indica el grado de acierto a la hora de predecir el siguiente evento que generará un estudiante. Esto es, si el modelo predice que existe una probabilidad de 1 de que el estudiante realice una acción y el estudiante ejecuta esa acción, el error será 0. De lo contrario, si el estudiante no la realiza, el error será 1 (sección 9.1.1). Para calcular el error promedio del modelo se dividió el conjunto de todos los *logs* en un conjunto de entrenamiento y un conjunto de pruebas, y se creó un modelo a partir del conjunto de entrenamiento. A continuación, se utilizó este modelo para predecir los eventos del conjunto de pruebas y se calculó el error promedio obtenido al realizar estas predicciones.

Con el segundo objetivo se desea constatar si los métodos de agrupamiento pueden clasificar a los estudiantes en clústeres o grupos, de forma que cada grupo requiera una tutoría distinta. Para ello, se compararon los grupos creados con diferentes algoritmos de agrupamiento, comprobando cuántos estudiantes en cada uno de ellos cometieron errores relevantes desde el punto de vista pedagógico.

Finalmente, para alcanzar el tercer objetivo, se utilizó el visualizador del MCCE para encontrar un conjunto de errores típicos. A continuación, se modificaron ciertos elementos del laboratorio virtual así como la estrategia de tutoría asociada a algunos de los errores típicos identificados previamente. Posteriormente se contabilizó el número de errores que cometieron los nuevos estudiantes en el laboratorio modificado.

Es importante que una propuesta de investigación sea validada con datos reales según su campo de aplicación. Los datos utilizados para validar el MCCE y su visualizador fueron tomados de los *logs* de actividad de alumnos registrados por el tutor automático del laboratorio virtual de biotecnología, detallado en la sección 3.3 (Riofrío-Luzcando et al. 2012; Rico et al. 2012).

En el presente trabajo hemos utilizado básicamente dos tipos de métodos para crear los clústeres de estudiantes. El primero se basa en las características o parámetros encontrados en el proceso de entrenamiento y el segundo en la secuencia de acciones realizada por cada estudiante. En este sentido, el MCCE se ha validado utilizando las funciones de agrupamiento detalladas en el apartado 6.8, además del método secuencial:

- Agrupamiento por errores.
- Agrupamiento por errores y tiempo.
- Agrupamiento por eventos en cada zona del autómata.

Para cada una de estas funciones, se contrastaron los resultados obtenidos con dos métodos de clusterización: *XMeans* y *Expectation Maximization (EM)*. Se escogieron estos dos métodos porque crean los grupos sin necesidad de establecer a priori el número de clústeres que se desean obtener. Las implementaciones utilizadas para estos algoritmos son las que se encuentran disponibles en el popular *framework* para minería de datos llamado Weka (Hall et al. 2009).

Para el método de agrupamiento en base a la secuencia de acciones realizada por cada estudiante, se utilizó el algoritmo creado por Microsoft para este cometido, llamado *Microsoft Sequence Clustering* (Microsoft 2016). Éste es un algoritmo híbrido que combina técnicas de minería de datos con cadenas de Markov para identificar el número de grupos y sus secuencias. Para esto, examina todas las probabilidades de transición entre estados y mide las distancias entre todas las secuencias posibles en el conjunto de datos para determinar qué secuencias son las mejores para ser usadas como entradas para el agrupamiento. Posteriormente, utiliza este resultado como entrada de una implementación del algoritmo *Expectation Maximization* mediante el cual se generan los grupos de secuencias.

Hemos descartado la implementación del algoritmo *Sequential Information Bottleneck* (sIB) (Slonim, Friedman, & Tishby 2002), que se encuentra en Weka, debido a que esté requiere como entrada el número (k) de clústeres se desean obtener.

9.1. Validación de la Precisión de las Predicciones del Modelo de Comportamiento Colectivo de Estudiantes

Para esta validación se utilizaron los registros de actividad recogidos por el tutor automático del laboratorio virtual de biotecnología en los años 2013, 2014 y 2015 (85 estudiantes en total). Estos alumnos tomaron el curso de Bioquímica y Biotecnología, el cual forma parte del pnsum de estudio de la carrera de Ingeniera Forestal de la Escuela Tcnica Superior de Ingenieros de Montes de la Universidad Politcnica de Madrid (Rico, Ramrez, Riofro-Luzcando, & Berrocal-Lobo 2017; Ramrez, Rico, Riofro-Luzcando, Berrocal-Lobo, & de Antonio 2017).

El autmata creado sin clusterizar los *logs* de los 85 estudiantes (14943 registros de eventos) consta de 532 estados y 2778 transiciones (1454 normales y 1324 vectoriales). El nmero de estados est detallado en la tabla 9.1 para cada zona del autmata.

Tabla 9.1 Estados por zona del MCCE de validacin sin clusterizar

Zona	N�mero de estados
Flujo correcto	115
Errores irrelevantes	228
Errores relevantes	189

Como se ha indicado ya, la meta de esta validacin es calcular el error promedio que se obtiene al predecir, mediante el MCCE, el siguiente evento que puede generar un estudiante en cada paso de una prctica. En el siguiente apartado se explica el procedimiento que se aplic para calcular dicho error promedio.

9.1.1. Clculo del error promedio del modelo de comportamiento colectivo de estudiantes

Para calcular la media del error del modelo, se realiza el siguiente procedimiento:

1. Seleccionar aleatoriamente el 90 % de estudiantes.

2. Construir el modelo mediante el proceso definido en la sección 6.5 (algoritmo 1) utilizando los *logs* del 90 % de estudiantes anteriormente seleccionados.
3. Comprobar en qué medida el modelo creado predice los registros del 10 % restante de estudiantes.
4. Repetir desde el paso 1 al 3 un número n de veces para garantizar así la representatividad de la selección aleatoria de estudiantes.

El 90 % de los estudiantes del paso 1 representa el **conjunto de entrenamiento** y el 10 % restante, utilizado en el paso 3, representa el **conjunto de pruebas** para la validación del modelo. Este conjunto es empleado para estimar cuán preciso en sus predicciones puede ser el modelo en un entrenamiento real, de forma similar a la que utilizan muchos modelos de minería de datos o estadística.

El valor asignado para el número n de veces que se repiten los pasos del 1 al 3 fue de 150. A este valor se llegó después de verificar que los resultados y las gráficas obtenidas para $n = 75, 100$ y 150 , eran similares, es decir, que la tendencia del error de predicción era la misma.

El proceso para validar el alineamiento (paso 3) del conjunto de pruebas con respecto al modelo es definido por el algoritmo 7. Para este proceso, las entradas son el MCCE, el conjunto de pruebas de estudiantes, y la confianza y soporte que determinan el submodelo a ser validado. Por ejemplo, si $\text{soporte} = \text{confianza} = 0$, entonces será validado el modelo entero; mientras que si $\text{soporte} > S \wedge \text{confianza} > C$, solo será validado el submodelo formado por los estados con soporte mayor que S y por las transiciones con confianza mayor que C .

Los valores de soporte y confianza utilizados como entrada del algoritmo de validación son 0, 0.1, 0.25, 0.5, 0.75 y 0.9, debido a que estos valores han sido considerados lo suficientemente representativos y completos para ilustrar la evolución del error.

Los estados temporales en el algoritmo 7 son estados que no se encuentran en el modelo, pero tienen que ser creados temporalmente como parte de la validación. Es decir, si un estudiante del conjunto de prueba genera un evento, pero el MCCE no contiene ningún estado que represente el resultado de dicho evento, el algoritmo crea un nuevo estado que se vincula al estado anterior, y se calcula el error con respecto a este nuevo estado. Este estado es considerado temporal, ya que no se agrega al MCCE. Esto quiere decir que solo persiste, mientras se calcula el error para el *log* del estudiante que se está procesando. Esta operación se repite hasta que el siguiente evento en el registro de actividades del estudiante da lugar a un estado ya existente en el MCCE.

Algoritmo 7 Proceso de validación

Entradas

Entrada $MCCE$
Entrada $soporte$
Entrada $confianza$
Entrada $conj_prueba$

Salidas

Salida $\bar{E}$

```

for each  $log\_estudiante \in conj\_prueba$  do
  Obtener  $cl\acute{u}ster \in MCCE|log\_estudiante$  se ajusta mejor en  $cl\acute{u}ster$ 
  Obtener el  $aut\acute{o}mata \in cl\acute{u}ster$ 
  Obtener  $s_0 \in aut\acute{o}mata$ 
   $estado\_anterior = s_0$ 
  for each  $evento \in log\_estudiante$  do
    if  $\exists estado \in aut\acute{o}mata|estado$  es el resultado de  $evento$  then
       $existe(t) \equiv \exists t \in aut\acute{o}mata|\delta(t) = (estado\_anterior, estado)$ 
      if  $existe(t)$  then
        if  $estado_{sop} \geq soporte \wedge t_{conf} \geq confianza$  then
          Calcular  $E_{evento}$  según la ecuación 9.1, 9.2 or 9.3
        else
           $E_{evento} = 0$ 
        end if
      else
         $E_{evento} = 1$ 
      end if
    else
       $E_{evento} = 1$ 
       $estado =$  nuevo estado temporal desde  $evento$ 
    end if
     $estado\_anterior = estado$ 
  end for
  Actualizar  $\bar{E}$  con  $E_{evento}$  según la ecuación 9.4
end for

```

El error asociado a la predicción de un evento realizado por un alumno se formula como la alineación del evento con el MCCE. Ésta se calcula mediante una ecuación similar al *Replay Fitness* definido por Buijs, van Dongen, van der Aalst, y P (2014), el cual es empleado para cuantificar hasta qué punto el modelo puede reproducir las trazas registradas en un *log* en *process discovery*. La ecuación 9.1 define este cálculo para transiciones con frecuencias normales ($\in T_N$), donde t es la transición que coincide con

el evento ejecutado por el estudiante y s es el estado inicial de t (las funciones ϕ_N y γ están definidas en las ecuaciones 6.7 y 6.6 de la sección 6.3.4).

$$E_{evento} = 1 - \frac{\phi_N(t)}{\gamma(s)} \quad (9.1)$$

Ahora bien, para calcular este error en el caso de las transiciones vectoriales ($\in T_V$) será necesario distinguir dos casos. Si el estudiante sale del bucle en la primera iteración, se utiliza una ecuación similar a la anterior para calcular el error, donde la frecuencia de la transición es reemplazada por la frecuencia almacenada en el primer elemento del vector (la función ϕ_V está definida en la ecuación 6.8).

$$E_{evento} = 1 - \frac{\phi_V(t)[1]}{\gamma(s)} \quad (9.2)$$

En cambio, si el estudiante no sale del bucle en la primera iteración, la frecuencia de la transición es reemplazada por la suma de todos los elementos del vector de frecuencias, exceptuando el primero, es decir:

$$E_{evento} = 1 - \frac{\sum_{i=2}^{\phi_V(t)count} \phi_V(t)[i]}{\gamma(s)}. \quad (9.3)$$

La razón para realizar la distinción entre estos dos casos al momento de calcular el error de una transición vectorial es que (como se explica en la sección 8.5.2) al momento de ofrecer tutoría al estudiante, es mejor disponer de predicciones diferentes dependiendo de si éste cae en el bucle del error una o más de una vez. Por ejemplo, si es más probable que un alumno cometa un fallo solamente una vez, será menos necesario darle una ayuda para que evite repetir el mismo fallo.

Para calcular la media del error del evento, cada error E_i calculado por las ecuaciones antes mencionadas, es multiplicado por el número de estudiantes en el conjunto de prueba que han generado el mismo evento n_i . La sumatoria de estos productos es dividida por la sumatoria del número de estudiantes que han generado cada evento (ecuación 9.4).

$$\bar{E} = \frac{\sum_{i=1}^{no\ ev} E_i n_i}{\sum_{i=1}^{no\ ev} n_i} \quad (9.4)$$

Escrita de otra forma:

$$\bar{E} = \frac{E_1 n_1 + E_2 n_2 + \dots + E_{no\ ev} n_{no\ ev}}{n_1 + n_2 + \dots + n_{no\ ev}}$$

9.1.2. Resultados

A continuación, se van a detallar los errores promedios obtenidos con cada método de clusterización, con el fin de estudiar qué método ofrece mejores resultados. Para ello, se van a considerar los promedios de los errores calculados para los distintos modelos que se van a construir para cada combinación de método de clusterización y función de agrupamiento.

Error de predicción por función de agrupamiento

Para cada una de las funciones de agrupamiento, se presentará una tabla con la media del error calculada para cada uno de los métodos de clusterización (XMeans y EM) y por cada combinación de soporte y confianza. En estas tablas se destacan los valores inferiores, lo que ayuda en la decisión de escoger el mejor método. Estos valores están resaltados en cada tabla por una gama de colores definidos en la tabla 9.2.

Tabla 9.2 Gama de colores para valores inferiores de medias de error de predicción.

Valores inferiores con $sop. = 0$		Valores inferiores con $sop. = 0.5$
Valores inferiores con $sop. = 0.1$		Valores inferiores con $sop. = 0.75$
Valores inferiores con $sop. = 0.25$		Valores inferiores con $sop. = 0.9$

Agrupamiento por errores

Tabla 9.3 Media del error por método para agrupamiento por errores.

Método	Nº Clústeres	Sop. Conf.	0	0.1	0.25	0.5	0.75	0.9
XMeans	2	0	0.399	0.362	0.350	0.311	0.240	0.218
		0.1	0.489	0.418	0.391	0.309	0.159	0.144
		0.25	0.469	0.393	0.366	0.280	0.123	0.077
		0.5	0.418	0.326	0.317	0.273	0.120	0.067
		0.75	0.373	0.213	0.207	0.194	0.120	0.067
		0.9	0.455	0.122	0.110	0.095	0.086	0.067
EM	2	0	0.394	0.358	0.346	0.308	0.238	0.217
		0.1	0.483	0.415	0.388	0.312	0.162	0.131
		0.25	0.463	0.389	0.366	0.285	0.126	0.069
		0.5	0.416	0.330	0.323	0.281	0.124	0.062
		0.75	0.361	0.213	0.208	0.198	0.124	0.062
		0.9	0.435	0.117	0.109	0.097	0.088	0.062

Agrupamiento por errores y tiempo

Tabla 9.4 Media del error por método para agrupamiento por errores y tiempo.

