Manuel González de Rivera Fuentes
Maximiliano Paredes Velasco

Aprendizaje con programación Colaborativa

Número 2008-02

Serie de Informes Técnicos DLSI1-URJC
ISSN 1988-8074
Departamento de Lenguajes y Sistemas Informáticos I
Universidad Rey Juan Carlos

Índice

1 Motivación 5
2 Objetivos 8
3 Descripción del estado del arte 9
3.1 Sistemas *Group Work* de programación colaborativa 10
3.1.1 Discusión de los Sistemas *Group Work* 12
3.1.2 Resumen 14
3.2 Sistemas de aprendizaje con programación colaborativa 14
3.2.1 Discusión de los sistemas de aprendizaje colaborativo 24
3.2.2 Resumen 27
4 Reflexión 29
5 Conclusión 30
Referencias 30

Aprendizaje con Programación Colaborativa

Manuel González de Rivera Fuentes y Maximiliano Paredes Velasco

Universidad Rey Juan Carlos
Departamento de Lenguajes y Sistemas Informáticos I
maximiliano.paredes@urjc.es

Summary. El aprendizaje de la programación es una tarea compleja ya que intervienen diversos factores tanto a nivel conceptual como prácticos. Es especialmente significativo que la realización de trabajos prácticos, en los que el aprendiz se enfrenta ante el desarrollo de un programa, ayuda considerablemente al aprendizaje de la programación. Nosotros pensamos que el desarrollo de estas actividades prácticas realizadas en grupo, dirigidas por estrategias de aprendizaje colaborativo, puede mejorar el proceso de aprendizaje. En este trabajo realizamos un estudio de herramientas informáticas para aprender a programar y presentamos una serie de aplicaciones para aprender la disciplina de la programación mediante estrategias de aprendizaje colaborativo, con el objetivo de acercar las nuevas tecnologías a las aulas y facilitar la enseñanza y aprendizaje a profesores y estudiantes.

1 Motivación

El aprendizaje colaborativo de lenguajes de programación se entiende como una serie de metodologías de enseñanza y entrenamiento, además de la ayuda de las tecnologías disponibles, donde cada participante del grupo del que forma tiene la responsabilidad tanto de su propio aprendizaje como el de sus compañeros, creando así una especie de relación de simbiosis en el que todos los participantes dependen de los demás, además de sí mismos. La enseñanza de programación en entornos colaborativos quiere facilitar ambientes donde se pueda producir el diálogo entre los estudiantes, con el objetivo de que puedan resolver entre ellos todos los problemas que puedan surgir; la meta a conseguir es que el intercambio de situaciones e interacciones sociales pueda facilitar tanto el aprendizaje individual como el aprendizaje a nivel de grupo.

En la pasada década hemos podido observar el nacimiento de gran cantidad de tecnologías basadas en la Web que permiten el uso de maneras tradicionales y no tradicionales para el aprendizaje colaborativo. Con la llegada de Internet (su difusión universal, su sencilla utilización y sus capacidades) se ha descubierto una serie de nuevas posibilidades para la formación. Internet no se ha generalizado por una simple cuestión de moda sino como una poderosa herramienta que nos ayudará en el continuo aprendizaje para ampliar nuestra formación personal. Estos sistemas colaborativos permiten que los estudiantes tengan la oportunidad de interactuar con otros estudiantes y compartir nuevas ideas. Además pueden ser implantados en aulas

con la ayuda de un instructor que es el encargado de gestionar y guiar la colaboración. En estos ambientes el objetivo es alcanzar las metas propuestas al grupo de trabajo [Soller, 2005].

Sin embargo es muy temprano para poder implantar un soporte online en estos sistemas de manera generalizada en las aulas tradicionales, ya que a día de hoy hay muchas opiniones que discrepan sobre la cuestión de que trabajar online y fuera de las aulas sea más efectivo que el modelo tradicional de asistencia presencial. El aprendizaje colaborativo no pretende sustituir la docencia tradicional, sino que quiere que sea un poderoso complemento extra para el aprendizaje de un lenguaje de programación y una solución válida para aquellas personas que no pueden asistir de manera presencial a las clases docentes.

Por experiencia general, los comienzos en el aprendizaje de la programación siempre ha sido un proceso complicado, ya que es una disciplina totalmente diferente a lo que se ha visto desde entonces por parte del programador novel. La programación exige cambiar de manera radical el modo de pensar y analizar las cosas. Aun habiendo adquirido los conocimientos teóricos necesarios, se ha detectado una gran dificultad de aplicar esos conocimientos teóricos en la resolución de problemas prácticos. Para ser un buen programador, éste ha de adquirir conocimientos sobre la sintaxis del lenguaje de programación a aprender, debe conocer sus conceptos abstractos y esta dificultad se traduce en carencias de motivación para programar [Esteves, 2004].

Generalmente, los estudiantes suelen aprender a programar a través de cursos que dependen de conocimientos teóricos y de la memorización de éstos, pero la programación básica necesita de un mayor enfoque práctico, basado principalmente en resolver actividades de programación [Esteves, 2006].

Actualmente hay una gran demanda en el aprendizaje lenguajes de programación. Gran parte de las técnicas de enseñanza fueron creadas para estudiantes con un perfil determinado, por eso a lo largo de los años se ha buscado el diseño de un lenguaje de programación que sea lo más sencillo posible, pero las metodologías utilizadas hacen que todavía sean aptas para una parte determinada de la población. Sin embargo, a pesar de que se busque una manera diferente de programar, que permita ser cercano a personas cuyo perfil no es el idóneo para la creación de *Software*, los lenguajes de programación visual han supuesto un gran avance para alcanzar este objetivo. Con el afán de buscar las causas de las dificultades de los estudiantes en aprender programación, Kölling [Kölling, 2003] junto a sus compañeros Quig, Patterson y Rosenberg, llegaron a la conclusión, en el caso del aprendizaje de lenguajes de programación orientados a objetos, que éstos no son más complicados de aprender que los estructurados. La causa es que las herramientas utilizadas no son las más apropiadas. Pero las principales causas de este problema las resume en tres: a) los entornos de programación no son orientados a objetos, ya que la interacción del usuario no es a través de objetos sino por líneas de código fuente, b) los entornos son demasiado complejos, normalmente los entornos de programación están adaptados para programadores profesionales, lo que hace que la interfaz esté demasiada cargada

para programadores noveles, c) los entornos centran principalmente al usuario en la creación de interfaces. Al utilizar gráficos los entornos, los estudiantes no pueden adquirir conceptos de orientación de objetos adecuados. Muy pocos entornos muestran la estructura de clases que tienen los gráficos. Los estudiantes no poseen herramientas en estos entornos que ayuden a mejorar su estilo de programación, únicamente les señala los errores puntuales que puedan cometer durante el proceso de escritura de código.

El aprendizaje colaborativo es una de las principales motivaciones en la investigación de nuevas formas de enseñar. Se pretende buscar una serie de estrategias con el objetivo de que la interacción entre un grupo de estudiantes sea óptima, que cada componente del grupo aprenda y a su vez ayude al que no lo haga. Además de que es beneficioso para el estudiante ya que aprende a trabajar en equipo, es un concepto muy utilizado de manera satisfactoria en el mundo empresarial. En el aprendizaje de lenguajes de programación de manera colaborativa los resultados no son el objetivo principal, sino lo importante es la experiencia adquirida por los estudiantes [Newman, 1989].

Este modo de aprendizaje está tan aceptado que hasta los propios implicados, es decir, los estudiantes, lo apoyan de manera mayoritaria. En un estudio realizado por Williams y Kessler [Williams, 2000], de las universidades de Carolina del Norte y Utah respectivamente, hicieron un estudio donde había un grupo de estudiantes que trabajaban de una manera individual y otro que trabajaba en parejas donde tenían que resolver ciertos ejercicios de programación, llegando a la conclusión que el 84% de los estudiantes prefería programar en pareja, además un 96% de ellos confesaba que disfrutaban trabajando con otro.

En una aplicación tanto orientada al aprendizaje como orientada al desarrollo colaborativo deben estar presentes una serie de características como son: las actividades comunes, el entorno compartido y el espacio/tiempo. Llamamos actividades comunes a aquellas tareas comunes que los participantes del grupo llevan a cabo; el entorno compartido nos da la posibilidad de tener informado a cada miembro del proyecto sobre el estado de éste, lo que cada miembro está trabajando, etc.; y el espacio/tiempo soporta que la interacción del grupo de trabajo se produzca en el mismo lugar y momento [Hsu, 1993]. En cuanto a la interacción la podemos encontrad de dos tipos: síncrona o asíncrona, que a su vez puede ser distribuida o centralizada.