Método	Nº Clústeres	Sop. Conf.	0	0.1	0.25	0.5	0.75	0.9
XMeans	2	0	0.400	0.363	0.350	0.312	0.240	0.218
		0.1	0.493	0.423	0.394	0.312	0.159	0.145
		0.25	0.473	0.397	0.370	0.285	0.123	0.080
		0.5	0.422	0.331	0.323	0.277	0.120	0.070
		0.75	0.372	0.212	0.207	0.194	0.120	0.070
		0.9	0.452	0.117	0.106	0.092	0.083	0.070
EM	1	0	0.409	0.366	0.353	0.315	0.247	0.202
		0.1	0.521	0.432	0.397	0.313	0.146	0.108
		0.25	0.504	0.401	0.357	0.280	0.109	0.047
		0.5	0.444	0.313	0.300	0.272	0.109	0.047
		0.75	0.426	0.199	0.188	0.184	0.109	0.047
		0.9	0.569	0.111	0.089	0.082	0.075	0.047

Agrupamiento por eventos en cada zona del autómata

Tabla 9.5 Media del error por método para agrupamiento por eventos por zona.

Método	Nº Clústeres	Sop. Conf.	0	0.1	0.25	0.5	0.75	0.9
XMeans	2	0	0.393	0.354	0.340	0.303	0.236	0.210
		0.1	0.485	0.414	0.383	0.302	0.154	0.140
		0.25	0.465	0.389	0.358	0.274	0.119	0.059
		0.5	0.414	0.321	0.309	0.269	0.119	0.057
		0.75	0.367	0.207	0.202	0.190	0.119	0.057
		0.9	0.444	0.112	0.104	0.092	0.085	0.057
EM	3	0	0.380	0.341	0.318	0.281	0.220	0.186
		0.1	0.528	0.443	0.384	0.294	0.176	0.130
		0.25	0.511	0.425	0.371	0.277	0.152	0.101
		0.5	0.463	0.365	0.324	0.272	0.145	0.092
		0.75	0.401	0.236	0.218	0.197	0.145	0.091
		0.9	0.483	0.175	0.140	0.107	0.095	0.091

El método que mejor predice los eventos de un alumno, según nuestros datos, por cada función de agrupamiento, se definió teniendo en cuenta los siguientes criterios:

- Mayor número de valores inferiores (según cada tabla).
- Número de clústeres debe ser mayor que 1.

De esta forma los métodos seleccionados por cada función son los presentados por la tabla 9.6:

Tabla 9.6 Métodos seleccionados por función de agrupamiento.

Función	Método
Errores	Expectation Maximization (EM)
Errores y tiempo	XMeans
Eventos por zona	XMeans

Las tablas de las medias de error para las funciones de agrupamiento por error y tiempo y por eventos por zona presentan claras diferencias, por lo cual no es difícil decidir qué método ha sido el mejor para cada una de esas funciones. Sin embargo, para la primera, ha sido necesario descartar a EM pues solo ha devuelto un clúster.

Para el caso del agrupamiento por errores, en la tabla 9.3 los valores de las medias de error, para casi todos los valores de soporte y confianza, son muy cercanos, incluso, el número de valores mínimos entre XMeans y EM son casi los mismos. No obstante, se ha escogido a EM como el mejor método para esta función de agrupamiento por tener mayor número de valores mínimos.

Comparación de medias de error por función de agrupamiento

Teniendo en cuenta la selección de métodos antes mencionada, la tabla 9.7 engloba la comparativa de las medias de error de las funciones antes vistas, incluyendo el agrupamiento por secuencias y el error producido por el autómata sin dividir por grupos.

En esta tabla se puede apreciar que las diferencias entre las funciones son muy pequeñas (alrededor de 0.01), cuando el soporte y confianza son iguales a 0.5. Además, para la mayoría de soportes cuando la *confianza* ≥ 0.75 , así como también para confianzas cuando el *soporte* ≥ 0.75 , la función sin clusterizar se convierte en la mejor, ya que tiene valores de error menores que las otras funciones.

Como se explicó en capítulos anteriores, el propósito de este modelo es el de proveer tutoría automática para las transiciones más probables (sección 8.5.1). Conceptualmente, esto es equivalente a trabajar con un sub-modelo con confianzas altas (por ejemplo

Tabla 9.7 Media de error por método y función de agrupamiento.

Función	Método	Nº Clústeres	Sop. Conf.	0	0.1	0.25	0.5	0.75	0.9
Errores	EM	2	0	0.394	0.358	0.346	0.308	0.238	0.217
			0.1	0.483	0.415	0.388	0.312	0.162	0.131
			0.25	0.463	0.389	0.366	0.285	0.126	0.069
			0.5	0.416	0.330	0.323	0.281	0.124	0.062
			0.75	0.361	0.213	0.208	0.198	0.124	0.062
			0.9	0.435	0.117	0.109	0.097	0.088	0.062
Errores y Tiempo	XMeans	2	0	0.400	0.363	0.350	0.312	0.240	0.218
			0.1	0.493	0.423	0.394	0.312	0.159	0.145
			0.25	0.473	0.397	0.370	0.285	0.123	0.080
			0.5	0.422	0.331	0.323	0.277	0.120	0.070
			0.75	0.372	0.212	0.207	0.194	0.120	0.070
			0.9	0.452	0.117	0.106	0.092	0.083	0.070
Eventos por Zona	XMeans	2	0	0.393	0.354	0.340	0.303	0.236	0.210
			0.1	0.485	0.414	0.383	0.302	0.154	0.140
			0.25	0.465	0.389	0.358	0.274	0.119	0.059
			0.5	0.414	0.321	0.309	0.269	0.119	0.057
			0.75	0.367	0.207	0.202	0.190	0.119	0.057
			0.9	0.444	0.112	0.104	0.092	0.085	0.057
Secuencias	Microsoft Sequence Clustering	6	0	0.358	0.323	0.307	0.264	0.189	0.153
			0.1	0.445	0.383	0.361	0.290	0.146	0.123
			0.25	0.427	0.365	0.349	0.280	0.136	0.105
			0.5	0.377	0.306	0.300	0.274	0.120	0.053
			0.75	0.324	0.201	0.197	0.192	0.120	0.053
			0.9	0.341	0.100	0.093	0.084	0.066	0.053
No clústeres			0	0.412	0.369	0.356	0.315	0.247	0.202
			0.1	0.522	0.433	0.400	0.312	0.146	0.107
			0.25	0.504	0.400	0.356	0.277	0.105	0.042
			0.5	0.444	0.312	0.298	0.269	0.105	0.042
			0.75	0.427	0.197	0.186	0.182	0.105	0.042
			0.9	0.574	0.105	0.085	0.077	0.070	0.042

confianza = 0.5) y *soporte* = 0 para no prescindir de ningún estado. Para visualizar la tendencia de las medias de error para un caso como este, se presenta la figura 9.1.

9.1.3. Discusión

Como se esperaba, la tabla 9.7 muestra que la tendencia descendente de los errores cuando la confianza y el soporte crecen. Esto puede deberse a que las transiciones más frecuentes y los estados más visitados reflejan un comportamiento más común entre los

Figura 9.1 Media de error por método y función de agrupamiento cuando $soporte = 0$.

estudiantes en el entorno de entrenamiento. Por lo tanto, las predicciones basadas en tales transiciones y estados son más confiables.

Creemos que el mínimo error cosechado para $soporte = 0$ y $confianza = 0$ (0.358 para el método de agrupamiento por secuencias) es razonablemente bueno para un sistema de entrenamiento como el laboratorio de biotecnología (del que se tomaron los datos). Es decir, es un buen resultado tomando en cuenta que en este tipo de entornos los estudiantes tienen mucha libertad de acción.

Además, tenemos que tomar en cuenta que el sistema inteligente de tutoría considerará las transiciones más probables de ser ejecutadas por futuros estudiantes (por ejemplo, aquellas con confianzas mayores a 0.5), en especial aquellas que dirigen a estados de error, para generar la ayuda necesaria para cada alumno. Por lo tanto, los errores de predicción para confianzas altas serán incluso menores (menores que 0.27 según nuestros datos) que los obtenidos con $confianza = soporte = 0$.

Asimismo, como se explicó en la sección 6.6, el MCCE será actualizado con cada nueva acción de los estudiantes que se encuentren utilizando el entorno de entrenamiento. Esto ayudará a reducir los errores de predicción debido a que el modelo tenderá a ajustar su estructura de mejor manera a los estudiantes que se encuentran en ese momento bajo supervisión.

Otra conclusión que se puede obtener de los datos presentados en la tabla 9.7, es que las diferencias entre las funciones de agrupamiento son pequeñas. Por ejemplo, con *soporte* = 0 y *confianza* = 0, es decir, el modelo entero sin filtrar por soporte ni confianza, el menor error es 0.358 (en el agrupamiento por secuencias) y el máximo error es 0.412 (en el método sin clusterizar). A pesar de que estas diferencias sean tan pequeñas, el mejor método de clusterización claramente es el de secuencias, principalmente para *soporte* = 0, como muestra la figura 9.1. Esta conclusión apoya la propuesta de separar a los alumnos por grupos según su comportamiento a través del proceso de entrenamiento.

9.2. Validación de la Clasificación de los Alumnos

En esta sección de la validación se pretende comprobar cuán bien clasifican a los estudiantes las funciones de agrupamiento (objetivo 2 de la validación del MCCE), de acuerdo a su comportamiento dentro del entorno de entrenamiento. Para esto, se han calculado las frecuencias de los errores pedagógicamente importantes más comunes (del proceso del laboratorio de biotecnología) que han cometido los alumnos presentes en los clústeres creados con las funciones de agrupamiento antes explicadas.

Para este propósito, se seleccionaron varios estados de error localizados en las zonas de errores relevantes de todos los autómatas, y posteriormente, para cada función de agrupamiento se compararon las frecuencias de estos errores en cada clúster creado con dicha función.

En este análisis se presenta una tabla para cada función de agrupamiento, en las cuales se resaltan las frecuencias de los errores sobre el 50 %. La primera columna de cada una de las tablas representa el código del error, la última columna incluye las varianzas de las frecuencias de error entre clústeres, y las columnas intermedias contienen las frecuencias de los estados de error para los autómatas generados para cada clúster.

9.2.1. Resultados

De igual forma, las tablas presentadas a continuación resaltan las frecuencias en base a una gama de colores definida por la tabla 9.8.

Tabla 9.8 Gama de colores para valores de frecuencias de cada error por clúster.

Frecuencias bajo 50 %		Frecuencias entre 60 % and 70 %
Frecuencias entre 50 % y 60 %		Frecuencias sobre 70 %

Agrupamiento por errores

Tabla 9.9 Frecuencias por clúster para el agrupamiento por errores (EM)

Error	Clúster 1	Clúster 2	Varianza
1.20_1.16	0.68	1	0.051
1.20_1.14	0.68	0.667	0.0001
2.52_2.40	0.64	0.567	0.003
2.52_2.38	0.72	0.433	0.041
2.52_2.37	0.52	0.517	0.00001
2.52_2.39	0.72	0.417	0.046
2.52_2.35	0.68	0.417	0.035
3.52_3.34	0.64	0.417	0.025
2.52_2.41	0.52	0.45	0.002
2.52_2.34	0.56	0.35	0.022
Promedio	0.636	0.523	0.023

Agrupamiento por errores y tiempo

Tabla 9.10 Frecuencias por clúster para el agrupamiento por errores y tiempo (XMeans)

Error	Clúster 1	Clúster 2	Varianza
1.20_1.16	0.726	1	0.038
1.20_1.14	0.677	0.652	0.0003
2.52_2.40	0.565	0.652	0.004
2.52_2.38	0.452	0.696	0.030
2.52_2.37	0.516	0.522	0.00002
2.52_2.39	0.435	0.696	0.034
2.52_2.35	0.435	0.652	0.023
3.52_3.34	0.419	0.652	0.027
2.52_2.41	0.452	0.522	0.002
2.52_2.34	0.371	0.522	0.011
Promedio	0.505	0.657	0.017

Agrupamiento por eventos en cada zona del autómata

Tabla 9.11 Frecuencias por clúster para el agrupamiento por eventos por zona (XMeans)

Error	Clúster 1	Clúster 2	Varianza
1.20_1.16	0.545	1	0.103
1.20_1.14	0.364	0.716	0.062
2.52_2.40	0.182	0.649	0.109
2.52_2.38	0.091	0.581	0.120
2.52_2.37	0.091	0.581	0.120
2.52_2.39	0.091	0.568	0.114
2.52_2.35	0.091	0.554	0.107
3.52_3.34	0.182	0.527	0.059
2.52_2.41	0.091	0.527	0.095
2.52_2.34	0.091	0.459	0.068
Promedio	0.18	0.616	0.096

Agrupamiento por secuencias

Tabla 9.12 Frecuencias por clúster para el agrupamiento por secuencias

Error	Cl. 1	Cl. 2	Cl. 3	Cl. 4	Cl. 5	Cl. 6	Var.
1.20_1.16	0.688	0.833	0.857	0.6	0.5	0.778	0.020
1.20_1.14	0.750	0.667	0.5	0.7	0.750	0.556	0.011
2.52_2.40	0.563	0.750	0.571	0.3	1	0.444	0.060
2.52_2.38	0.469	0.583	0.571	0.5	0.625	0.444	0.005
2.52_2.37	0.531	0.583	0.429	0.5	0.750	0.333	0.020
2.52_2.39	0.5	0.583	0.5	0.3	0.750	0.444	0.022
2.52_2.35	0.469	0.583	0.5	0.4	0.625	0.444	0.007
3.52_3.34	0.531	0.417	0.429	0.6	0.375	0.444	0.007
2.52_2.41	0.375	0.583	0.5	0.4	0.750	0.444	0.020
2.52_2.34	0.344	0.583	0.5	0.3	0.625	0.222	0.027
Promedio	0.522	0.617	0.536	0.460	0.675	0.456	0.020

9.2.2. Discusión

Las tablas anteriores muestran que todas las funciones de agrupamiento excepto la de secuencias, pueden agrupar satisfactoriamente a los alumnos en grupos diferentes, de

tal forma que no necesiten el mismo tipo de tutoría, ya que claramente en un clúster hay más frecuencias de errores que en el otro.