En este trabajo pretendemos mostrar que el aprendizaje de programación de manera colaborativa es una alternativa real al método tradicional de asistencia presencial a las aulas. Para ello mostraremos las soluciones que hay disponibles en el mercado que facilitan el aprendizaje colaborativo de la programación, destacando los aportes principales que pueden dar a los estudiantes en la enriquecedora tarea del aprendizaje de un lenguaje de programación. A su vez analizaremos las diferentes soluciones que hay actualmente para programadores avanzados en cuanto a programación colaborativa.

2 Objetivos

En este apartado se pretende mostrar los objetivos que se van a desarrollar en este trabajo visto el planteamiento que hemos planteado previamente. Existen dos ejes relacionados por los que nos movemos en este trabajo de investigación: la programación colaborativa orientada al desarrollo y el aprendizaje de programación colaborativamente. A continuación citamos los objetivos de manera específica:

- *Mostrar la situación actual que encontramos en el mundo de la programación colaborativa y su aplicación en el aprendizaje*. En este apartado queremos mostrar el estado del arte en esta materia, donde analizaremos las principales aportaciones de cada uno de los sistemas más importantes que hay actualmente. Se describirá su modo de funcionamiento y arquitectura (de una manera resumida). Este análisis de sistemas irá dividido a su vez en dos apartados: sistemas de programación colaborativa orientadas al desarrollo y sistemas que ayudan en el aprendizaje lenguajes de programación de manera colaborativa, haciendo especial hincapié en este último.

- *Comparativa de todos los sistemas analizados*. Pare ello incluiremos una tabla comparativa de todos los sistemas analizados anteriormente, donde se especificará su nombre, tipo, metodología y una breve descripción de la principal funcionalidad del sistema. En este caso los sistemas estarán clasificados en dos grupos (programación colaborativa y aprendizaje con programación colaborativa).

- *Crítica del estado del arte o discusión*. En este caso, se criticará objetivamente las deficiencias que encontramos en cada uno de los sistemas analizados, sin embargo se hará especial hincapié a las principales deficiencias que se encuentran en todos los trabajos en general, es decir, funcionalidades o tecnologías que se echan en falta en todos los sistemas.

- *Conclusiones*. Se describirán las conclusiones más importantes, en apoyo a los resultados obtenidos del análisis realizado en apartados anteriores.

3 Descripción del estado del arte

Para escribir este apartado, inicialmente tendríamos que preguntarnos qué necesidades tienen los maestros para transmitir los conocimientos sobre programación de una manera clara, que sea captada fácilmente por los estudiantes. En un estudio realizado por la Universidad del País Vasco [Ezeiza, 2007] se les hizo una encuesta a los propios profesores, donde tenían que responder a 50 preguntas y cuyas respuestas tenían las siguientes opciones: 1) actividades en pequeños grupos, 2) gestión del significado a través de la conversación y 3) colaboración en tareas. En ella un 71% de las respuestas marcadas en la opción correspondiente eran para la primera opción, un 54,1% para la segunda y un 58,4% para la última. Este encuesta estaba encaminada para realizar posteriormente un sistema de aprendizaje colaborativo basado en Moodle llamado COVCELL. Por lo que en base a estas necesidades se puede adaptar el sistema para satisfacer a éstas, llegando a la conclusión de que las necesidades de los profesores para tener un sistema de aprendizaje colaborativo es que tenga buenas herramientas de comunicación ya sean síncronas o asíncronas, un repositorio común para la realización de actividades y un entorno común para trabajar en grupo.

Otra rama a la que se está moviendo la programación colaborativa orientada al aprendizaje es mediante la realización de actividades en entornos 3D. Según un estudio realizado en la Universidad de Sevilla [Antunes, 2006], estos entornos están adquiriendo una gran popularidad en los cuales la gente interactúa en tiempo real dentro de mundos virtuales, lo que hace interesante un área de investigación y desarrollo. El ejemplo más representativos de estos entornos es el famoso Second Life, que permite a los profesores desarrollar libremente sus propios materiales, proveyendo de sus propios contenidos si es necesario. Universidades como la de Harvard utilizan Second Life como complemento en la enseñanza de programación donde se suben los videos de las clases y materiales docentes, además de encontrarte con alumnos de clase o hacer tutorías virtuales con el profesor.

En el apartado de programación colaborativa, nos encontramos en la situación de que lo ideal es que el sistema tenga sistemas de trabajo en grupo síncronos y asíncronos [Magnusson, 1993], ya que en el desarrollo y en el mantenimiento de software es necesario que haya comunicación asíncrona en caso de que el participante se encuentre trabajando solo en ese momento, la herramienta síncrona sería ideal para cuando haya varias personas trabajando en el mismo proyecto al mismo tiempo. En otro estudio [Posner, 1993] que recoge las valoraciones de programadores, concluye que la programación conjunta On-line y Off-line es la ideal de aplicar en un sistema que los soportase, además de ser un sistema flexible para que el transito entre síncrono y asíncrono se realice sin restricciones. Para que se puedan cumplir estas dos modalidades de comunicación, es necesario la presencia de Internet en el sistema para que funcione.

En base a estas necesidades solicitadas por profesores y programadores, vamos a analizar las aplicaciones que más se ajustan a estas necesidades y que hay actualmente. En primer lugar analizaremos las aplicaciones que soportan programación colaborativa que ayuden en el trabajo a los programadores. En segundo

lugar, haremos un análisis mas extendido sobre las aplicaciones disponibles de programación colaborativa dirigidas para el aprendizaje, con el fin de ayudar a la labor educativa de los profesores y que enriquezca el aprendizaje de lenguajes de programación a los alumnos.

3.1 Sistemas *Group Work* de programación colaborativa

Coven

Este proyecto europeo tiene como objetivo facilitar un entorno computacional para teletrabajo y presencia virtual a modo de conseguir las facilidades necesarias para soportar sistemas de trabajo cooperativo, además de demostrar el valor adicional que puede demostrar la realidad virtual a los usuarios del sistema. Coven [Chu-Carroll, 2000] dispone de una plataforma de ambiente virtual cooperativo con los servicios necesarios para el soporte al teletrabajo en sistemas de ambiente virtual distribuidos.

RECIPE

Realizado en la Universidad de Griffith en Australia en el año 2000 [Shen, 2000]. Es un sistema que permite a programadores distribuidos geográficamente participar de manera conjunta en el diseño, codificación, prueba, depuración y documentación de un mismo programa o proyecto. Shen propone un sistema para la programación colaborativa en tiempo real basado en Internet. En realidad el sistema es un middleware que permite enviar datos de entrada desde varios puestos a un software en modo texto convencional y distribuir la salida a todos esos puestos para que todos los usuarios puedan ver las acciones de todos.

La arquitectura del sistema está centralizada en una máquina que contendrá los archivos del proyecto que estamos desarrollando y las herramientas de compilación (por ejemplo javac), depuración (gdb) y un shell de UNIX para ejecución y prueba del software desarrollado; además el sistema dispone de un editor colaborativo denominado REDUCE que permite crear y modificar archivos de código fuente entre varios desarrolladores. El servidor de RECIPE permite la gestión de sesiones en las que un desarrollador comienza una nueva tarea con una de las herramientas disponibles en el sistema y a la que se pueden unir otros usuarios bajo determinados permisos. Por otro lado los “sitios” RECIPE envían las entradas del usuario que está utilizando ese “sitio” al servidor y reciben la salida que es distribuida, cuando se produce cualquier cambio, a todos los “sitios” RECIPE que están en esa sesión.

Phprojekt

Su primera versión nació en el año 2002, pero hasta el día de hoy sigue en continua mejora. Phprojekt [Phprojekt] es una aplicación modular de licencia GNU para la coordinación de actividades en grupo e individuales, y permite además compartir información y documentos vía Intranet e Internet. Es una aplicación Web con licencia GPL de trabajo en grupo implementada en PHP para facilitar su uso.

Está diseñado para albergar todos los proyectos de software que se realicen en una organización en un servidor común dentro de una Intranet o incluso entre organizaciones a través de Internet.

Existe un sistema de Gestión de privilegios para que en caso de que haya múltiples proyectos o subproyectos. El usuario que quiera acceder al proyecto, éste ha de ser perteneciente a éste. Tiene una herramienta donde se ven las tareas que se van cumpliendo y, por lo tanto, se ve el estado en que se encuentra el proyecto. Para la comunicación, el sistema tiene un sistema de Chat para la comunicación síncrona y un foro para cada proyecto y un cliente de e-mail integrado en caso de la comunicación sea asíncrona. También posee un sistema de votaciones donde los componentes votan para la toma de decisiones. La columna vertebral de este sistema es la Gestión de archivos, ya que los usuarios pueden compartir archivos en un espacio común, con posibilidad de imponer restricciones de acceso al archivo, también se pueden organizar estos archivos por categorías.

WubHub

Desarrollado por Adam Cheyer y Joshua Levy en 2006 [Cheyer, 2006]. Es una herramienta Web colaborativa que integra una Wiki, un portal Web y un entorno de programación. Los comandos usados en WubHub permiten desarrollar tareas desde la Web, tales como búsquedas, extracción y combinación de información de las páginas Web.

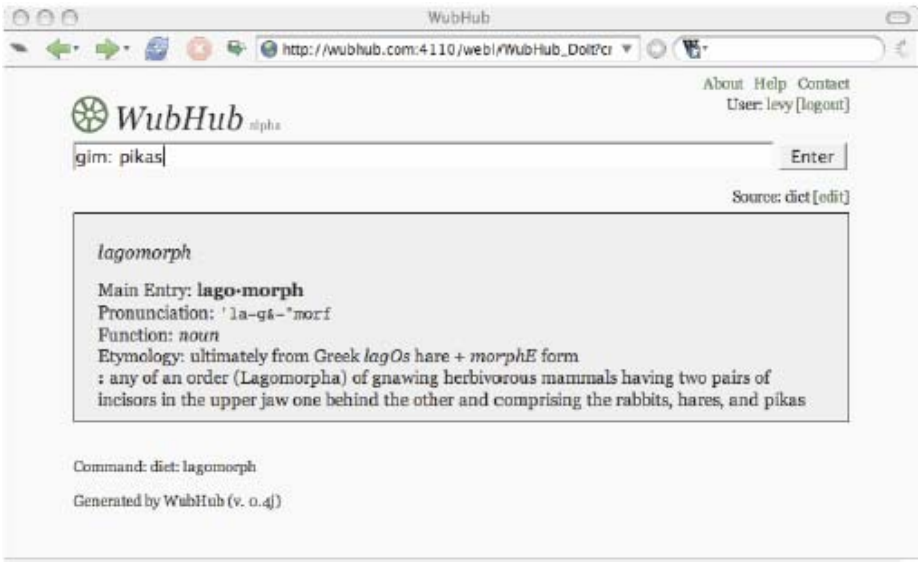

Figura 1: Resultado de búsqueda de comando

El usuario entra a un portal Web donde en su interfaz se encontrará una línea de comandos y un marco donde se mostrará el contenido. Las entradas son aceptadas

como argumentos donde se retornará un valor, el cual será mostrado en el marco. Estos comandos creados podrán ser compartidos en la Wiki para que los demás los conozcan y puedan utilizarlos. El usuario introduce un comando en un buscador, éste le dará el resultado con un ejemplo de aplicación del comando, además de la posibilidad de modificarlo. La arquitectura de WubHub se divide en tres partes principales: un mecanismo de almacenamiento, un entorno de ejecución y un frontal. Estas tres partes están integradas en un único servidor, la interfaz Web de usuario funciona gracias a una aplicación web en el servidor

Croquet System

Croquet [Breeding, 2005] es una nueva plataforma de desarrollo de código abierto iniciado en el año 2001 para crear aplicaciones en línea multiusuario y altamente colaborativas. Desarrolla una arquitectura en red que soporta comunicación, colaboración, compartimiento de recursos y computación asíncrona entre múltiples usuarios. Usando Croquet los desarrolladores de software pueden crear poderosas y altamente colaborativas aplicaciones 2D y 3D. Detrás de Open Croquet se encuentran especialistas como Alan Kay, Julián Lombardi, Mark McCahill, Andreas Raab, David P. Reed y David A. Smith.

Croquet es una combinación de software de computador y arquitectura de red que soporta colaboración entre múltiples usuarios dentro del contexto de un sistema de información distribuida a larga escala. Derivado de Squeak, que ofrece una arquitectura de red p2p lo cual posibilita la comunicación, colaboración, compartición de recursos, y computación síncrona entre múltiples usuarios en múltiples dispositivos. Cada parte del sistema está diseñada con la idea de habilitar en "tiempo real", interacciones idénticas entre grupos de usuarios. La arquitectura de este sistema está diseñada para facilitar el desarrollo de aplicaciones colaborativas sin tener que dedicar mucho esfuerzo y experiencia en cómo funcionan las aplicaciones replicadas.

3.1.1 Discusión de los Sistemas *Group Work*

El sistema Coven está dirigido para negocio y para masas, además de que en un principio es un proyecto a nivel europeo que ofrecía grandes expectativas, sin embargo, desde el año 2000 no se ha vuelto a publicar nuevas versiones, por lo que podemos considerar que el proyecto está abandonado. El sistema RECIPE es un gran proyecto con vistas a este campo ya que nos permite trabajar con otros programadores que se encuentren ubicados en otros puntos del planeta. Con este sistema se puede programar de una manera eficaz y colaborativa gracias al editor común. El sistema de asignación de permisos a cada usuario del sistema es una utilidad muy favorable para la coordinación, con esto nos permitirá que en ciertos momentos del proyecto asignemos los permisos a cada usuario según las necesidades del proyecto. Otro punto a destacar de RECIPE es que tenemos la posibilidad de utilizarlo con otros compiladores, esta flexibilidad nos permitirá utilizar nuestro compilador habitual en caso de que el compilador de RECIPE no nos agrade. Además que este sistema es totalmente compatible para el aprendizaje de lenguajes de programación debido a que

posee herramientas de uso común para desarrollar una actividad de manera conjunta. Sin embargo, tiene como desventajas el no poder almacenar el trabajo que va haciendo el programador para obtener conclusiones posteriormente, se hecha en falta un historial del trabajo realizado, además de no poseer herramientas especializadas de comunicación y conocimiento.

El Phprojekt ofrece una aportación novedosa llamada TimeCard. Esta utilidad nos permite registrar el trabajo que va realizando cada miembro del proyecto y permite registrar el tiempo empleado por cada programador en realizar la tarea, todo esto con fines estadísticos, lo que nos permite hacer un seguimiento de lo que hace cada miembro cada día. Tiene soporte para varios idiomas, módulos que facilitan la ampliación del sistema, un calendario común para que todo el mundo esté informado sobre futuros eventos o reuniones, comunicación síncrona y asíncrona. Es un sistema que permite la movilidad gracias a su posibilidad de acceder mediante Wap y PDA. Sin embargo una gran desventaja es la ausencia de herramientas específicas para el tratamiento del código fuente o de productos software, debido a que está planteado para proyectos en general y no únicamente para proyecto de software.

Del sistema basado en Web Wub Hub cabe destacar la posibilidad de que todo el mundo puede introducir nuevos comandos, además de tener una interfaz muy simple, lo que hace que su navegación sea rápida para utilizar sus servicios. Aspectos en contra a destacar es que la potencia del depurador es muy limitada, la dudosa fiabilidad que puedan tener los nuevos comandos subidos por los usuarios a no ser que haya una exigente moderación a la hora de incluir nuevos comandos, más difícil todavía es la moderación de las modificaciones, privilegio que también tienen activo los usuarios. Aun está en versión prototipo.

Del sistema Croquet tiene como principal ventaja que es multiplataforma, lo que permite que pueda ser instalado en los sistemas operativos más conocidos como Windows, Linux o Mac. Tiene una arquitectura escalable lo que hace que facilite la ampliación de la complejidad de la aplicación que se está realizando, es ideal para realizar simulaciones y visualizaciones colaborativas 3D. Como limitación puede ser la utilización del Lenguaje Smalltalk dentro de Squeak que puede hacer complicada la adaptación.

Como podemos ver en este apartado del análisis hemos analizado sistemas y proyectos donde cada uno tiene un forma de funcionamiento muy diferenciada respecto a los demás. De esta manera hemos podido resaltar sus principales cualidades y sus carencias, con el objetivo de tener mayor facilidad en encontrar una problemática general. Sin embargo, la problemática general analizada sería en base a estos sistemas analizados, lo que no tendríamos el número de fuentes suficientes para obtener unas conclusiones generales y convincentes, por lo que la problemática general la plantearemos una vez analizados todos los sistemas de aprendizaje de programación colaborativo, tema que plantearemos en los siguientes apartados (especialmente en “Reflexión”).