Esto se debe a que el agrupamiento por secuencias no agrupa teniendo en cuenta una métrica relacionada con el rendimiento de cada alumno, por ejemplo, los errores que se han cometido, sino la similitud de las secuencias. Por esta razón, es razonable que las otras funciones, las cuales, sí toman en cuenta métricas del rendimiento del alumno, dividan a los estudiantes de mejor manera.

Sin embargo, de entre estas funciones (errores, errores y tiempo y eventos por zona), la que mejor agrupa a los estudiantes (según los datos recolectados por el laboratorio de biotecnología) es la de eventos por zona. Eso se puede apreciar en la tabla 9.11, en la que el clúster con mejor rendimiento global solamente tiene un error con una frecuencia sobre el 50 %, mientras que en las tablas de las otras funciones el mejor clúster contiene varias frecuencias por encima del 50

Si se toma como ejemplo el error de la segunda fila de la tabla 9.11 (1.20_1.14), éste en el clúster 1 tiene una frecuencia baja del 36.4 %, al contrario que en el clúster 2 donde la frecuencia es alta (71.6 %). Esto implica que pocos estudiantes del clúster 1 deberán necesitar ayuda para evitar ese error debido a que muy pocos alumnos en el pasado lo han cometido. Al contrario, la mayoría de los estudiantes que coincidan en el clúster 2, deberán necesitar ayuda para prevenir caer en ese error, como se explica en la sección 8.5.

Esta diferencia entre los grupos, que convierte en la mejor a la función de eventos por zonas, se ratifica con la alta varianza de las frecuencias entre los clústeres para esta función. Como se puede apreciar, también en la tabla 9.11, el promedio de las varianzas de las frecuencias en los estados de error es un valor mucho mayor (0.096) al de las tablas de las otras funciones (0.023, 0.017 y 0.020).

Tomando en cuenta los resultados de este análisis y el anterior, la mejor función para agrupar a los estudiantes (según los datos utilizados para dicho estudio) es la de eventos por zona. Esto es debido no solamente a que es la mejor agrupando a los estudiantes por su rendimiento en el entorno, gracias al estudio de comparación de frecuencias por clúster, sino también a que es la segunda función con menor error de predicción (después de la de secuencia), especialmente cuando el *soporte* = 0 y la *confianza* = 0.

9.3. Validación de la Utilidad del Visualizador

Mediante esta validación se intenta comprobar que, si se cambian ciertos contenidos del entorno virtual de entrenamiento y los mensajes de tutoría relacionados con determinadas

VALIDACIÓN

zonas críticas del proceso, los nuevos estudiantes cometerán menos errores que los anteriores.

Para esto primero se identificaron las zonas críticas del proceso, es decir, aquellas en las que los alumnos de años anteriores cometieron más errores. Para esto se utilizó el visualizador del MCCE (definido en el capítulo 7), mediante el cual se encontraron los errores más comunes, como se puede ver en la sección , utilizando para ello los registros de actividad recolectados por el tutor automático del laboratorio virtual de biotecnología, en los años 2013, 2014 y 2015. Los errores encontrados gracias al visualizador se describen en la tabla 9.13, donde la primera columna describe el error encontrado; la segunda a la acción en la cual al validar su ejecución el tutor ha generado los eventos de error; la tercera indica el código de la acción fallida; y la última columna presenta el número del cambio que se ha realizado en la tutoría o en el entorno virtual, cambios que están detallados en las tablas 9.14, 9.15 y 9.16.

Tabla 9.13 Errores comunes encontrados gracias al visualizador

Error	Acción	Fallo	Cambio
No ajustan el pH de la mezcla	f1t20	f1t14	1
No agregan uno o más de los componentes del medio de cultivo y/o no se han añadido en el orden correcto	f1t20	f1t12, f1t13, f1t16	2
No encuentran fácilmente el autoclave	f1t25	faileddropbotella	3 y 13
No agregan uno o más de los componentes de la mezcla de PCR y/o no se han añadido en el orden correcto	f2t52	f2t34, f2t35, f2t36, f2t37, f2t38, f2t39, f2t40, f2t41	4
No agregan uno o más de los componentes de la mezcla de ligación y/o no se han añadido en el orden correcto	f3t52	f3t33, f3t34, f3t35, f3t36, f3t37	5
No agregan Agar al medio de cultivo	f1t20	f1t16	7 y 12
No agregan Tampón de Polimerasa a la mezcla de PCR	f2t52	f2t37	8
No agregan Enzima Taq Polimerasa a la mezcla de PCR.	f2t52	f2t40	9
No agregan Tampón Ligasa a la mezcla de ligación.	f3t52	f3t35	10
No agregan Producto de PCR a la mezcla de ligación.	f3t52	f3t34	11

Una vez encontradas estas acciones, se cambiaron sus mensajes de tutoría o se agregaron ayudas dentro del entorno de entrenamiento, por ejemplo, poner el nombre del instrumental sobre aquellos que los estudiantes no encontraban fácilmente. Los mensajes

de tutoría que se cambiaron, así como los que se agregaron a acciones que no tenían, se encuentran detallados en la tabla 9.14.

Tabla 9.14 Cambios en mensajes de tutoría

No.	Mensaje Anterior	Nuevo Mensaje
1	No existía un mensaje que recuerde al alumno de ajustar el pH	Acuérdate de ajustar el pH antes de añadir el agar
2	Toca el agitador para encenderlo.	Toca el agitador para encenderlo. Revisa la composición del medio en el PROTOCOLO
3	No existía un mensaje que ayude al alumno a encontrar el autoclave	Busca el AUTOCLAVE en otra sala, éste es como una olla a presión con el tamaño de una lavadora
4	No existía un mensaje que recuerde al alumno de revisar qué componentes forman la mezcla para PCR	Revisa la composición de la mezcla de PCR en el PROTOCOLO
5	No existía un mensaje que recuerde al alumno de revisar qué componentes forman la mezcla para la ligación	Revisa la composición de la mezcla de ligación en el PROTOCOLO
6	Solo se presentaba la lista de errores relevantes	se añade también: CONCLUSIÓN: PARA EVITAR ERRORES REVISAR MÁS EL PROTOCOLO

Además de estos mensajes de tutoría, se realizaron otros cambios en algunos nombres de químicos, para que los estudiantes los identifiquen más fácilmente. Estos cambios se presentan en la tabla 9.15.

Tabla 9.15 Cambios en nombres de químicos

No.	Mensaje Anterior	Nuevo Mensaje
7	Bactoagar	Agar
8	Tampón de Polimerasa	Tampón de la Enzima de Polimerasa
9	Enzima Taq Polimerasa	ADN Polimerasa
10	Tampón Ligasa	Tampón de la Enzima Ligasa (clonasa)
11	Producto de PCR	ADN amplificado por PCR

También se realizaron otros cambios dentro del laboratorio virtual, detallados en la tabla 9.16.

Posteriormente, alumnos que cursaron en el año 2016 la misma asignatura que los antes mencionados, utilizaron el laboratorio virtual. De los cuales se recolectaron sus registros de acciones, de la misma forma que en los años anteriores.

Tabla 9.16 Otros cambios en el laboratorio virtual

No.	Desc. Cambio
12	Se quitó el químico Agarosa.
13	Se agregó el nombre de Autoclave sobre el objeto 3D que lo representa

Los datos, analizados a continuación, han sido divididos en los dos períodos antes mencionados, es decir:

Período 1 Alumnos de los años 2013, 2014 y 2015 (pre cambio de tutoría).

Período 2 Alumnos del 2016 (post cambio de tutoría).

9.3.1. Resultados

Análisis de calificaciones Antes de analizar los resultados obtenidos sobre las frecuencias de errores obtenidas para los estudiantes que utilizaron el laboratorio en el año 2016, es importante verificar estadísticamente que tanto estos alumnos, como los de los años anteriores son muestras que pertenecen a un mismo universo.

Para esto se compararon las calificaciones finales en la asignatura de Bioquímica y Biotecnología (para la cual se utilizó el laboratorio virtual), de los estudiantes del año 2016 con los de los años anteriores (2013, 2014 y 2015).

En un primer paso, se comprobó si los dos conjuntos de datos cumplan la prueba de normalidad, para saber qué tipo test se puede ejecutar posteriormente. Las hipótesis para la comprobación de la normalidad de estos conjuntos de datos son:

H_0 Los datos de calificaciones de los estudiantes siguen una distribución normal.

H_1 Los datos de calificaciones de los estudiantes no siguen una distribución normal.

En los histogramas de los dos conjuntos de datos (figuras 9.2a y 9.3a), así como también en los gráficos de normalidad (figuras 9.2b y 9.3b), se puede apreciar que los datos no cumplen una normal.

Además, al ejecutar el test de Kolmogorov-Smirnova para las calificaciones de los alumnos del *período 1*, se ha obtenido un nivel de significancia de 0.0000007. Se ha seleccionado este test, ya que son 77 alumnos.

Con respecto a las calificaciones del *período 2*, al ser solamente 21 estudiantes, se ha ejecutado el test de Shapiro-Wilk, el cual otorgó un nivel de significancia de 0.00149.

Por lo tanto, para los dos conjuntos de datos se tiene que rechazar la H_0 , es decir, los datos de las calificaciones para los dos períodos no siguen una distribución normal.

Figura 9.2 Pruebas de normalidad de las calificaciones de alumnos entre 2013 y 2015 (*período 1*).

Figura 9.3 Pruebas de normalidad de las calificaciones de alumnos del 2016 (*período 2*).

Finalmente, se compararon las distribuciones de las calificaciones entre estos dos períodos utilizando una prueba no paramétrica, debido a que no siguen una distribución normal. Las hipótesis para este análisis son las siguientes:

H_0 Analizando las calificaciones, los estudiantes del *período 1* y del *período 2* proceden de la misma población.

H_1 Analizando las calificaciones, los estudiantes del *período 1* y del *período 2* no proceden de la misma población.

La prueba utilizada fue la U de Mann-Whitney mediante la cual se obtuvo un nivel de significancia de 0.142. Por lo cual se mantiene la H_0 , es decir, la distribución de las calificaciones es la misma para los dos períodos. Gracias a esto, se pueden comparar las frecuencias de los errores cometidos por los estudiantes en estos dos periodos.

Análisis de frecuencias de errores Para este análisis, se calcularon las diferencias entre las frecuencias de los estados de error producidos por los estudiantes de los años 2013, 2014 y 2015, y los cometidos por los alumnos del 2016. Además, se diferenciaron los eventos de error que han pueden haberse visto afectados por un cambio de los que no.

De igual forma que en el análisis de las calificaciones de los alumnos, el primer paso es verificar que el conjunto de datos sigue una distribución normal. Para esto, se plantearon las siguientes hipótesis:

H_0 Las diferencias de frecuencias de error de los estudiantes de los dos períodos siguen una distribución normal.

H_1 Las diferencias de frecuencias de error de los estudiantes de los dos períodos no siguen una distribución normal.

En el histograma (figura 9.4a) y el gráfico de normalidad (figura 9.4b) se puede apreciar que estas diferencias entre las frecuencias de error sí cumplen las características de una distribución normal. Esto se ratifica con la prueba de Kolmogorov-Smirnova (ya que se trata de 120 errores), la cual proporcionó un nivel de significancia de 0.2, con lo cual, se conserva la H_0 .

Gracias a que los datos siguen una distribución normal, se puede ejecutar una prueba T para comparar las distribuciones de las diferencias de las frecuencias de los errores asociados a algún cambio con las diferencias de las frecuencias de los errores que no están asociados a ningún cambio. Para esto, se plantearon las siguientes hipótesis:

H_0 Las variaciones en las frecuencias de error no dependen de un cambio en el entorno de entrenamiento.

H_1 Las variaciones en las frecuencias de error sí dependen de un cambio en el entorno de entrenamiento.

La prueba de Levene de igualdad de varianzas devolvió un nivel de significancia del 0.662, por lo cual se asumen varianzas iguales. Tomando en cuenta esto, la prueba T generó un nivel de significancia del 0.0000031, como indica la tabla 9.17. Con estos resultados, se rechaza la H_0 , es decir, los cambios que han existido en las frecuencias

Figura 9.4 Pruebas de normalidad de las diferencias de frecuencias de error para los dos períodos.

de error sí dependen de las mejoras que se han realizado en el laboratorio virtual de biotecnología.

Tabla 9.17 Resultados de la prueba T para las diferencias de errores cometidos.

t	4.896
gl	118
Sig. (bilateral)	0.0000031
Diferencia entre medias	0.11941
Diferencia de error estándar	0.02439
95 % de intervalo de confianza de la diferencia	Inferior 0.07111
	Superior 0.1677

Las tablas 9.19, 9.20 y 9.21 presentan las frecuencias de estudiantes de cada error que debería haberse visto afectado por los cambios antes mencionados, y para cada período. Se han eliminado de las tablas aquellos errores que tienen frecuencias de menos del 30 % en los dos períodos, puesto que al tener una frecuencia relativamente baja no son de interés para la tutoría automática. Además, en estas tablas se resaltan las frecuencias y las diferencias más altas con la gama de colores definida en la tabla 9.18.

Tabla 9.18 Gama de colores para valores de frecuencias de cada error por clúster.

	Frecuencias entre 25 % y 50 %
	Frecuencias sobre 50 %

VALIDACIÓN

La tabla 9.19 presenta las frecuencias asociadas a los cambios de mensajes que otorgan al estudiante una tutoría general, es decir, los cambios 2 y 4, 5.

Tabla 9.19 Frecuencias en estados de error con cambios de mensajes generales

Cambio	Accion	Fallo	Frec. 2013-2015 (%)	Frec. 2016 (%)	Diferencia
2	f1t20	f1t12	0.162	0.353	-0.191
		f1t13	0.147	0.353	-0.206
		f1t16	0.662	0.412	0.250
	Promedio		0.324	0.373	-0.049
4	f2t52	f2t34	0.456	0.353	0.103
		f2t35	0.529	0.412	0.118
		f2t36	0.485	0.294	0.191
		f2t37	0.588	0.412	0.176
		f2t38	0.574	0.353	0.221
		f2t39	0.574	0.412	0.162
		f2t40	0.647	0.353	0.294
		f2t41	0.515	0.471	0.044
	Promedio		0.546	0.382	0.164
5	f3t52	f3t33	0.397	0.176	0.221
		f3t34	0.588	0.118	0.471
		f3t35	0.309	0.118	0.191
		f3t36	0.309	0.118	0.191
		f3t37	0.397	0.176	0.221
	Promedio		0.400	0.141	0.259
Promedio Total			0.459	0.305	0.153

Por lo contrario, la tabla 9.20 muestra las frecuencias asociadas a cambios que han introducido mensajes de tutoría más específicos (cambios 1 y 3). Se incluye en esta tabla el cambio número 13, ya que influye en los mismos errores que el cambio 3, aunque no sea una mejora en la tutoría.