3.1.2 Resumen

Nombre	asíncrono/ síncrono	Metodología	Breve descripción
Coven	Síncrono	Colaboración	Plataforma de ambiente virtual cooperativo con los servicios necesarios para el soporte al teletrabajo en sistemas de ambiente virtual distribuidos.
RECIPE	Síncrono	Colaboración	Es un sistema que permite a programadores distribuidos geográficamente participar de manera conjunta en el diseño, codificación, prueba, depuración y documentación de un mismo proyecto.
Phprojekt	Asíncrono y síncrono	Discusión Colaboración	Es una aplicación modular para la coordinación de actividades en grupo, permite además compartir información y documentos vía Intranet e Internet.
WubHub	Asíncrono	Colaboración Información	Es una herramienta Web colaborativa que integra una Wiki, un portal Web y un entorno de programación.
Croquet system	Síncrono	Colaboración	Es una nueva plataforma de desarrollo de código abierto para crear aplicaciones en línea multiusuario y altamente colaborativas.

3.2 Sistemas de aprendizaje con programación colaborativa

AlgoArena

Creado por Hideyuki Suzuki and Hiroshi Kato [Suzuki, 2002] en 1995. Es un sistema enfocado a estudiantes noveles de programación, que consiste en ambientarlo en un combate de Sumo en el que cada participante tiene un luchador, con dicho luchador combate con otros programadores que a su vez tienen a otro luchador, previamente los usuarios han de estar registrados en el sistema. El lenguaje utilizado para programar es LOGO, ideal para aprender a programar junto a Pascal. Se van proponiendo ejercicios de programación durante el combate y los combatientes proponen soluciones; el propósito de esta aplicación es que los usuarios compitan entre ellos para poner a prueba sus habilidades en programación. Se pretende

también formar una comunidad donde discutir sus problemas e intereses en programación. En un experimento realizado en una clase, se asignó un luchador por cada dos estudiantes y se consiguió como objetivo que se trabajase en equipo ya que iban discutiendo y dando sus puntos de vista durante el transcurso del ejercicio o combate, además de ampliar el espíritu de competitividad entre ellos.

College

Realizado por el grupo de investigación CHICO [Bravo, 2004] (Universidad de Castilla-Mancha) en 2004. Es un sistema dirigido al aprendizaje de JAVA que soporta la edición, compilación y ejecución de programas de manera colaborativa por parte de los alumnos, y la revisión del trabajo efectuado por los estudiantes por parte del profesor con el objetivo de extraer conclusiones sobre el proceso de programación en grupo, facilitando a su vez la tarea evaluadora del profesor.

Para su funcionamiento, el alumno o profesor inicia la sesión en un servidor. Cada sesión se define mediante un nombre, un tipo, un fichero y un horario en el que se puede realizar. Dicho fichero contiene el ejercicio a resolver por los alumnos. Una vez iniciada la sesión el participante tendrá la posibilidad de consultar los participantes que se encuentran conectados en ese instante. Al compilar código los alumnos discuten los errores a través de un Chat, la edición de código se realiza por turnos para mayor coordinación. El sistema está desarrollado en Java, y opera siguiendo un modelo cliente/servidor sobre redes TCP/IP para facilitar su utilización sobre Internet/intranet. La parte de sincronización colaborativa está basado en mecanismos del sistema DomoSim-TPC, un entorno de aprendizaje colaborativo enfocado a la enseñanza de la robótica.

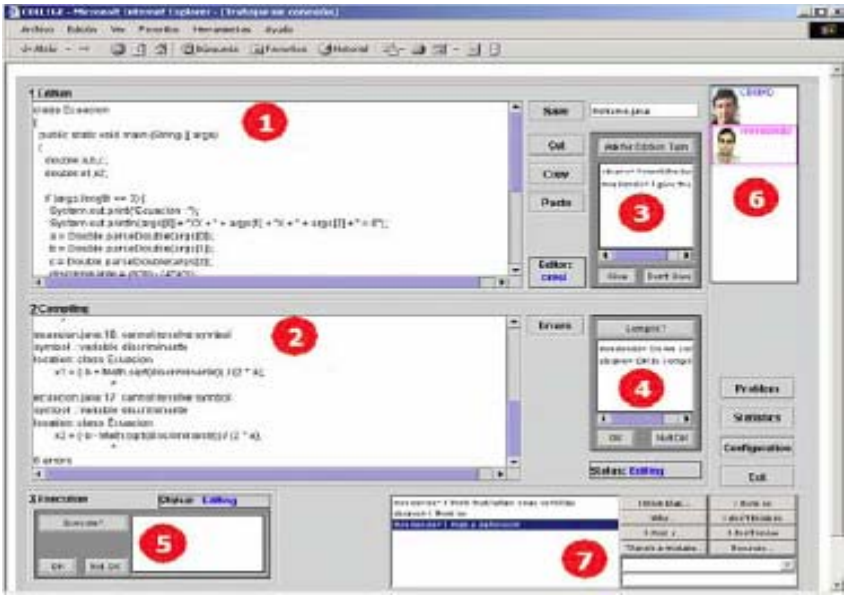

Figura 2: Un sesión de Programación Colaborativa en COLLEGE.

OOP-Anim

Este sistema está creado por los autores son Micaela Esteves de la Escola Superior de Tecnologia Gestão de Leiria y António Mendes del Centro de Informática e Sistemas de la Universidade de Coimbra, ambos de Portugal, en el año 2004 [Esteves, 2003] [Esteves, 2004]. Es un entorno de aprendizaje colaborativo escrito en JAVA que ayuda a los estudiantes a aprender nociones de programación orientada en objetos, concretamente el lenguaje JAVA. Permite simular y animar la ejecución de programas hechos con este leguaje de programación. Según va escribiendo código el alumno, cada referencia u objeto queda reflejado en la animación. El modo de docencia es que el profesor propone una ejercicio y un alumno o varios han de resolverlo. La interfaz gráfica está formada por una parte donde está el código en JAVA, en otro lugar están las animaciones y en la zona restante se encuentra la información sobre clases, objetos y referencias.

HabiPro

Realizado en 2002 por Aurora Vizcaíno [Vizcaíno, 2000] [Vizcaíno, 2007]. HabiPro es un sistema síncrono, cliente/servidor y distribuido que pretende desarrollar en los alumnos buenos hábitos de programación. La pantalla está formada por una parte donde viene el enunciado del problema, otra con la parte de código a escribir y una ventana de Chat con sus usuarios que están conectados en ese momento. Los alumnos usan la ventana del chat para discutir qué solución de las propuestas (no es obligatorio que todos los alumnos propongan una solución) consideran la correcta. Cuando el grupo llega a un acuerdo, todos los alumnos deben presionar el botón "check" que hay al lado de la ventana que contiene la solución que el grupo considera correcta. El sistema no evaluará la solución, indicando si es o no correcta, hasta que todos los alumnos pulsen el botón "check". De esta forma se obliga a que los alumnos tengan que tomar una decisión y decantarse sólo por una solución.

HabiPro está formado por una interfaz, la misma para todos los estudiantes, sean o no simulados, de hecho tiene un gestor de información que analiza las entradas recibidas y las distribuye entre los Modelos de Estudiantes (ME), el Modelo del Grupo (MG) y/o el modelo del estudiante simulado (en inglés Simulated Student Behaviour Model, SSBM).

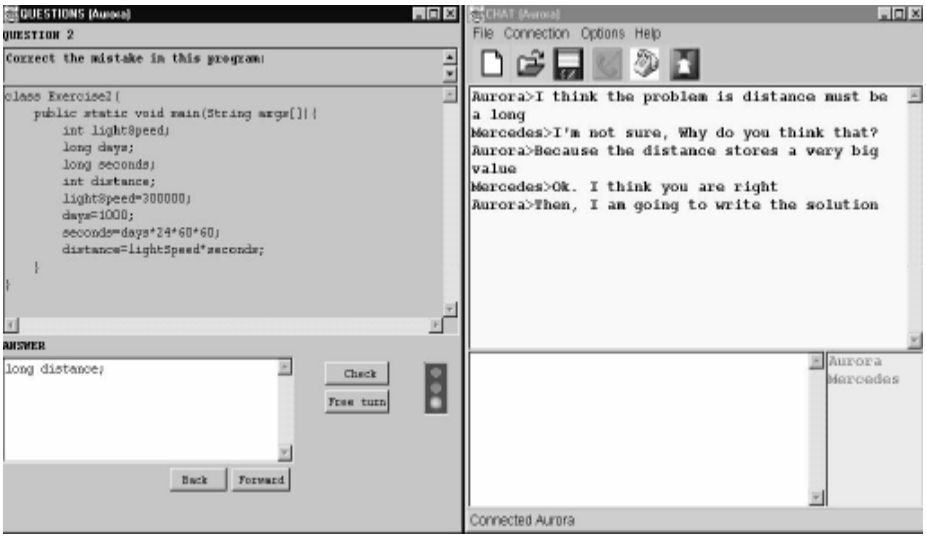

Figura 3: Interfaz de la aplicación HabiPro

Una novedad que diferencia este sistema es la presencia del citado estudiante simulado. El estudiante simulado ayuda en la participación de los demás componentes del grupo y que éstos se centren en el tema. Éste tiene una base de datos para saber de qué están hablando, donde actuaría en caso de que haya dos frases consecutivas que no aparece en su base de datos, como por ejemplo que hubiera una conversación OFF-topic. Un caso de ejemplo sería que los estudiantes están hablando de fútbol, en este caso el estudiante simulado contesta con "A mí no me gusta el fútbol, vamos a seguir con los ejercicios". El estudiante simulado sugiere que se continúe con los ejercicios. Otra función importante del estudiante simulado es que si se han propuesto dos soluciones incorrectas, el estudiante simulado propone una posible solución a la real.

ECPASCAL

Realizada en 2003 realizada por Villegas H., Atramiz I., Delgado M., Dorta de la área científica venezolana. ECPASCAL [Atramiz, 2003] [Villegas, 2003] es un entorno de programación colaborativo con soporte para el aprendizaje. A través de esta aplicación los profesores pueden editar, compilar, desarrollar y corregir los errores realizados en un programa escrito en PASCAL para compartir estas tareas con los estudiantes. Como consecuencia, cada estudiante podría observar las acciones dadas por el profesor desde un ordenador.

Varios usuarios inician una sesión, pero solamente uno de ellos es el que la administra, los usuarios son identificados por un color. El usuario administrador será quien permita que otro usuario pueda modificar el código fuente. Las modificaciones se reflejan gracias a que el texto modificado queda coloreado con el color que identifica al usuario. Tiene también una barra de tareas donde se ve qué usuario está haciendo tareas de programación. Permite compartir los programas realizados por cada uno de los usuarios para unificarlos. Para que la tarea de debugging sea

colaborativa, el sistema tiene una ventana de Chat para que cada usuario comunique cualquier cosa. Gracias a la herramienta de Chat los estudiantes se dan cuenta de los errores que cometen y comparten ideas para corregirlos.

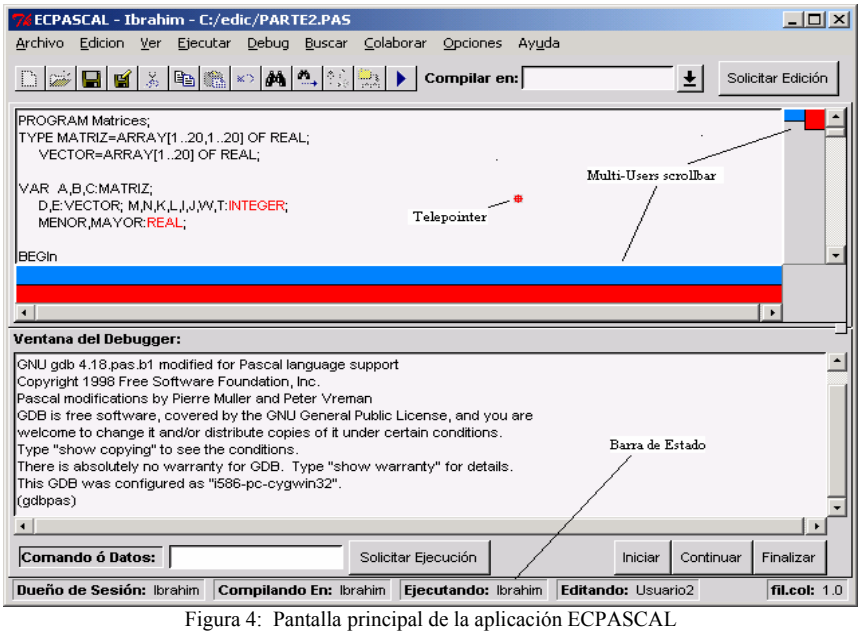

Figura 4: Pantalla principal de la aplicación ECPASCAL

Jeco

Realizado por Andrés Moreno, Niko Myller, y Erkki Satinen de la Universidad de Joensuu en Finlandia en 2004. Jeco [Moreno, 2004] es un entorno de aprendizaje de lenguajes de programación colaborativo a través de la discusión de documentos y fragmentos de código. Este sistema integra a su vez otros dos sistemas: Jeliot 3 y Woven Stories. El primero anima la ejecución del programa mostrando el modo en que las expresiones son evaluadas y cómo son creados los objetos. Woven Stories permite visualizar documentos dentro de la aplicación, de esta manera podemos visualizar documentos, ver código, animaciones y Chat en una misma pantalla. El servidor Woven Stories alberga todos los documentos mediante una base de datos común.

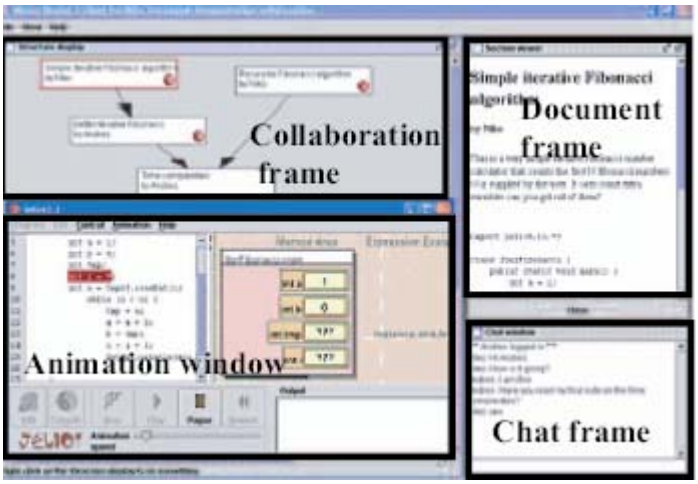

Figura 5: Interfaz de usuario de Jeco

El funcionamiento consiste en que los usuarios se conectan al servidor de Woven Stories a través de un programa cliente. La interfaz tiene una parte donde se visualiza el documento, otra ventana de Chat para discutir y otra parte donde se visualiza el código que ha subido un usuario. El cliente Jeco al estar conectado con el servidor Woven Stories a través de Internet nos permite trabajar fuera del aula.

DPE

Realizado por Chang-Hyun Jo, Allen J. Arnold en 2003 en un proyecto conjunto entre la Universidad de Dakota y Fullerton [Chang-Hyun, 1999] [Chang-Hyun, 2003]. Es un entorno de programación colaborativo pero con aplicación para ser usado por profesores y alumnos. Los participantes se mantienen comunicados en tiempo real gracias a Internet. Los programadores pueden ejecutar sus programas de manera remota, además de que los managers, testeadores o profesores pueden compilarlos desde el lugar donde se encuentren, pueden hacer todo tipo de acciones (editar, compilar y ejecutar) conjuntas desde una misma ventana gracias a Internet.

Está realizado en JAVA. DPE es un programa basado en Internet y en modelo de Cliente/Servidor. Además utiliza CORBA IIOP para las comunicaciones que no son multimedia y RTP para el sonido y el video en tiempo real. Aunque aparentemente se crea que es estrictamente un entorno de programación, también es utilizado como entorno de aprendizaje de programación colaborativo. Es muy parecido al sistema ECPASCAL solo que más avanzado y con más posibilidades de comunicación.

AnimPascal

Realizado por Satratzemi en 2001. AnimPascal [Satratzemi, 2001], es un ejemplo de entorno de aprendizaje de programación. El propósito de AnimPascal es ayudar a los estudiantes a comprender las fases de desarrollo, verificación, depuración y ejecución de un programa. AnimPascal permite guardar las acciones de los estudiantes. Se

consideran estas acciones como pasos en el proceso de solución del problema. Esto permite comprender la forma en que cada estudiante resuelve el problema. A partir de esta información, AnimPascal tiene dos objetivos: ayudar a los programadores principiantes en todo el proceso de desarrollo hasta la ejecución de un programa y ayudar a los profesores para descubrir "lagunas" de conocimiento y problemas de aprendizaje de sus estudiantes.