Tabla 9.20 Frecuencias en estados de error con cambios que incluyen mensajes específicos

Cambio	Accion	Fallo	Frec. 2013-2015 (%)	Frec. 2016 (%)	Diferencia
1	f1t20	f1t14	0.662	0.353	0.309
3 y 13	f1t25	faileddropbotella	0.382	0.118	0.265
Promedio			0.522	0.235	0.287

Finalmente, la tabla 9.21 expone las frecuencias de los errores asociados a cambios que se han realizado en la terminología utilizada en el laboratorio virtual, es decir, los cambios 7, 8, 9, 10, 11 y 12.

Tabla 9.21 Frecuencias en estados de error con cambios en el entorno virtual

Cambio	Accion	Fallo	Frec. 2013-2015 (%)	Frec. 2016 (%)	Diferencia
7 y 12	f1t20	f1t16	0.662	0.412	0.250
8	f2t52	f2t37	0.588	0.412	0.176
9	f2t52	f2t40	0.647	0.353	0.294
10	f3t52	f3t35	0.309	0.118	0.191
11	f3t52	f3t34	0.588	0.000	0.588
Promedio			0.559	0.259	0.300

Es importante recalcar que tanto los cambios de tutoría como los que se realizaron en el laboratorio virtual, fueron realizados gracias a que se encontraron los errores con la ayuda del visualizador del modelo colectivo de estudiantes.

La primera columna de las tablas 9.19, 9.20 y 9.21 muestra los cambios que se han hecho en el entorno de entrenamiento, representados por el número que precede a cada uno de ellos en las listas detalladas anteriormente. La segunda y tercera columnas presentan los códigos de la acción que ejecuta el alumno y el error que se produce respectivamente.

Tabla 9.22 Frecuencias en estados de error con cambios de mensaje al final de cada fase

Cambio	Fallo	Frec. 2013-2015 (%)	Frec. 2016 (%)	Diferencia
6	f1-f2-f3	0.368	0.059	0.309
	f1-f2	0.147	0.412	-0.265
	f1	0.088	0.353	-0.265

Finalmente, se ha incluido el cambio número 6 en la tabla 9.22 en lugar de en la tabla 9.19, a pesar de que se trate de un cambio de mensaje general, ya que no previene un solo error, sino cualquier error relevante que se pueda producir en las siguientes fases. Es decir, con la mejora realizada en el mensaje que se presenta al final de cada fase al alumno, se pretendía evitar que cometa menos errores relevantes en el futuro.

En esta última tabla la fila con el error *f1-f2-f3* indica la frecuencia de estudiantes que han cometido errores relevantes en las tres fases; la segunda con el error *f1-f2*, los

que han fallado en las dos primeras fases y no en la tercera; y la última fila (*f1*) los que se equivocaron solamente en esa fase.

9.3.2. Discusión

Mediante este análisis se ha intentado validar la utilidad del visualizador para encontrar patrones de comportamiento de los estudiantes dentro del proceso de entrenamiento. Para esto, se buscaron los errores más frecuentes, presentados en la tabla 9.13, y posteriormente se realizaron varios cambios tanto en la tutoría como en el entorno de entrenamiento (tablas 9.14, 9.15 y 9.16). Finalmente, se comprobó si estos errores reducían sus frecuencias en un nuevo grupo de estudiantes (tablas 9.19, 9.20, 9.21 y 9.22).

Lo más llamativo de los datos mostrados por las tablas 9.19, 9.20 y 9.21 es que ningún error ha tenido una frecuencia de más del 50 % entre los alumnos del segundo periodo. En cambio, entre los errores que han cometido los estudiantes del primer período, 14 de ellos tienen una frecuencia mayor al 50 %. Esto significa que en todos los errores que han sido más frecuentes entre el 2013 y el 2015, en el 2016 han bajado considerablemente la cantidad de estudiantes que han caído en estos. Es importante destacar que los errores que en el primer período han tenido frecuencias alrededor del 66 % y el 64 %, en el 2016 han pasado a tener frecuencias entre un 25 % y 30 %.

Esta reducción en la frecuencia de errores en el segundo período se ratifica mediante los promedios presentados en la última fila de las tablas, en donde se puede visualizar que los alumnos del 2016 cometieron un 15.3 % menos de errores con los nuevos mensajes de tutoría general; un 28.7 % menos con los de tutoría específica; y un 30 % con los cambios en la terminología, los cuales, aunque no hayan sido mejoras en la tutoría, también podrían ser considerados como cambios específicos que han sido de gran ayuda para los alumnos. Es razonable que estos dos últimos tipos de mejoras hayan conseguido una mayor reducción en la frecuencia de errores, puesto que ayudan de una forma más explícita a que el alumno no cometa el error.

A pesar de estas mejoras positivas en la reducción de errores, el cambio de tutoría número 2 no ha ayudado a que los estudiantes cometan menos errores en la creación del medio de cultivo, ya que el promedio de diferencias para este cambio es de -4.9 %. La solución para que se reduzca los errores en esta parte del proceso del laboratorio, podría ser agregar mensajes de tutoría más específicos, por ejemplo, para la acción de “agregar agar” (*f1t16*), en la cual se ha mantenido una frecuencia alta de errores. No obstante, debe ser el profesor de la materia el que decida si merece la pena aumentar la ayuda del tutor inteligente en esos pasos más problemáticos de la práctica, o si es preferible mantener este nivel de ayuda y mejorar la explicación del concepto en su clase magistral.

Gracias al cambio 6 se ha notado que los alumnos del 2016 han tomado la precaución de leer el proceso detenidamente si han cometido errores en una fase para así no cometer errores en las fases siguientes (tabla 9.22). Pese a que el porcentaje de errores del 2016 para solo la primera fase es mayor que en los años anteriores ($35.3\% > 8.8\%$), se puede concluir que este cambio ha sido exitoso, ya que el porcentaje de estudiantes que cometen errores en las tres fases en el 2016 es menor que en los otros años ($5.9\% < 36.8\%$). Es decir, que en el 2016 un 35.3 % de estudiantes han tomado más precauciones en las siguientes fases del proceso y no han cometido más errores, mientras que en los otros años solamente un 8.8 % de alumnos lo han hecho.

En resumen, las mejoras realizadas en la tutoría y en el laboratorio de biotecnología en varias acciones críticas (encontradas gracias al visualizador del MCCE), han reducido considerablemente la cantidad de errores producidos por los estudiantes. Es decir, ha ayudado a éstos a realizar correctamente el proceso, y si han cometido fallos, en los siguientes pasos del proceso han tomado más precauciones para evitar nuevos errores. Gracias al análisis estadístico antes detallado, se comprobó que esta reducción en la cantidad de errores en el 2016 depende significativamente de las mejoras realizadas.

CAPÍTULO 10

CONCLUSIONES Y TRABAJOS FUTUROS

10.1. Conclusiones

La investigación presentada en este trabajo representa un paso adelante hacia el desarrollo de un SIT que aproveche un modelo individual de un estudiante y un modelo de comportamiento colectivo de estudiantes calculado mediante la utilización de técnicas de EDM, para ofrecer una mejor retroalimentación de tutoría en un entorno de aprendizaje procedimental, sea cual sea su dominio de aplicación.

A la vista de los resultados obtenidos, podemos concluir que se han alcanzado todos los objetivos planteados inicialmente. Para cada uno de los tres objetivos generales, se ha presentado en esta tesis una contribución original respecto al estado del arte. A continuación, se van a resumir dichas contribuciones.

La primera contribución de esta tesis es un modelo de comportamiento colectivo de estudiantes para predecir los eventos que podría generar un estudiante en un entorno de entrenamiento procedimental. Este modelo se crea a partir de los registros de acciones o *logs* de los estudiantes que han utilizado con anterioridad el entorno de entrenamiento en el que se desean predecir las acciones. Estos *logs* son clusterizados para así clasificar a los alumnos, y para cada uno de los clústeres obtenidos, se construye un autómata extendido, el cual representa una síntesis de las secuencias de eventos generados por los estudiantes de un clúster.

El modelo propuesto ha sido validado utilizando los *logs* de estudiantes recogidos en el laboratorio virtual de biotecnología en los años 2013, 2014 y 2015. Esta validación demostró que las predicciones del modelo pueden ser lo suficientemente fiables como para dar soporte a una tutoría inteligente adaptada a cada tipo de alumno. En este sentido,

CONCLUSIONES Y TRABAJOS FUTUROS

como ya se mostró en el capítulo 9, con la mejor función de agrupamiento (eventos por zona) se han conseguido errores de predicción de alrededor del 30 % (para *soporte* = 0 y *confianza* $\approx$ 0.5). Además, conforme avance el estudiante en la práctica, debería disminuir ese error, ya que el modelo se ajustará mejor a los estudiantes bajo supervisión, al irse actualizando con cada nuevo evento generado.

Por otro lado, la mejor función de agrupamiento (eventos por zona) ha demostrado ser muy eficaz a la hora de agrupar a los alumnos atendiendo a su rendimiento durante una práctica. Esto posibilita la definición de una estrategia de tutoría adaptada a cada tipo de alumno.

La segunda contribución de este trabajo es un visualizador del modelo de comportamiento colectivo de estudiantes, el cual dibuja el autómata del clúster seleccionado y permite navegar e interactuar con él. Es decir, acercar o alejar la vista, buscar una acción determinada, o seleccionar un estado o transición para visualizar su información detallada. La visualización del modelo ha sido diseñada siguiendo las recomendaciones para dibujar grafos propuestas por Mazza (2009) y la taxonomía TTT descrita por Shneiderman (1996), la cual define los distintos tipos de interacción posibles entre los usuarios y los sistemas de presentación de información. Por otro lado, para solucionar el problema del posicionamiento de los estados, se ha adoptado la separación en las tres zonas que ya existen en el modelo, como recomienda Card et al. (1999).

Este visualizador fue creado con la finalidad de encontrar eventos críticos en el proceso de entrenamiento que sirvan para mejorar la tutoría automática y la docencia de un profesor. En este sentido, se ha demostrado que puede tener diversas aplicaciones tales como: asignar calificaciones a los alumnos que han realizado el proceso de entrenamiento; comparar el comportamiento de los alumnos de distintos clústeres; identificar errores comunes que indiquen al profesor que ciertos conceptos no se han asimilado correctamente; etc.

Con el fin de comprobar la utilidad del visualizador, se recurrió a él para encontrar los errores más típicos que los estudiantes cometieron en los años 2013, 2014 y 2015 (sección 9.3). Una vez identificados esos errores típicos, se realizaron cambios tanto en la configuración de la tutoría automática como en el laboratorio virtual, con el fin de prevenir estos errores, y posteriormente se repitió la práctica en el año 2016. Las mejoras realizadas fueron de tres tipos: inclusión de nuevos mensajes de tutoría generales, inclusión de nuevos mensajes de tutoría específicos y mejoras en la terminología utilizada dentro del entorno virtual.

El resultado fue que en el 2016 los estudiantes cometieron un promedio de 15.3 % menos errores con ayuda de los nuevos mensajes generales; un 28.7 % menos con ayuda

de los nuevos específicos; y un 30 % menos con las otras mejoras. Además, cabe destacar que ningún error relevante pedagógicamente obtuvo una frecuencia de más del 50 % en el 2016, mientras que en los años anteriores existieron 14 errores con una frecuencia mayor al 50 %.

La última contribución de esta tesis, pero no por ello la menos importante, es el diseño de una estrategia de tutoría que utilice los resultados obtenidos por el modelo del estudiante propuesto por Clemente et al. (2011); Clemente (2011) y el modelo de comportamiento colectivo de estudiantes.

Esta estrategia se inspira en la técnica pedagógica del andamiaje (*scaffolding*) (Wood et al. 1976). Siguiendo esta técnica, con cada evento generado por el alumno, el tutor automático actualiza el modelo del estudiante y el modelo colectivo, para que con los resultados que éstos generen, sea posible recalcular su zona de desarrollo próximo (Vygotsky 1978) y se pueda orientar adecuadamente la tutoría ofrecida a cada alumno.

10.2. Trabajos Futuros

Uno de los propósitos más importantes de una tesis doctoral es que a partir de ésta nazcan proyectos que incentiven a la comunidad científica a contribuir con trabajos, sean estos de investigación o no, para que el conocimiento humano mantenga su continuo crecimiento. De esta forma, algunos de los trabajos futuros que se podrían derivar de esta tesis son los siguientes:

Mejoras en el modelo de comportamiento colectivo de estudiantes. Como mejoras al modelo se pueden realizar las siguientes tareas:

- Crear un modelo de predicción del clúster que se asigna por defecto a los alumnos nuevos, es decir, el clúster que se asigna al inicio del entrenamiento al estudiante. Si este modelo no dispone de datos sobre el rendimiento del estudiante en la asignatura que se trata en el entrenamiento, emplearía datos de otras asignaturas, tests psicológicos o simplemente datos demográficos.
- Un algoritmo de minería de grafos que utilice las particularidades del autómata propuesto en este documento, como la separación por zonas, para encontrar patrones interesantes dentro de dicho autómata.

Estos patrones podrían destacarse en la visualización del modelo de comportamiento colectivo de estudiantes, cuando lo solicite el usuario, de forma que se resalten

los nodos que conforman el patrón, así como las transiciones que llevan a dicho estados.

- Validar el modelo con los registros de actividad de otros entornos de entrenamiento procedimental.

Mejoras en el visualizador Aparte de la mejora mencionada anteriormente relativa a los patrones encontrados por un algoritmo de minería de grafos, el visualizador del modelo propuesto en esta tesis admitiría las siguientes mejoras:

- Agregar una funcionalidad que ofrezca la edición de la retroalimentación o los mensajes de tutoría asociados a eventos o estados en el autómata extendido. Por ejemplo, cuando se encuentre una acción que necesita una mejor tutoría o un error que se crea que se puede evitar con un mensaje, resultaría muy útil poder agregar o modificar la tutoría existente para dicha acción o error por medio del propio visualizador.