Cada vez que un estudiante recompila, se guarda automáticamente la última versión del código fuente y la correspondiente salida del compilador. El profesor puede recorrer las distintas compilaciones para buscar y ver el historial de errores cometidos por el/los alumno/s.

ProLearn

Creado por la Universidad de Coimbra [Mendes, 2006]. Es una herramienta que ayuda a los programadores noveles a resolver problemas usando comunicación basada en texto. Cuando los estudiantes crean el algoritmo, tienen la posibilidad de pedir ayuda si es necesaria. El principal objetivo del sistema es la atención del estudiante, de la manera más cercana a los problemas que vaya teniendo en el aprendizaje de programación.

ProLearn esta separado en tres módulos: el primer módulo maneja el lenguaje natural de conocimiento en diálogos entre estudiantes, el segundo módulo contiene los pasos y estrategias para desarrollar un algoritmo particular, el tercero analiza las acciones del estudiantes para crear una estrategia más conveniente para el estudiante si es necesario.

PL-Detective

Realizado en la Universidad de Colorado en 2004 [Amer, 2004]. Permite aprender nociones de programación de manera colaborativa. Este sistema es una implementación y una extensión de un lenguaje llamado MYSTERY, en realidad este lenguaje es prácticamente igual a C.

PL-Detective exporta siete interfaces llamadas interfaces semánticas. Cada interfaz semántica corresponde a un aspecto de la semántica de MYSTERY. Los grupos de estudiantes realizan programas en MYSTERY a través de una interfaz Web. El profesor o los propios estudiantes pueden diseñarse sus propios ejercicios. La parte colaborativa del programa es que se han agrupado a varios alumnos en un ordenador que tiene instalado la aplicación y discuten mientras realizan la aplicación. PL-Detective está formado por dos componentes: el primero compila los programas MYSTERY y el segundo los traduce a JAVA.

ELP

Realizado por Nghi Truong, Peter Bancroft y Paul Roe en la Universidad de Queensland (Australia) en 2003 [Truong, 2003] [Teague, 2007]. ELP es un entorno

interactivo de desarrollo basado en web para aprender a programar de manera colaborativa. Los estudiantes han de registrarse en el sistema, una vez validado el acceso el estudiante tiene un abanico de temas para aprender a programar, donde cada tema tiene una serie de ejercicios. En la ventana de temas cada alumno de manera individualizada tiene un historial de todos los ejercicios realizados. Los ejercicios realizados por el alumno tienen la posibilidad de ser descargados a la máquina del alumno.

La arquitectura de ELP está desarrollada usando tecnología servlet donde aloja en un servidor las aplicaciones utilizadas. Opera bajo un servidor Linux y su contenedor web es Apache Tomcat. Para almacenar los ejercicios a realizar utiliza la tecnología XML.

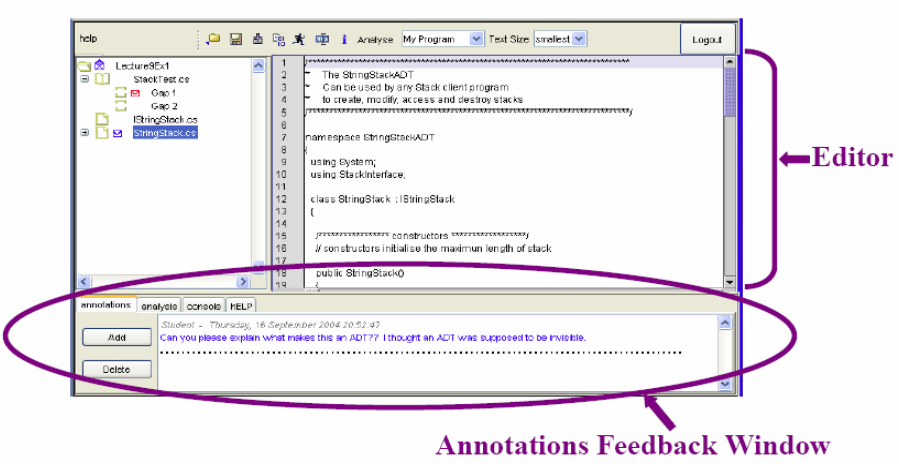

Figura 6: Interfaz de usuario

La interfaz de ELP contiene una barra de herramientas, un editor de código, un treeview (donde se almacenan las anotaciones de los estudiantes) y una ventana inferior donde visualizar el enunciado del ejercicio, analizar el código, ayuda o chatear. En la ayuda se puede visualizar tutoriales para facilitar la agrupación de recursos de aprendizaje en una misma aplicación.

BlueJ

Desarrollado en la universidad Monash de Melbourne y en la de Sidney en 2003 [Kölling, 2003]. BlueJ, es un entorno de aprendizaje enfocado a la programación orientada a objetos, que utiliza Java como lenguaje. Una vez instalado el sistema en el ordenador local, el estudiante dispone de un entorno que está dotado de todas las herramientas que necesita para desarrollar una aplicación con un lenguaje orientado a objetos. Hace especial énfasis en la relación entre el diagrama de clases y el código para facilitar la adquisición de conceptos de orientación a objetos.

A través del diagrama de clases como vista principal permite un estilo de interacción diferente a otros entornos. Mediante este diagrama el estudiante visualiza las clases y sus relaciones; además permite la creación de nuevas clases y nuevas relaciones gráficamente.

Por último, según el planteamiento pedagógico de los autores de BlueJ hay peligro de que se dedique mucho tiempo a conceptos de orientación a objetos y se descuiden otros conceptos como las estructuras de datos y algoritmos; hay que tener en cuenta que estos conceptos siguen siendo importantes y dedicarles tiempo para que los estudiantes también los entiendan y puedan aplicarlos correctamente.

En Julio de 2007 han sacado una ampliación de esta aplicación donde incorpora un entorno que tiene soporte para trabajo en grupo a través de un repositorio común para el grupo. También incorpora scripts que permiten incorporar dentro de tutoriales la interacción con el entorno.

Gild

Gild [Storey, 2003] [Rigby, 2005] es un proyecto que facilita el aprendizaje y la enseñanza del lenguaje de programación JAVA a través de este Plugin para Eclipse, enfocado especialmente a programadores noveles. El proyecto comenzó en 2003 en la Universidad de Victoria (Canadá), pero como es un proyecto libre, aun se siguen haciendo mejoras hoy en día.

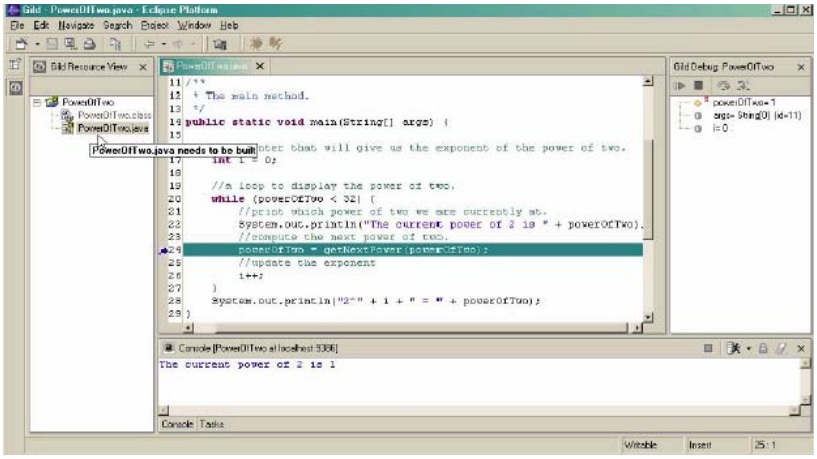

Figura 7: Eclipse con el plugin Gild

Para los estudiantes este plugin simplifica el entorno del Eclipse, pero las otras herramientas también están accesibles. Para los profesores permite integrar los materiales del curso con ejemplos del código y ejercicios. Estos ejemplos de códigos están alojados en la web de Gild, el propio plugin tiene un enlace directo a dicha web.

Para el trabajo en grupo, los estudiantes tienen un repositorio común donde almacenar los cambios realizados y descargar el código editado por parte de los demás componentes del grupo de trabajo.

Praktomat

Este sistema fue realizado en 2001 por la Universidad de Nassau. Praktomat [Zeller, 2000], es un sistema que permite a los estudiantes leer, revisar y evaluar programas de otros compañeros para mejorar la calidad y el estilo, todo ello a través de la Web. Los alumnos envían sus prácticas y pueden recuperar las prácticas enviadas por algún compañero para revisarlas y corregirlas (esta asignación se realiza por el sistema automáticamente). Una vez que una práctica esté revisada el autor pueden obtener las revisiones y volver a enviar el programa mejorado.

Sobre cada programa el sistema realiza una primera prueba automática en la que el sistema compila y realiza pruebas sobre cada programa. Parte de los casos de prueba son públicos para los estudiantes; pero hay otro conjunto de casos de prueba que son confidenciales si el usuario lo desea. Por otra parte el sistema facilita las revisiones cruzadas entre estudiantes estableciendo un sistema automático de revisión ciega en la que un alumno revisor pide al sistema una práctica para revisar; pero no conoce el autor de esta. Este proceso de revisión fomenta el aprendizaje y la mejora del estilo, por tanto la calidad del código de los revisores y revisados se ve mejorada en posteriores prácticas. El lenguaje de programación utilizado es Python y se puede usar en los sistemas operativos Linux, BSD y UNIX.

CourseMaster

Realizado por la Universidad de Nottingham en 2002 (última versión disponible) [Bell, 1999] [CourseMaster, 2000]. Es un sistema cliente/servidor para aprender a programar. Provee de funciones que permiten la evaluación automatizada del trabajo del estudiante. Además tiene la capacidad de detectar si se ha producido plagio en el programa del estudiante. El estudiante podrá desarrollar un programa, administrarlo, hacerlo rendir en el servidor y recibir la regeneración de manera instantánea.

CourseMaster es una completa reimplantación del sistema Ceilidh [Foxley, 2001], después de haber 10 años utilizando el sistema en algunas instituciones. Está escrito en JAVA y utiliza una base de datos para almacenar y compartir los documentos realizados por los alumnos. Para que los alumnos puedan compartir y descargar los ejercicios propuestos, éstos han de estar registrados previamente. Tiene una habitación virtual de encuentro entre los participantes donde se pueden comunicar de manera síncrona (mediante una sala de Chat) o asíncrona (MessageBoard)

3.2.1 Discusión de los sistemas de aprendizaje colaborativo

Para esta discusión, disponemos de mayor número de sistemas a analizar, por lo que podremos detectar una problemática general en el campo del aprendizaje de lenguajes de programación de manera colaborativa.

AlgoArena, como se destacaba en la presentación de este sistema en el apartado anterior, su principal funcionalidad era que despertaba el espíritu competitivo en los programadores noveles. Sin embargo, también hay que destacar que en base a ese espíritu competitivo permite facilitar el difícil primer paso que se produce en el aprendizaje de programación, un mundo muy abstracto, esto hace que el alumno tenga mayor motivación a aprender y a mejorar, añadiendo la posibilidad de comparar su nivel de programación al de los demás compañeros. Este sistema únicamente es útil como introducción a la programación ya que se centra en el aprendizaje de las nociones más básicas, tampoco es un punto a favor su antigüedad de la última versión, habiendo en la actualidad soluciones tecnológicas que se podrían haber aplicado.

En College, una gran ventaja es la gran autonomía en el aprendizaje por parte de los alumnos, no es el profesor quien se encarga de definir las sesiones de trabajo, ya que los encargados de esto son los propios alumnos. El haberlo desarrollado en JAVA es otro punto a favor en la autonomía de sistema operativa, ya que la plataforma JAVA nos permite que sus programas puedan ser ejecutados en sistemas operativos como Windows, Linux o Mac. Una limitación de College es que exige una comunicación síncrona, lo que hace necesario que todos los integrantes estén al mismo tiempo conectado, lo que supone una gran dificultad coincidir en mismos horarios, tampoco tiene un repositorio de conocimiento que guarde automáticamente los pasos para corregir un error.

OOP-Anim tiene como principal aportación no encontrada en otros sistemas la de poder simular y animar las soluciones, lo que permite que el alumno encuentre y, sobre todo, comprenda el por qué de su error que pueda cometer. Además de comprender mejor el funcionamiento del código que acaba de escribir. Los estudiantes simulan su solución propuesta y ven su correspondiente animación, lo que hace que sea un aprendizaje activo y productivo. Su interfaz sencilla permite que los estudiantes pierdan el menor tiempo posible en familiarizarse al entorno, pudiéndose utilizar en modo individual o en grupo. Aunque son muchas las virtudes en esta aplicación, su principal desventaja es la de no soportar el modelo cliente/ servidor, lo que hace que la única manera de trabajo colaborativo sea la de trabajar varias personas en un único equipo.

De HabiPro la principal aportación innovadora es la existencia del estudiante simulado que ya hemos descrito en el apartado anterior, un gran aporte ya que es un elemento muy útil para el trabajo colaborativo y que ayuda en casos de que ningún componente del grupo sea capaz de continuar con el ejercicio propuesto. Se ha mostrado muy eficaz en cuanto a la disposición a aprender, incluso en los estudiantes más pasivos. Es un sistema síncrono, pero no asíncrono, lo que no nos permite

trabajar en momentos que los demás no están conectados. Tampoco tiene un modo Offline que nos permitiera trabajar por nuestra cuenta. A pesar de ello, no son desventajas relevantes, por lo que es un sistema muy eficaz en el aprendizaje colaborativo de lenguajes de programación.

La herramienta ECPASCAL simplifica el trabajo y disminuye el tiempo de trascripción, ya que los estudiantes que trabajaron en forma individual tardaron más tiempo, e incluso algunos de ellos no completaron el trabajo. Además ayuda a trabajar de manera cooperativa, otro punto a destacar es su facilidad de uso. Como puntos en contra tenemos la no automatización del espaciado en la estructura del código mientras el usuario escribe el programa, lo que hace más dificultosa la tarea de comprensión del código, la interfaz del Chat no puede establecer una conversación en privado con otros compañeros.

Jeco tiene un componente llamado Jeliot 3 que utiliza un vocabulario común de creación al que usan los estudiantes, otra ventaja es su visualización automática. Soporta grupos de trabajo síncronos y asíncronos. Como elemento diferenciador de otros sistemas es la integración de Woven Stories como una subventana, con el fin de poder visualizar un manual mientras se intenta resolver un ejercicio de programación de manera colaborativa. Sin embargo, no permite editar código ni compilarlo de manera colaborativa, es útil como complemento de uso con otra aplicación (ej. Habipro).Por si solo, no es un método muy dinámico para aprender programación.

El sistema DPE tiene como bondad la gran capacidad de realizar tareas al mismo tiempo, ya que todos los usuarios ven los cambios y visualizan las salidas al mismo tiempo. La comunicación se puede dividir en subgrupos de discusión, que permite al profesor dividir a los alumnos en varios grupos de trabajo y que discutan entre ellos. DPE soporta a la vez comunicación, colaboración y coordinación. Como trabajo futuro está la comunicación por video, opción aún no disponible, únicamente se puede utilizar para aprender con JAVA.

En cuanto al sistema AnimPascal, su aportación más destacada es que permite guardar el camino seguido por cada alumno en la solución de un problema, lo que abre la posibilidad de ver su historial de errores cometidos y tenerlos en cuenta para proyectos futuros, posibilidad que también puede consultar el profesor para conocer la evolución del alumno. En contra, tenemos la ausencia de ayuda sobre la interpretación de los errores cometidos ni sobre la solución de los mismos. Además necesitan instalación y configuración por parte del usuario. Los estudiantes tienen problemas en la transición hacia entornos comerciales (Eclipse, NetBeans, JBuilder), al ser un entorno diferente a éstos.

En el caso de ProLearn, gracias al análisis que realiza el sistema a cada alumno de manera individualizada, le ayuda en la resolución de problemas, le estimula a aprender proponiendo ejercicios acorde a los fallos que pueda haber tenido. En contra de este proyecto, es que el proyecto aún está en desarrollo, la última versión sólo soporta el aprendizaje individual, pero en el trabajo futuro se contempla la posibilidad de trabajar de manera colaborativa.

PL Detective permite al alumno ser autodidacta ya que éste puede diseñarse sus propios ejercicios acordes a las carencias en programación que pueda tener, además que esos ejercicios diseñados si no se logran resolver, tenemos la opción de que la aplicación nos lo resuelva. A pesar de ser una interfaz Web que nos da la libertad de utilizarlo en cualquier lugar con conexión con Internet, es una aplicación monopuesto que nos limita en la experiencia colaborativa, aunque haya evaluaciones de la aplicación en trabajo en grupo. Otra desventaja es el uso de Mystery (un lenguaje propio similar a C) como lenguaje para aprender en lugar de utilizar uno real, el alumno ha de adaptarse a un lenguaje parecido o a otro nuevo innecesariamente. También se echa en falta la ausencia de herramientas de comunicación para trabajar de manera colaborativa a distancia.