Este proceso de edición debe ser intuitivo y fácil de usar por parte del usuario. Por ejemplo, si la acción o error está presente en varios clústeres, el sistema debe darse cuenta de ello y ayudar al usuario en la edición del mensaje no solo para el clúster en el que se ha encontrado visualmente la acción o error, sino también para todos los demás en los que se encuentre también presente dicha acción o error.

- Investigar qué otros tipos de gráficos puede ser útiles para el profesor. Es decir, gráficos que utilicen los registros de actividad de los alumnos para sintetizar dicha información, y así ayudar al profesor a mejorar la forma en la que imparte sus clases, o a un experto en pedagogía a mejorar la retroalimentación que otorga el tutor automático.
- Validar la usabilidad del visualizador. Para ello, sería necesario llevar a cabo pruebas con usuarios que sirvan para evaluar la eficacia con la que pueden realizar ciertas tareas.

Mejoras en la tutoría inteligente A continuación, se enumeran las posibles mejoras que se proponen para la estrategia de tutoría presentada en esta tesis. Algunas de estas mejoras requerirían implementar o mejorar otros componentes de la arquitectura propuesta por de Antonio et al. (2005); Imbert et al. (2007); Méndez Pozo (2008).

- Definir el proceso de tutoría autorregulada. Esto es, determinar la estrategia más idónea para presentar la lista de errores cometidos por el estudiante. Además,

desarrollar una herramienta, basada en el visualizador, que permita dibujar el autómatas con los pasos que ha realizado el estudiante.

Diseñar e implementar un módulo de generación automática de tests a partir de la secuencia acciones que ha seguido el alumno, además de los errores cometidos.

- Detectar factores no cognitivos (Sottolare, DeFalco, & Connor 2014). Por ejemplo, para identificar el nivel de compromiso de los estudiantes con el proceso de entrenamiento, el nivel de atención, el nivel de destreza en el manejo del entorno, la respuesta emocional del estudiante (frustración, alegría, etc.), etc.

Si bien el diseño de un tutor automático que pueda percibir las emociones de un estudiante es complicado, si se desarrolla correctamente, puede dar lugar a un sistema que tenga más éxito que un tutor humano en la identificación de los cambios de emocionales y pueda reaccionar a ellos más rápidamente (Holden & Sinatra 2014).

- Adaptar la estrategia de andamiaje (propuesta en el capítulo 8) para que tome en cuenta las emociones de los estudiantes (Graesser 2016; D'mello & Graesser 2012) calculadas por medio del módulo de detección de factores no cognitivos.
- Modificar el proceso de creación de mensajes de tutoría para aquellos casos en que el estudiante no encuentra un objeto. Es decir, que este tipo de mensajes se generen automáticamente utilizando un sistema que calcule la ruta que hay que seguir y obtenga la saliencia de dicho objeto (G. Lara, De Antonio, & Peña 2016; G. Lara, De Antonio, Peña, Muñoz, & Casillas 2017; G. Lara 2016).
- Agregar un proceso que genere mensajes o premios motivacionales, como, por ejemplo, logros o monedas como en los videojuegos, los cuales incentiven a los alumnos a continuar con el proceso de aprendizaje (Wilson et al. 2008).
- Validar empíricamente la estrategia de tutoría propuesta. Es decir, verificar que la retroalimentación entregada por el SIT al alumno, ayuda significativamente al alumno a sobrepasar su ZDP.

Bibliografía

- Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. En *Acm sigmod record* (Vol. 22, 2, pp. 207-216). ACM.
- Agrawal, R. & Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. En *Proceedings of the 20th International Conference on Very Large Data Bases* (pp. 487-499). VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
- Aleven, V. A. & Koedinger, K. R. (2002, marzo). An effective metacognitive strategy: learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science*, 26(2), 147-179. doi:http://dx.doi.org/10.1016/S0364-0213(02)00061-7
- Antunes, C. (2008). Acquiring Background Knowledge for Intelligent Tutoring Systems. En *EDM* (pp. 18-27).
- Arrow, K. J. (1962). The Economic Implications of Learning by Doing. *The Review of Economic Studies*, 29(3), 155-173.
- Arroyo, I., Cooper, D. G., Burleson, W., Woolf, B. P., Muldner, K., & Christopherson, R. (2009). Emotion Sensors Go To School. En *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling* (pp. 17-24). Amsterdam, The Netherlands, The Netherlands: IOS Press.
- Azevedo, R., Moos, D. C., Greene, J. A., Winters, F. I., & Cromley, J. G. (2007). Why is externally-facilitated regulated learning more effective than self-regulated learning with hypermedia? *Educational Technology Research and Development*, 56(1), 45-72. doi:10.1007/s11423-007-9067-0
- Baker, R. S. (2010). Data mining for education. *International encyclopedia of education*, 7, 112-118.
- Baker, R. S. (2014). Educational Data Mining: An Advance for Intelligent Systems in Education. *Intelligent Systems, IEEE*, 29(3), 78-82.
- Baker, R. S., Gowda, S. M., Wixon, M., Kalka, J., Wagner, A. Z., Salvi, A., ... Rosi, L. (2012). Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. *International Educational Data Mining Society*.
- Bang-Jensen, J. & Gutin, G. Z. (2008). *Digraphs: theory, algorithms and applications*. Springer Science & Business Media.
- Barnes, T. & Stamper, J. (2007). Toward the extraction of production rules for solving logic proofs. En *AIED07, 13th International Conference on Artificial Intelligence in Education, Educational Data Mining Workshop* (pp. 11-20).
- Barnes, T. & Stamper, J. (2010). Automatic Hint Generation for Logic Proof Tutoring Using Historical Data. *Journal of Educational Technology & Society*, 13(1), 3-12.
- Bartle, R. A. (2004). *Designing virtual worlds*. New Riders.

BIBLIOGRAFÍA

- Bienkowski, M., Feng, M., & Means, B. (2012). Enhancing teaching and learning through educational data mining and learning analytics: An issue brief. *US Department of Education, Office of Educational Technology*, 1-57.
- Bogarín, A., Romero, C., Cerezo, R., & Sánchez-Santillán, M. (2014). Clustering for Improving Educational Process Mining. En *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge* (pp. 11-15). New York, NY, USA: ACM.
- Bondy, J. A. & Murty, U. S. R. (1976). *Graph Theory With Applications*. Oxford, UK, UK: Elsevier Science Ltd.
- Boulos, M. N. K., Hetherington, L., & Wheeler, S. (2007). Second Life: an overview of the potential of 3-D virtual worlds in medical and health education. *Health Information and Libraries Journal*, 24(4), 233-245.
- Bransford, J., Barron, B., Pea, R. D., Meltzoff, A., Kuhl, P., Bell, P., ... Reeves, B. (2005). Foundations and opportunities for an interdisciplinary science of learning. *The Cambridge handbook of the learning sciences*, 39-77.
- Brookfield, S. D. (1988). Understanding and Facilitating Adult Learning. *School Library Media Quarterly*, 16(2), 99-105.
- Buijs, J. C., van Dongen, B. F., van der Aalst, W. M., & P. (2014, marzo). Quality Dimensions in Process Discovery: The Importance of Fitness, Precision, Generalization and Simplicity. *International Journal of Cooperative Information Systems*, 23(01), 1440001. doi:10.1142/S0218843014400012
- Bull, S. & Kay, J. (2007). Student models that invite the learner in: The SMILI() Open learner modelling framework. *International Journal of Artificial Intelligence in Education*, 17(2), 89-120.
- Bunge, M. (1980). *Epistemología: curso de actualización*. Siglo xxi.
- Carbonell, J. R. (1970). AI in CAI: An Artificial-Intelligence Approach to Computer-Assisted Instruction. *Man-Machine Systems, IEEE Transactions on*, 11(4), 190-202.
- Card, S. K., Mackinlay, J. D., & Shneiderman, B. (1999). *Readings in information visualization: using vision to think*. San Francisco, CA, USA: Morgan Kaufmann.
- Chi, M. & VanLehn, K. (2010). Meta-Cognitive Strategy Instruction in Intelligent Tutoring Systems: How, When, and Why. *Journal of Educational Technology & Society*, 13(1), 25-39.
- Clark, P. & Niblett, T. (1989). The CN2 Induction Algorithm. *Mach. Learn.* 3(4), 261-283. doi:10.1023/A:1022641700528
- Clemente, J. (2011). *Una Propuesta de Modelado del Estudiante Basada en Ontologías y Diagnóstico Pedagógico-Cognitivo no Monótono* (Tesis doctoral, Universidad Politécnica de Madrid).
- Clemente, J., Ramírez, J., & de Antonio, A. (2011, julio). A proposal for student modeling based on ontologies and diagnosis rules. *Expert Systems with Applications*, 38(7), 8066-8078. doi:10.1016/j.eswa.2010.12.146
- Clemente, J., Ramírez, J., & de Antonio, A. (2014, febrero). Applying a student modeling with non-monotonic diagnosis to Intelligent Virtual Environment for Training/Instruction. *Expert Systems with Applications*, 41(2), 508-520. doi:http://dx.doi.org/10.1016/j.eswa.2013.07.077
- Coates, G. (1992). Program from Invisible Site-a virtual sho, a multimedia performance work presented by George Coates Performance Works. *San Francisco, CA*.
- Cook, D. J. & Holder, L. B. (2006). *Mining graph data*. John Wiley & Sons.

- Corbett, A. T. & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4), 253-278. doi:10.1007/BF01099821
- Dale, E. (1969). *Audiovisual methods in teaching* (3.^a ed.). The Dryden Press.
- Dalgarno, B. (2002). The potential of 3D virtual learning environments: A constructivist analysis. *Electronic Journal of Instructional Science and Technology*, 1-19. doi:Journal-article(Web)
- Dalgarno, B. & Lee, M. J. W. (2010). What are the learning affordances of 3-D virtual environments? *British Journal of Educational Technology*, 41(1), 10-32. doi:10.1111/j.1467-8535.2009.01038.x
- de Jesús Sen, Á. (2016, enero). *Mejoras y extensiones en el Front-End de un modelo de predicción de acciones de un estudiante en un laboratorio virtual 3D*, Universidad Politécnica de Madrid. Madrid, España.
- de Antonio, A., Ramírez, J., Imbert, R., & Méndez, G. (2005). Intelligent virtual environments for training: An agent-based approach. *Multi-Agent Systems and Applications IV*, 82-91.
- de Antonio, A. & Troncoso, B. (2008). Scaffolding Tutoring Strategy on Virtual Environments for Training. *Ingeniare: Revista Chilena de Ingeniería*, 16(2), 220-231.
- Dekker, G. W., Pechenizkiy, M., & Vleeshouwers, J. M. (2009). Predicting Students Drop Out: A Case Study. En *International Working Group on Educational Data Mining*. ERIC.
- Dib, C. Z. & Barojas, J. (1988). Formal, non-formal and informal education: concepts/applicability. En *AIP conference proceedings* (Vol. 173, 1, pp. 300-315). AIP.
- Diligenti, M., Frasconi, P., & Gori, M. (2003). Hidden tree Markov models for document image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4), 519-523. doi:10.1109/TPAMI.2003.1190578
- D'mello, S. & Graesser, A. (2012). AutoTutor and Affective Autotutor: Learning by Talking with Cognitively and Emotionally Intelligent Computers That Talk Back. *ACM Trans. Interact. Intell. Syst.* 2(4), 23:1-23:39. doi:10.1145/2395123.2395128
- Donath, J., Dragulescu, A., Zinman, A., Viégas, F., & Xiong, R. (2010). Data Portraits. En *ACM SIGGRAPH 2010 Art Gallery* (pp. 375-383). SIGGRAPH '10. New York, NY, USA: ACM. doi:10.1145/1836786.1836793
- Dotor Jara, D. (2015, enero). *Rediseño del tutor automático de un laboratorio virtual* (Tesis doctoral, Universidad Politécnica de Madrid, Boadilla del Monte, Madrid).
- Džeroski, S. (2001). *Relational data mining*. Springer Berlin Heidelberg.
- Eco, U. (2001). *Como se hace una tesis*. Gedisa editorial.
- Edelkamp, S. & Schrödl, S. (2003). Route planning and map inference with global positioning traces. En *Computer Science in Perspective* (pp. 128-151). Springer.
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. En *Second International Conference on Knowledge Discovery and Data Mining (KDD-96)* (Vol. 96, 34, pp. 226-231). AAAI Press.
- Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996). Knowledge Discovery and Data Mining: Towards a Unifying Framework. *KDD*, 96, 82-88.
- Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996, marzo). From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17(3), 37. doi:10.1609/aimag.v17i3.1230

BIBLIOGRAFÍA

- Fournier-Viger, P., Faghihi, U., Nkambou, R., & Nguifo, E. (2010). Exploiting sequential patterns found in users' solutions and virtual tutor behavior to improve assistance in ITS. *Educational Technology and Society*, 13(1).
- Fournier-Viger, P., Nkambou, R., & MephuNguifo, E. (2010). Learning procedural knowledge from user solutions to ill-defined tasks in a simulated robotic manipulator BT - Handbook of Educational Data Mining. En C. Romero, S. Ventura, M. Pechenizkiy, & R. S. J. d. Baker (Eds.), *Handbook of Educational Data Mining* (32, pp. 451-465). CRC Press.
- Fournier-Viger, P., Nkambou, R., Nguifo, E. M., Mayers, A., & Faghihi, U. (2013). A Multiparadigm Intelligent Tutoring System for Robotic Arm Training. *Learning Technologies, IEEE Transactions on*, 6(4), 364-377.
- Fournier-Viger, P., Nkambou, R., & Nguifo, E. (2008). A data mining framework for the acquisition of procedural knowledge in intelligent tutoring systems. En *Proceedings - ICCE 2008: 16th International Conference on Computers in Education*.
- Freire, P. (2005). *Pedagogía del oprimido* (2.^a ed.). México: Siglo XXI.
- Gigante, M. A. (1993). *Virtual reality: definitions, history and applications*. London: Academic Press London.
- Godoy, D. & Amandi, A. (2010). Link Recommendation in E-Learning Systems Based on Content-Based Student Profiles. En *Handbook of Educational Data Mining* (Cap. 19, pp. 273-286). CRC Press.
- Goldberg, B. & Spain, R. (2014). Creating the Intelligent Novice: Supporting Self-Regulated Learning and Metacognition in Educational Technology. En *Design Recommendations for Intelligent Tutoring Systems* (Cap. 10, Vol. 2, pp. 109-134). Orlando: U.S. Army Research Laboratory.
- Graesser, A. C. (2016). Conversations with AutoTutor Help Students Learn. *International Journal of Artificial Intelligence in Education*, 1-9. doi:10.1007/s40593-015-0086-4
- Greenbaum, P. (1992). The lawnmower man. *Film and video*, 9(3), 58-62.
- Guerrero, R. & Ramírez, J. (2015). *Especificación del comportamiento de un avatar en un mundo virtual mediante una ontología*.
- Gusfield, D. (1997). *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. New York, NY, USA: Cambridge University Press.
- Hadwin, A. F., Wozney, L., & Pontin, O. (2005). Scaffolding the Appropriation of Self-regulatory Activity: A Socio-cultural Analysis of Changes in Teacher-student Discourse about a Graduate Research Portfolio. *Instructional Science*, 33(5), 413-450. doi:10.1007/s11251-005-1274-7
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.* 11(1), 10-18. doi:10.1145/1656274.1656278
- Hall, M., Witten, I., & Frank, E. (2011). *Data mining: Practical machine learning tools and techniques* (3.^a ed.). Burlington: Morgan Kaufmann.
- Han, J., Kamber, M., & Pei, J. (2012). *Data mining: concepts and techniques* (3.^a ed.). Elsevier Inc.
- Hand, D. J., Mannila, H., & Smyth, P. (2001). *Principles of data mining*. Cambridge: MIT press.
- Hasher, M., Hornik, K., Grun, B., & Buchta, C. (2005). Introduction to arules - A computational environment for mining association rules and frequent item sets. *Journal of Statistical Software*.

- Heaton, J. (2007). *Introduction to linden scripting language for Second Life*. Heaton Research, Inc.
- Heilig, M. L. (1962, agosto). Sensorama simulator. Google Patents.
- Hernández Orallo, J., Ramírez Quintana, M. J., & Ferri Ramírez, C. (2004). *Introducción a la Minería de Datos*. Madrid: Pearson Educación.
- Hershkovitz, A. & Nachmias, R. (2009). Learning about online learning processes and students' motivation through Web usage mining. *Interdisciplinary Journal of Knowledge and Learning Objects*, 197-214.
- Hofmann, M. & Klinkenberg, R. (2013). *RapidMiner: Data mining use cases and business analytics applications*. CRC Press.
- Hofmeyr, S. A., Forrest, S., & Somayaji, A. (1998). Intrusion Detection Using Sequences of System Calls. *J. Comput. Secur.* 6(3), 151-180.
- Holden, H. K. & Sinatra, A. M. (2014). A Guide to Scaffolding and Guided Instructional Strategies for ITSs. En *Design Recommendations for Intelligent Tutoring Systems* (Cap. 22, pp. 265-281). Orlando: U.S. Army Research Laboratory.
- Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2006). *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.
- Huang, K. (2015, julio). *Visualization of a student predictive model for 3D virtual environments driven to procedural training*, Universidad Politécnica de Madrid.
- Imbert, R., Sánchez, L., de Antonio, A., Méndez, G., & Ramírez, J. (2007). A multiagent extension for virtual reality based intelligent tutoring systems. En *International Conference on Advanced Learning Technologies* (pp. 82-84). IEEE.
- Jackson, S. L., Krajcik, J., & Soloway, E. (1998). The Design of Guided Learner-adaptable Scaffolding in Interactive Learning Environments. En *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 187-194). CHI '98. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. doi:10.1145/274644.274672
- Järvelä, S. & Järvenoja, H. (2011). Socially constructed self-regulated learning and motivation regulation in collaborative learning groups. *Teachers College Record*, 113(2), 350-374.
- Jin, W., Barnes, T., Stamper, J., Eagle, M., Johnson, M., & Lehmann, L. (2012). *Program representation for automatic hint generation for a data-driven novice programming tutor*. doi:10.1007/978-3-642-30950-2_40
- Jin, W., Lehmann, L., Johnson, M., Eagle, M., Mostafavi, B., Barnes, T., & Stamper, J. (2011). Towards automatic hint generation for a data-driven novice programming tutor. En *Workshop on Knowledge Discovery in Educational Data, 17th ACM Conference on Knowledge Discovery and Data Mining*. Citeseer.
- Johnson, C. I. & Mayer, R. E. (2010, noviembre). Applying the self-explanation principle to multimedia learning in a computer-based game-like environment. *Computers in Human Behavior*, 26(6), 1246-1252. doi:http://dx.doi.org/10.1016/j.chb.2010.03.025
- Johnson, W. L. & Rickel, J. (1997). Steve: an animated pedagogical agent for procedural training in virtual environments. *SIGART Bulletin*, 8, 1-4.
- Kaminski, M. & Pinter, S. S. (1992, junio). Finite automata on directed graphs. *Journal of Computer and System Sciences*, 44(3), 425-446. doi:10.1016/0022-0000(92)90012-8
- Kitchner, K. S. (1983). Cognition, metacognition, and epistemic cognition. *Human development*, 26(4), 222-232.

BIBLIOGRAFÍA

- Knorr, E. M. & Ng, R. T. (1998). Algorithms for Mining Distance-Based Outliers in Large Datasets. En *Proceedings of the 24rd International Conference on Very Large Data Bases* (pp. 392-403). VLDB '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
- Lajoie, S. P. (2005). Extending the Scaffolding Metaphor. *Instructional Science*, 33(5), 541-557. doi:10.1007/s11251-005-1279-2
- Lajoie, S. P. & Derry, S. J. (1993). *Computer environments as cognitive tools for enhancing learning*. Routledge.
- Lara, G. (2016, junio). *Modelo computacional para la generación de indicaciones en la localización de objetos en entornos virtuales : aspectos espaciales y perceptivos*, ETSI_Informatica.
- Lara, G., De Antonio, A., & Peña, A. (2016). Computerized spatial language generation for object location. *Virtual Reality*, 20(3), 183-192. doi:10.1007/s10055-016-0289-5
- Lara, G., De Antonio, A., Peña, A., Muñoz, M., & Casillas, L. (2017). Scheme for the automatic generation of directions to locate objects in virtual environments. En J. Mejia, M. Muñoz, Á. Rocha, T. San Feliu, & A. Peña (Eds.), *Trends and Applications in Software Engineering: Proceedings of CIMPS 2016* (pp. 123-132). Cham: Springer International Publishing. doi:10.1007/978-3-319-48523-2_12
- Lara, J. A., Lizcano, D., Martínez, M. A., Pazos, J., & Riera, T. (2014, marzo). A system for knowledge discovery in e-learning environments within the European Higher Education Area – Application to student data from Open University of Madrid, UDIMA. *Computers & Education*, 72, 23-36. doi:http://dx.doi.org/10.1016/j.compedu.2013.10.009
- Leontiev, A. N. (1981). *Problems of the Development of the Mind*. Moscow: Moscow: Progress.
- Lewin, K. (1951). *Field theory in social science: selected theoretical papers*. (Dorwin Cartwright, Ed.). Oxford: Harpers.
- Livingstone, D., Kemp, J., & Edgar, E. (2008). From multi-user virtual environment to 3D virtual learning environment. *ALT-J: Research in Learning Technology*, 16(3), 139-150.
- Lucci, M. A. (2006). La propuesta de Vygotsky: la psicología socio-histórica. *Profesorado: Revista de curriculum y formación del profesorado*, 10(2), 10.
- Luria, A. R., Leontiev, A., & Vigotsky, L. S. (2004). *Psicología y pedagogía*. Ediciones Akal.
- Mannila, H. & Meek, C. (2000). Global Partial Orders from Sequential Data. En *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 161-168). KDD '00. New York, NY, USA: ACM. doi:10.1145/347090.347122
- Mannila, H., Toivonen, H., & Inkeri Verkamo, A. (1997). Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery*, 1(3), 259-289. doi:10.1023/A:1009748302351
- Mannila, H., Toivonen, H., & Verkamo, A. I. (1995). Discovering frequent episodes in sequences. En *In 1st Conference on Knowledge Discovery and Data Mining (KDD'95)* (pp. 210-215).
- Marsh, T. (2003). Staying there: an activity-based approach to narrative design and evaluation as an antidote to virtual corpsing. En *Being there: Concepts, effects and measurement of user presence in synthetic environments, studies on new technologies and practices in communication series*. Amsterdam, The Netherlands: Citeseer.

- Mavrikis, M. (2010, diciembre). Modelling student interactions in intelligent learning environments: constructing bayesian networks from data. *International Journal on Artificial Intelligence Tools*, 19(06), 733-753. doi:10.1142/S0218213010000406
- Mazza, R. (2004). *Using information visualisation to facilitate instructors in web-based distance learning* (Tesis doctoral, Università della Svizzera italiana).
- Mazza, R. (2009). *Introduction to information visualization*. Springer Science & Business Media.
- Mazza, R. (2010). Visualization in educational environments. En C. Romero, S. Ventura, M. Pechenizkiy, & R.S.J. d. Baker (Eds.), *Handbook of educational data mining* (pp. 9-26).
- Mazza, R. & Botturi, L. (2007). Monitoring an Online Course with the GISMO Tool: A Case Study. *Journal of Interactive Learning Research*, 18(2), 251-265.
- Mazza, R. & Dimitrova, V. (2007). CourseVis: A graphical student monitoring tool for supporting instructors in web-based distance courses. *International Journal of Human-Computer Studies*, 65(2), 125-139. doi:10.1016/j.ijhcs.2006.08.008
- McGreal, R. (2004). Learning objects: A practical definition. *International Journal of Instructional Technology and Distance Learning (IJITDL)*, 9(1).
- McKnight, C., Dillon, A., & Richardson, J. (1991). *Hypertext in context*. Cambridge University Press.
- Méndez Pozo, G. (2008). *Una arquitectura software basada en agentes y recomendaciones metodológicas para el desarrollo de entornos virtuales de entrenamiento con tutoría inteligente* (Tesis doctoral).
- Michael, D. R. & Chen, S. L. (2005). *Serious Games: Games That Educate, Train, and Inform*. Muska & Lipman/Premier-Trade.
- Microsoft. (2016). Microsoft Sequence Clustering Algorithm Technical Reference.
- Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., & Euler, T. (2006). Yale: Rapid prototyping for complex data mining tasks. En *12th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 935-940). ACM.
- Mitra, S., Pal, S. K., & Mitra, P. (2002). Data mining in soft computing framework: a survey. *IEEE Transactions on Neural Networks*, 13(1), 3-14. doi:10.1109/72.977258
- Moshman, D. (1982, diciembre). Exogenous, endogenous, and dialectical constructivism. *Developmental Review*, 2(4), 371-384. doi:10.1016/0273-2297(82)90019-3
- Mostow, J. & Beck, J. (2006). Some Useful Tactics to Modify, Map and Mine Data from Intelligent Tutors. *Nat. Lang. Eng.* 12(2), 195-208.
- Murray, T. (1999). Authoring Intelligent Tutoring Systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education (IJAIED)*, 10, 98-129.
- Narayanasamy, V., Wong, K. W., Fung, C. C., & Rai, S. (2006). Distinguishing Games and Simulation Games from Simulators. *Comput. Entertain.* 4(2). doi:10.1145/1129006.1129021
- Olney, A. M. (2014). Scaffolding Made Visible. En *Design Recommendations for Intelligent Tutoring Systems* (Cap. 26, pp. 327-340). Orlando: U.S. Army Research Laboratory.
- Organisation for Economic Co-operation and Development. (2014). *PISA 2012 Results: What Students Know and Can Do (Volume I, Revised Edition, February 2014): Student Performance in Mathematics, Reading and Science*. OECD Publishing.
- Papadopoulos, L., Pentzou, A.-E., Louloudiadis, K., & Tsiatsos, T.-K. (2013). Design and Evaluation of a Simulation for Pediatric Dentistry in Virtual Worlds. *Journal of medical Internet research*, 15(10), e240.

- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc.
- Peña-Ayala, A. (2014, marzo). Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications*, 41(4), 1432-1462. doi:10.1016/j.eswa.2013.08.042
- Perera, D., Kay, J., Koprinska, I., Yacef, K., & Zaïane, O. R. (2009). Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(6), 759-772. doi:10.1109/TKDE.2008.138
- Piaget, J. (1969). *The intellectual development of the adolescent*.
- Piaget, J. & Duckworth, E. (1970, enero). Genetic Epistemology. *American Behavioral Scientist*, 13(3), 459-480. doi:10.1177/000276427001300320
- Piaget, J., Inhelder, B., & Chipman, H. H. (1976). Piaget and His School: A Reader in Developmental Psychology. Eds. Bärbel Inhelder and Harold H. Chipman, Charles Zwingmann. NY: Springer-Verlag.
- Pintrich, P. R. (2004). A Conceptual Framework for Assessing Motivation and Self-Regulated Learning in College Students. *Educational Psychology Review*, 16(4), 385-407. doi:10.1007/s10648-004-0006-x
- Pol, J., Volman, M., & Beishuizen, J. (2010). Scaffolding in Teacher-Student Interaction: A Decade of Research. *Educational Psychology Review*, 22(3), 271-296. doi:10.1007/s10648-010-9127-6
- Polsani, P. R. (2006). Use and abuse of reusable learning objects. *Journal of Digital Information*, 3(4).
- Porayska-Pomsta, K., Mavrikis, M., D'Mello, S., Conati, C., & Baker, R. S. J. d. (2013). Knowledge Elicitation Methods for Affect Modelling in Education. *Int. J. Artif. Intell. Ed.* 22(3), 107-140. doi:10.3233/JAI-130032
- Quinlan, J. R. (1986). Induction of Decision Trees. *Mach. Learn.* 1(1), 81-106. doi:10.1023/A:1022643204877
- Rabbany, R., Takaffoli, M., & Zaïane, O. R. (2011). Analyzing participation of students in online courses using social network analysis techniques. En *Proceedings of educational data mining*. Citeseer.
- Ramírez, J., Rico, M., Riofrío-Luzcando, D., Berrocal-Lobo, M., & de Antonio, A. (2017, mayo). Students' Evaluation of a Virtual World for Procedural Training in a Tertiary-Education Course. *Journal of Educational Computing Research*, 0735633117706047. doi:10.1177/0735633117706047
- Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N. T., Koedinger, K. R., Junker, B., ... Choksey, S. (2005). The Assistment project: Blending assessment and assisting. *Proceedings of the 12th Annual Conference on Artificial Intelligence in Education*, 555-562.
- Reffay, C. & Chanier, T. (2003). How Social Network Analysis can help to Measure Cohesion in Collaborative Distance-Learning. En B. Wasson, S. Ludvigsen, & U. Hoppe (Eds.), *Designing for Change in Networked Learning Environments: Proceedings of the International Conference on Computer Support for Collaborative Learning 2003* (pp. 343-352). Dordrecht: Springer Netherlands. doi:10.1007/978-94-017-0195-2_42
- Rico, M., Ramírez, J., Riofrío, D., Berrocal-Lobo, M., & De Antonio, A. (2012). An architecture for virtual labs in engineering education. En *Global Engineering Education Conference (EDUCON), 2012 IEEE* (pp. 1-5).

- Rico, M., Ramírez, J., Riofrío-Luzcando, D., & Berrocal-Lobo, M. (2017, febrero). A Cost-Effective Approach for Procedural Training in Virtual Worlds. *Journal of Universal Computer Science*, 23(2), 208-232.
- Rieber, L. P. (1992). Computer-based microworlds: A bridge between constructivism and direct instruction. *Educational Technology Research and Development*, 40(1), 93-106. doi:10.1007/BF02296709
- Rieber, L. P. (2005). Multimedia learning in games, simulations, and microworlds. *The Cambridge handbook of multimedia learning*, 549-567.
- Riofrío-Luzcando, D., Ramírez, J., & Berrocal-Lobo, M. (2012). *Diseño e Implementación de un Laboratorio Virtual de Biotecnología*. (Tesis doctoral, ETS de Ingenieros Informáticos, Universidad Politécnica de Madrid).
- Ritter, S., Anderson, J., Koedinger, K., & Corbett, A. (2007). Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*, 14(2), 249-255. doi:10.3758/BF03194060
- Ritter, S., Carlson, R., Sandbothe, M., & Fancsali, S. E. (2015). Carnegie Learning's Adaptive Learning Products. En *Educational Data Mining 2015: 8th International Conference on Educational Data Mining*. Madrid.
- Robins, S. & Mayer, R. E. (1993). Schema training in analogical reasoning. *Journal of educational Psychology*, 85(3), 529.
- Rokach, L. & Maimon, O. (2005). Clustering Methods. En *Data Mining and Knowledge Discovery Handbook* (pp. 321-352). Boston, MA: Springer US. doi:10.1007/0-387-25465-X_15
- Romero, C., Espejo, P. G., Zafra, A., Romero, J. R., & Ventura, S. (2013, marzo). Web usage mining for predicting final marks of students that use Moodle courses. *Computer Applications in Engineering Education*, 21(1), 135-146. doi:10.1002/cae.20456
- Romero, C. & Ventura, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33(1), 135-146.
- Romero, C. & Ventura, S. (2010). Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6), 601-618.
- Romero, C. & Ventura, S. (2013). Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(1), 12-27.
- Romero, C., Ventura, S., Pechenizkiy, M., & Baker, R. S. (2010). *Handbook of Educational Data Mining*. CRC Press.
- Rowe, J., Mott, B., McQuiggan, S., Robison, J., Lee, S., & Lester, J. (2009). Crystal island: A narrative-centered learning environment for eighth grade microbiology. En *workshop on intelligent educational games at the 14th international conference on artificial intelligence in education, Brighton, UK* (pp. 11-20).
- Saracevic, M. H. (2014). Concept and Types of Virtual Environments: Research about positive impact on teaching and learning. *UNITE: University Journal of Information Technology and Economics*, 1(1).
- Scheuer, O. & McLaren, B. (2012). Educational Data Mining. En N. Seel (Ed.), *Encyclopedia of the Sciences of Learning SE - 618* (pp. 1075-1079). Springer US. doi:10.1007/978-1-4419-1428-6_618
- Schraw, G., Dunkle, M. E., & Bendixen, L. D. (1995). Cognitive processes in well-defined and ill-defined problem solving. *Applied Cognitive Psychology*, 9(6), 523-538.

BIBLIOGRAFÍA

- Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. En *Visual Languages, 1996. Proceedings., IEEE Symposium on* (pp. 336-343). IEEE.
- Simons, K. D. & Klein, J. D. (2007). The Impact of Scaffolding and Student Achievement Levels in a Problem-based Learning Environment. *Instructional Science*, 35(1), 41-72. doi:10.1007/s11251-006-9002-5
- Singh, Y. & Chauhan, A. S. (2009). Neural networks in data mining. *Journal of Theoretical and Applied Information Technology*, 5(6), 36-42.
- Sleeman, D. & Brown, J. S. (1982). *Intelligent Tutoring Systems*. London : Academic Press.
- Slonim, N., Friedman, N., & Tishby, N. (2002). Unsupervised Document Classification Using Sequential Information Maximization. En *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 129-136). SIGIR '02. New York, NY, USA: ACM. doi:10.1145/564376.564401
- Smyth, P. (1997). Clustering Sequences with Hidden Markov Models. En *Advances in Neural Information Processing Systems* (pp. 648-654). MIT Press.
- Sottolare, R. A., DeFalco, J. A., & Connor, J. (2014). A Guide to Instructional Techniques, Strategies and Tactics to Manage Learner Affect, Engagement, and Grit. En *Design Recommendations for Intelligent Tutoring Systems* (Cap. 2, p. 7). U.S. Army Research Laboratory.
- Srikant, R. & Agrawal, R. (1996). Mining Sequential Patterns: Generalizations and Performance Improvements. En *Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology* (pp. 3-17). EDBT '96. London, UK, UK: Springer-Verlag.
- Steuer, J. (1992, diciembre). Defining Virtual Reality: Dimensions Determining Telepresence. *Journal of Communication*, 42(4), 73-93. doi:10.1111/j.1460-2466.1992.tb00812.x
- Sukhija, K., Jindal, M., & Aggarwal, N. (2015). The recent state of educational data mining: A survey and future visions. En *IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE), 2015* (pp. 354-359). doi:10.1109/MITE.2015.7375344
- Suraweera, P. & Mitrovic, A. (2002). KERMIT: A Constraint-Based Tutor for Database Modeling. En S. A. Cerri, G. Gouardères, & F. Paraguaçu (Eds.), *Intelligent Tutoring Systems: 6th International Conference, ITS 2002 Biarritz, France and San Sebastian, Spain, June 2-7, 2002 Proceedings* (pp. 377-387). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/3-540-47987-2_41
- Tan, P.-N., Steinbach, M., & Kumar, V. (2013). *Introduction to data mining* (1st). London: Pearson Education Limited.
- Tane, J., Schmitz, C., & Stumme, G. (2004). Semantic Resource Management for the Web: An e-Learning Application. En *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters* (pp. 1-10). WWW Alt. '04. New York, NY, USA: ACM. doi:10.1145/1013367.1013369
- Tang, T. Y. & McCalla, G. (2005). Smart Recommendation for an Evolving E-Learning System: Architecture and Experiment. *International Journal on ELearning*, 4(1), 105-129.

- Trcka, N., Pechenizkiy, M., & van der Aalst, W. (2010). Process mining from educational data. En C. Romero, S. Ventura, M. Pechenizkiy, & R. S. J. d. Baker (Eds.), *Handbook of Educational Data Mining* (pp. 75-92). CRC Press.
- Vellido, A., Castro, F., & Nebot, A. (2010). Clustering educational data. En C. Romero, S. Ventura, M. Pechenizkiy, & R. S. J. d. Baker (Eds.), *Handbook of Educational Data Mining* (6, pp. 75-92). CRC Press.
- Vicente, F., Adjei, S., Colombo, T., & Heffernan, N. (2015). Building Models to Predict Hint-or-Attempt Actions of Students. En *Educational Data Mining 2015: 8th International Conference on Educational Data Mining*. Madrid.
- Von Glasersfeld, E. (1984). An introduction to radical constructivism. *The invented reality*, 17-40.
- Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Harvard university press.
- Vygotsky, L. S. (1984, enero). Learning and intellectual development during school age. *Infancia y Aprendizaje*, 7(27-28), 105-116. doi:10.1080/02103702.1984.10822045
- Vygotsky, L. S. (1987). Lecture 4. Emotions and their development in childhood. *Collected works*, 1, 325-337.
- Wang, L. (2005). *Support vector machines: theory and applications*. Springer Science & Business Media.
- Watzlawick, P. (1984). *The Invented reality: how do we know what we believe we know?: contributions to constructivism*. norton.
- Weiss, S. M. & Kulikowski, C. A. (1991). *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
- Wenger, E. (1987). *Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge*. Los ALtos: Morgan Kaufmann.
- Wilson, K. A., Bedwell, W. L., Lazzara, E. H., Salas, E., Burke, C. S., Estock, J. L., ... Conkey, C. (2008). Relationships between game attributes and learning outcomes: Review and research proposals. *Simulation & Gaming*.
- Winn, W. (1993). A conceptual basis for educational applications of virtual reality. *Technical Publication R-93-9, Human Interface Technology Laboratory of the Washington Technology Center, Seattle: University of Washington*.
- Wood, D. (2003). The why? what? when? and how? of tutoring: The development of helping and tutoring skills in children. *Literacy, Teaching and Learning*, 7(1/2), 1.
- Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving*. *Journal of child psychology and psychiatry*, 17(2), 89-100.
- Yelland, N. & Masters, J. (2007, abril). Rethinking scaffolding in the information age. *Computers & Education*, 48(3), 362-382. doi:10.1016/j.compedu.2005.01.010
- Zapata-Rivera, J.-D. & Greer, J. (2001). Externalising learner modelling representations. En *Proceedings of Workshop on External Representations of AIED: Multiple Forms and Multiple Roles* (pp. 71-76). Citeseer.
- Zimmerman, B. J. (1990, enero). Self-Regulated Learning and Academic Achievement: An Overview. *Educational Psychologist*, 25(1), 3-17. doi:10.1207/s15326985ep2501_2
- Zimmerman, B. J. & Campillo, M. (2003). Motivating self-regulated problem solvers. *The psychology of problem solving*, 233-262.

APÉNDICE A

PROTOCOLO Y CONFIGURACIÓN DE TUTORÍA DEL LABORATORIO VIRTUAL DE BIOTECNOLOGÍA

PROTOCOLO Y CONFIGURACIÓN DE TUTORÍA DEL LABORATORIO VIRTUAL DE BIOTECNOLOGÍA

[illegible]

Identificador	Nombre Acción	Mensaje Acción Validada	Mensaje de Tutoría	Dependencias	Mensajes de Error por Dependencias	Incompatibilidades	Mensajes de Error por Incompatibilidades
f132	solaridad/cachuna	0-Botella en cachuna	Agrega todo microbiológico del Agnético.	f131	0-No se ha esterilizado por completo el aire	f142	0-S2 ha apagado la cabina antes de terminar el proceso.
f133	agregata	0-Botella en cachuna	0-Botella en cachuna	f132	1-La botella debe estar en la cabina seleccionada	f142	0-S2 ha apagado la cabina antes de terminar el proceso.
f134	entregat/bandeja	1-Bandeja con bote entregado al inventario	Toma la bandeja de tu inventario y súbala en la cabina				
f135	solar/bandeja	0-Bandeja en cabina	Toca la bandeja para verter el contenido de la botella en los botes.	f130-f131	0-No se ha encendido la cabina de flujo/03-No se ha esterilizado por completo el aire	f142	0-S2 ha apagado la cabina antes de terminar el proceso.
f136	lineabotes	0-Botes llenos con mezcla	Toca el esterilizador para esterilizar el instrumental.	f135-f132-f129	1-Debe haber una bandeja con botes en la cabina de flujo/1-La botella debe estar en la cabina seleccionada/0-A1 no entrar el bote/0-entrar en la cabina de flujo/03-No se ha encendido la cabina de flujo/03-No se ha esterilizado por completo el aire	f142	0-S2 ha apagado la cabina antes de terminar el proceso.
f137	estimar	1-Instrumental esterilizado	Si en la línea ada, ve al finitro y faze un bote con un tubo recado.	f133-f131			
f138	entregachupo	1-Planta entregada al inventario	Toma el bote con la planta que se encuentra en la capsa de objetos de tu inventario haciendo doble click sobre ella, después llénala cerca de la cabina de flujo y déjala ahí.				
f139	solar/chop/cachuna	0-Planta en cabina	Toma la bandeja con botes para cortar la tallo y generar explantes. Observa su aspecto haciendo zoom.	f133-f131	0-No se ha encendido la cabina de flujo/03-No se ha esterilizado por completo el aire		
f140	contapilar	0-Planta cortada	Toca la bandeja para ajustarla a tu mano	f133-f137-f139	1-Los botes no contienen el medio de cultivo/03-No se ha esterilizado el instrumental/1-La planta debe estar en la cabina	f142	0-S2 ha apagado la cabina antes de terminar el proceso.
f141	agregar/cachuna	0-Bandeja ajustada para ir a crecimiento	Agrega la cabina y lleva la bandeja al Finitro y súbala para colocarla en una de las repisas.	f140	1-No se ha cortado la planta.		
f142	solar/bate/crecimiento	0-Bandeja en crecimiento	Sal de la cabina y vuelve a entrar, habed pasado una semana. observa de cerca tus plantas en crecimiento.	f140	1-No se ha cortado la planta en tallos con hojas		
f144	Moquetarefinibition	0-Base finitron bloqueada	Vuelve a entrar y salir las veces necesarias para que termine de crecer la planta. Cada vez que entres observará el crecimiento de una semana de crecimiento.	f143	1-Las plantas no se encuentran en una repisa del Finitron.		
f145	crecer chupo	1-Planta crecida	Puedes continuar con la siguiente fase de la práctica, ve a la sala principal y toca una PCR libre para encenderla.	f145	1-Existe error en la primera fase		
f146	finicrecimiento	1-Crecimiento terminado	Esgueta a que la PCR se caliente mientras los fludrenarios y el protocolo, luego accrete a la máquina de flujo y orden una bandeja	f145	1-Debes haber llenado la bandeja con botes a crecer sus plantas en el finitron		
f201	crecerdepqr	1-PCR encendido					
f202	estimar/bases	0-Fludrenario creado		f03	1-No hay todos los fludrenarios		
f203	redefinir protocolo	0-Protocolo creado		f04	1-No hay todos los protocolos		
f204	calentamiento/flujo	1-PCR caliente		f21	1-No se ha encendido la PCR		
f205	entregat/bandeja/d	1-Bandeja de hielo entregada al inventario	Toma la bandeja con hielo y déjala los reactivos de PCR del Agnético, al finalizar súbela la bandeja en tu poyeta para comenzar la mezcla.				
f206	unat/bandeja/d	0-Bandeja de hielo ajustada	Revisa la composición de la mezcla de PCR en el PROTOCOLO	f23	1-No se ha entregado la bandeja		
f207	bandeja	1-ACN en la fila del gen en bandeja		f206	1-No tienes la bandeja en las manos		
f208	bandeja/temperatura	1-Organismos en bandeja		f206	1-No tienes la bandeja en las manos		
f209	bandeja/temperatura	1-Organismos en bandeja		f206	1-No tienes la bandeja en las manos		
f210	bandeja/temperatura	1-Tiempo de la enzima polimerasa en bandeja		f206	1-No tienes la bandeja en las manos		
f211	bandeja/temperatura	1-Creando de Magister en bandeja		f206	1-No tienes la bandeja en las manos		
f212	bandeja/temperatura	1-Mezcla estándar de dNTP en bandeja		f206	1-No tienes la bandeja en las manos		
f213	bandeja/temperatura	1-ACN polimerasa en bandeja		f206	1-No tienes la bandeja en las manos		
f214	bandeja/temperatura	1-Agua destilada en bandeja		f206	1-No tienes la bandeja en las manos		
f215	bandeja/temperatura	1-Agua destilada en bandeja		f206	1-No tienes la bandeja en las manos		
f216	bandeja/temperatura	1-Tiempo de la enzima polimerasa en bandeja		f206	1-No tienes la bandeja en las manos		
f217	bandeja/temperatura	1-Plasmido 555 en bandeja		f206	1-No tienes la bandeja en las manos		
f218	bandeja/temperatura	1-Plasmido mRbNA en bandeja		f206	1-No tienes la bandeja en las manos		
f219	bandeja/temperatura	1-Antibiótico Kanamicina en bandeja		f206	1-No tienes la bandeja en las manos		
f220	bandeja/temperatura	1-Antibiótico Gentamicina en bandeja		f206	1-No tienes la bandeja en las manos		
f221	bandeja/temperatura	1-Antibiótico Cloranfenicol en bandeja		f206	1-No tienes la bandeja en las manos		
f222	bandeja/temperatura	1-Tiempo TE en bandeja		f206	1-No tienes la bandeja en las manos		
f223	bandeja/temperatura	1-Producto de PCR en bandeja		f206	1-No tienes la bandeja en las manos		
f224	agregata	1-Agueta en bandeja					
f225	solar/bandeja/ysalpinaravez	0-Bandeja en poyeta	Si en la línea ada, toma un bote de biber apagar/0-entrar en la cabina de flujo/03-No se ha encendido la cabina de flujo/03-No se ha esterilizado por completo el aire				
f226	solar/bandeja/ysalpinaravez	0-Bandeja en poyeta					
f227	entregat/bandeja/ysalpinaravez	1-Bote de biber apagar/0-entrar en la cabina de flujo/03-No se ha encendido la cabina de flujo/03-No se ha esterilizado por completo el aire	Ajusta el bote a tu mano y súbela el bote en tu poyeta.				
f228	solar/bandeja/ysalpinaravez	0-Bote en poyeta	Toca el bote para sacar un tubo a la bandeja de hielo para la pipeta.	f227	1-No se ha entregado el bote		
f229	sacarbog/ysalpinaravez	0-Tubo en gralla	Toca el soporte de pipetas para obtener una pipeta amarilla.	f228	1-No se encuentra el bote con tubos en la poyeta		
f230	entregat/pipeta/ysalpinaravez	1-Pipeta entregada al inventario	Ajusta la pipeta a tu mano y cargue una punta amarilla.	f230	1-No se ha entregado la pipeta		
f231	unat/pipeta/ysalpinaravez	0-Pipeta ajustada					
f232	crecepuntat/ysalpinaravez	0-Punta cargada	Toca la bandeja y selecciona los componentes uno por uno para el protocolo, cuando hayas terminado presiona "Finalizar mezcla".	f231	1-Debes tener la pipeta en la mano		

PROTOCOLO Y CONFIGURACIÓN VIRTUAL DE BIOTECNOLOGÍA

[illegible]

Identificador	Nombre y Acción	Mensaje Acción Validada	Mensaje de Tutoría	Dependencias	Mensajes de Error por Dependencias	Mensajes de Error por Orden de Dependencias	Incompatibilidades	Mensajes de Error por Incompatibilidades
F305	entregarbandejaalido	1-Bandeja de hielo entregada al inventario	Toma la bandeja con hielo y déjala las esquinas de la base de ligadura del frigorífico, al finalizar suelta la bandeja en tu poyata para comenzar la mezcla					
F306	añaditardesajado	0-Bandeja de hielo ajustada	Revisa la composición de la mezcla de ligadura en el PROTOCOLO	[F305]	1-No se la entregado la bandeja			
F307	tomandeparamétrico	1-ADN plasmado en bandeja		[F306]	1-No tienes la bandeja en las manos			
F308	tomandezapir	1-ADN amplificado por PCR en bandeja		[F306]	1-No tienes la bandeja en las manos			
F309	tomandepoliguia	1-Tampón de la enzima ligasa en bandeja		[F306]	1-No tienes la bandeja en las manos			
F310	tomandepresión	1-Agua destilada en bandeja		[F306]	1-No tienes la bandeja en las manos			
F311	tomandepesadibola	1-Agua destilada en bandeja		[F306]	1-No tienes la bandeja en las manos			
F312	tomandegundacozq	1-Oligonucleótidos izquierdo en bandeja		[F306]	1-No tienes la bandeja en las manos			
F313	tomandegundadecol	1-Oligonucleótidos derecho en bandeja		[F306]	1-No tienes la bandeja en las manos			
F314	tomandepropolimerasa	1-Tampón de la enzima polimerasa en bandeja		[F306]	1-No tienes la bandeja en las manos			
F315	tomandegcl	1-Correo de Magisterio en bandeja		[F306]	1-No tienes la bandeja en las manos			
F316	tomandup	1-Mezcla estándar de dNTP en bandeja		[F306]	1-No tienes la bandeja en las manos			
F317	tomandpolimerasa	1-ADN polimerasa en bandeja		[F306]	1-No tienes la bandeja en las manos			
F318	tomandepesadibola	1-Agua destilada en bandeja		[F306]	1-No tienes la bandeja en las manos			
F319	tomandepesadibola	1-Plasmido mBNA en bandeja		[F306]	1-No tienes la bandeja en las manos			
F320	tomandhibicodensamixtura	1-Antibiótico Kanamicina en bandeja		[F306]	1-No tienes la bandeja en las manos			
F321	tomandhibicodensamixtura	1-Antibiótico Gentamicina en bandeja		[F306]	1-No tienes la bandeja en las manos			
F322	tomandhibicodensamixtura	1-Antibiótico Cloranfenicol en bandeja		[F306]	1-No tienes la bandeja en las manos			
F323	tomandtempete	1-Tampón TE en bandeja		[F306]	1-No tienes la bandeja en las manos			
F324	agregatasa	1-Azúcar en bandeja		[F306]	1-No tienes la bandeja en las manos			
F325	tomandezapir	1-Mezcla ligasa en bandeja	Toca el bote de bote apretador que hay sobre la arena para sacar un tubo a la bandeja de la arena	[F306]	1-No tienes la bandeja en las manos			
F326	tomandezapir	1-Mezcla ligasa en bandeja						
F327	tomandezapir	1-Mezcla ligasa en bandeja						
F328	tomandezapir	1-Mezcla ligasa en bandeja	Toca el bote de bote apretador que hay sobre la arena para sacar un tubo a la bandeja de la arena	[F326]	1-No se encuentra el bote con bote en la poyata			
F329	tomandezapir	1-Mezcla ligasa en bandeja	Añade la pipeta a tu mano y carga una punta amarilla	[F327]	1-No has obtenido la pipeta del soporte			
F330	tomandezapir	1-Mezcla ligasa en bandeja		[F329]	1-Debes tener la pipeta en tu mano			
F331	tomandezapir	1-Mezcla ligasa en bandeja	Toca la bandeja y selecciona los componentes uno por uno de la bandeja de la arena, cuando hayas terminado presiona "Finalizar mezcla"	[F330]	1-Debes tener la pipeta en tu mano			
F332	tomandezapir	1-Mezcla ligasa en bandeja		[F330]	1-Debes tener la pipeta en tu mano			
F333	agregandeparamétrico	1-ADN plasmado agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No estás un tubo en la bandeja de la arena: 1-No has tomado el tubo con la mano: 1-Debes tener una punta en la pipeta amarilla en la mano: 1-Debes estar una punta en la pipeta amarilla en la mano: 1-Debes tener la pipeta en la poyata: 1-No has tomado el ADN plasmado de la arena			
F334	agregandezapir	1-Producto de PCR agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el producto de PCR plasmado de la arena			
F335	agregandepoliguia	1-Tampón de Ligasa (Gruasa) agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el tampón de ligasa plasmado de la arena			
F336	agregandepoliguia	1-Agua destilada agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el agua destilada plasmado de la arena			
F337	agregandepresión	1-Mezcla estándar de dNTP agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el agua destilada plasmado de la arena			
F338	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F339	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F340	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F341	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F342	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F343	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F344	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F345	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F346	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			
F347	agregandup	1-ADN polimerasa agregado a la mezcla, recuerda presionar "Finalizar mezcla" para terminar		[F332-F331]	1-La bandeja debe estar en la poyata: 1-No has tomado el ADN polimerasa de la arena			

PROTOCOLO Y CONFIGURACIÓN DE TUTORÍA DEL LABORATORIO VIRTUAL DE BIOTECNOLOGÍA

Identificador	Nombre Acción	Mensaje Acción Validada	Mensaje de Tutoría	Dependencias	Mensajes de Error por Dependencias	Mensajes de Error por Orden de Dependencias	Incompatibilidades	Mensajes de Error por Incompatibilidades
E348	agregar template	1- Targuon TE (Tris EDTA) agregado a la mezcla, recuadrar presionar "Finalizar mezcla" para terminar		[E343-E343]	1- La bandeja debe estar en la portada(1-No has tomado el tiempo TE de la reverse			
E349	agregar agua	1-Cloruro de Magnesio agregado a la mezcla, recuadrar presionar "Finalizar mezcla" para terminar		[E343-E343]	1- La bandeja debe estar en la portada(1-No has tomado el colorido de ensayo de la reverse			
E350	agregar una frambuesa	1-AAA, ácido indolacético agregado a la mezcla, recuadrar presionar "Finalizar mezcla" para terminar		[E343-E344]	1- La bandeja debe estar en la portada(1-No has tomado el ácido indolacético de la reverse			
E351	agregar un ácido ligación	1- Mezcla ligación agregado a la mezcla, recuadrar presionar "Finalizar mezcla" para terminar		[E343-E345]	1- La bandeja debe estar en la portada(1-No has tomado la mezcla ligación de la reverse			
E352	fin mezcla	0-Fin de la mezcla de la ligación	Señala la página para que regreses al siguiente	[E343-E344-E345-E346-E347]	0-No se ha agregado el ADN(plantando a la mezcla)0-No se ha agregado el producto de PCR a la mezcla(0-No se ha agregado el producto de PCR a la mezcla)0-No se ha agregado la enzima(0-No se ha agregado la enzima)0-No se ha agregado el agua destilada a la mezcla	[E343-E344-E345-E346-E347-E348-E349-E350-E351]		0-No se puede agregar Mezcla estabilizadora de dNTPs a la mezcla de la ligación(0-No se puede agregar ADN polimerasa a la mezcla de la ligación(0-No se puede agregar Cloruro de Magnesio a la mezcla de la ligación(0-No se puede agregar tampón de la enzima polimerasa a la mezcla de la ligación(0-No se puede agregar a la mezcla de la ligación(0-No se puede agregar Anticógeno de Calamita a la mezcla de la ligación(0-No se puede agregar Anticógeno de Calamita a la mezcla de la ligación(0-No se puede agregar Targuon TE (Tris EDTA) a la mezcla de la ligación(0-No se puede agregar Cloruro de Magnesio a la mezcla de la ligación(0-No se puede agregar Mezcla ligación a la mezcla de la ligación
E353	solimpye el reactante	1- Puesta en soporte	Toca la bandeja de tubo o pequeña para obtener el tubo con la mezcla	[E345-2]	1- No se ha terminado la mezcla			
E354	entregue tubo mezcla	1- Tubo con mezcla entregado al inventario	Añade el tubo a tu mano y toca el termobloque para calentar	[E345-3]	1- No se ha solido la página			
E355	abrir termobloque	0- Termobloque abierto	Siéntate el tubo de ligación	[E345-1]	1- El termobloque no se encuentra recalentado			
E356	solimmezcla	0- Tubo en termobloque	Toca la tapa del termobloque para cerrarla	[E345-5]	1- Tienes que cerrar el termobloque			
E357	cerar termobloque	0- Termobloque cerrado	Toca el termobloque para iniciar la ligación del fragmento	[E345-6]	1- No has finalizado el tubo en el termobloque			
E358	amplificar fragmento	1- Cierre del tiempo requerido de ligación (tiempo real de 10h) y puedes recoger tu muestra	Toca el termobloque para obtener el tubo con la mezcla	[E345-7]	1- El termobloque debe estar cerrado para poder iniciar la amplificación			
E359	entregue mezcla ligación	1- Tubo con fragmento ligado entregado	Limpia el tubo cerca de un frascillo, abre la puerta a -20 "C" y adhiere la muestra para guardar el tubo con la muestra de la ligación	[E345-8]	1- No se ha terminado de amplificar el fragmento			
E360	guardar ligación	0- Lado para guardar la ligación	Señala el tubo para que ingrese a la reverse	[E345-9]	1- No se ha entregado el fragmento amplificado			
E361	solimmezcla reverse	1- Tubo en reverse	Establece la enzima en este enlace http://go.glb/5J26	[E346-0]	1- No se ha seleccionado guardar mezcla de ligación de la reverse			