En ELP cabe destacar la posibilidad de aprender C# (además de la posibilidad de aprender JAVA), un lenguaje que está teniendo mucha demanda en los últimos años. Los ejercicios están basados en conceptos conocidos para afianzar los conocimientos del alumno, reduciendo a su vez su complejidad. Existe también la posibilidad de compilar los ejercicios online a través del servidor, esto permite que no tengas que tener nada instalado (solamente JAVA Runtime Environment) en el ordenador para realizar y probar los ejercicios. Está optimizado para Internet Explorer, desconociendo el resultado que puede darse en otros navegadores como Opera o Mozilla Firefox.

El entorno BlueJ aporta la vista del diagrama de clases y el banco de objetos y la posibilidad de manipulación directa por parte del estudiante para facilitar que el estudiante se cree el modelo mental adecuado para entender la relación entre clases y objetos y el uso adecuado de cada uno de ellos, todo ello en un entorno sencillo y simple. Como desventajas tiene que el encargado de la instalación y configuración de la aplicación es el usuario, por lo que se hace necesario proporcionarle una guía de instalación y configuración, aunque esto sea un proceso fácil, es un obstáculo que ralentiza el comienzo del aprendizaje del estudiante. Otra desventaja es que no proporciona ayuda al estudiante sobre la interpretación de los errores cometidos, ni en la solución de éstos, lo que hace imprescindible la presencia del profesor para resolverlos.

El sistema Gild aprovecha todas las ventajas y herramientas que ofrece Eclipse, ya que nos facilita la familiarización desde el principio a un entorno de programación que muy probablemente utilizará el alumno en el futuro. Además clarifica los pasos de guardado, edición y compilación de código. Pero como inconvenientes encontramos la dificultad para poder acceder a ciertas APIs, en el caso del repositorio común es una herramienta que aún está en fase de prototipo, realizándose pruebas de evaluación. Se echa de menos también la incorporación de una ventana de Chat u otro medio de comunicación entre los participantes.

Praktomat nos ofrece colaboración en el aprendizaje de programación entre los propios alumnos, ya que éstos pueden evaluarse entre ellos. También tiene la posibilidad de garantizar la confidencialidad de corrección del código en caso de que

lo desee el alumno. Esta posibilidad de corrección se extiende a cualquier lenguaje de programación conocido por los participantes. La principal desventaja es que el sistema es únicamente asíncrono, imposibilidad de comunicarse entre los participantes en tiempo real.

Analizando Course Master, encontramos ventajas como la comunicación asíncrona y síncrona, permitiéndonos utilizar como lenguaje de aprendizaje cualquiera de los tres que soporta: C, C++ y JAVA. Además funciona en diferentes Sistemas Operativos como Windows 95, 98, NT, 2000, XP, Solaris o Linux. Como principal trabajo futuro se pretende implantar el envío automático de E-mails con los progresos que vaya obteniendo el estudiante al tutor, esto sería un aporte muy útil e insólito en el mercado, ya que ayudaría a automatizar la tarea de evaluación al profesor. El problema que encontramos es que los estudiantes son los que tienen que disponer de un entorno de programación para desarrollar sus actividades. Además requiere que los estudiantes escriban el programa completo desde el principio sin ayudas, lo que complica el aprendizaje.

3.2.2 Resumen

Nombre	**asíncrono/ síncrono**	**Metodología**	**Breve descripción**
AlgoArena	Síncrono	Autoaprendizaje Discusión	Es un sistema enfocado a estudiantes noveles de programación que consiste en ambientarlo en un combate de Sumo
College	Síncrono	Colaboración Discusión	Es un sistema dirigido al aprendizaje de JAVA que soporta la edición, compilación y ejecución de programas de manera colaborativa por parte de los alumnos, y la revisión del trabajo efectuado por los estudiantes por parte del profesor
OOP-Anim	Síncrono	Colaboración	Es un entorno de aprendizaje colaborativo escrito en JAVA que ayuda a los estudiantes a aprender nociones de programación orientada en objetos
HabiPro	Síncrono	Colaboración Discusión	Es un sistema síncrono, cliente/servidor y distribuido que pretende desarrollar en los alumnos buenos hábitos de programación
ECPASCAL	Síncrono	Colaboración	Es un entorno de programación

		Discusión	colaborativo con soporte para el aprendizaje de PASCAL
Jeco	Síncrono	Colaboración Discusión Información	Es un entorno de aprendizaje de lenguajes de programación colaborativo a través de la discusión de documentos y fragmentos de código.
DPE	Síncrono	Colaboración	Es un entorno de programación colaborativo con aplicación para ser usado por profesores y alumnos. Los programadores pueden ejecutar sus programas de manera remota.
AnimPascal	Asíncrono	Autoaprendizaje	Es un ejemplo de entorno de aprendizaje de programación. El propósito es ayudar a los estudiantes a comprender las fases de desarrollo, verificación, depuración y ejecución de un programa.
ProLearn		Autoaprendizaje	Es una herramienta que ayuda a los estudiantes a programar usando comunicación basada en texto
Pl-Detective	Asíncrono	Autoaprendizaje Discusión	Permite aprender nociones de programación de manera colaborativa.
ELP	Síncrono	Colaboración Discusión Información	Es un entorno interactivo de desarrollo basado en web para aprender a programar de manera colaborativa.
BlueJ	Asíncrono	Autoaprendizaje Colaboración Información	Es un entorno de aprendizaje enfocado a la programación orientada a objetos
Gild	Asíncrono	Colaboración Autoaprendizaje	Es un proyecto que facilita el aprendizaje y la enseñanza del lenguaje de programación JAVA a través de un Plugin para Eclipse
Praktomat	Asíncrono	Colaboración	Es un sistema que permite a los estudiantes leer, revisar y evaluar programas de otros compañeros para mejorar la calidad y el estilo, todo ello a través de la Web.
CourseMaster	Asíncrono y síncrono	Colaboración Discusión	Es un sistema cliente/servidor para aprender a programar.

4 Reflexión

Tras analizar uno a uno todos los sistemas más relevantes que hay en el mercado del aprendizaje colaborativo de lenguajes de programación, ya nos vemos capacitados para describir cómo se encuentra la situación de estos sistemas de manera general y la problemática que encontramos. Tras ver las desventajas de todos los sistemas en general se han encontrado deficiencias comunes como la no utilización de herramientas avanzadas de comunicación entre los participantes, concretamente en el caso de los sistemas que soportan comunicación. Actualmente disponemos de Internet de gran velocidad con un ancho de banda suficiente para soportar, por ejemplo, comunicaciones de audio y vídeo. El hecho de mantener una conversación y a la vez poder verse los participantes entre ellos, nos permite tener un mayor acercamiento entre los usuarios, en resumen un aprendizaje más cercano. Con las opciones colaborativas de sistemas de aprendizaje de lenguajes de programación existentes, prácticamente tenemos casi todas las soluciones que podrían satisfacer las necesidades de los profesores y alumnos, hay sistemas con un modo de aprendizaje colaborativo variado desde un repositorio común hasta depuradores comunes. Con esto no se quiere decir que con estas soluciones ya están resueltas las necesidades en la enseñanza de la programación, ya que siempre van a surgir nuevas ideas que mejoren a las soluciones existentes.

Sin embargo, lo mismo se podría dar un paso más en estos sistemas, como por ejemplo potenciar más la interactividad en base a las funcionalidades existentes a los sistemas actuales y los que están en desarrollo, es decir, complementar esas funcionalidades con herramientas interactivas como disponer de una herramienta interactiva que diese opción a escuchar un podcast o visualizar un videotutorial que explicase nociones de programación, todo ello mientras se realiza una actividad de programación con otros compañeros.

En muchos sistemas hemos encontrado que disponen un repositorio común donde todos los participantes pueden compartir los archivos. Sin embargo, sería algo interesante estudiar la posibilidad de incorporar en alguno de estos sistemas un sistema de intercambio de archivos entre los propios usuarios, es decir, que puedan mandarse archivos entre ellos de modo privado. Por ejemplo un sistema de intercambio P2P ayudaría bastante a la hora de intercambiarse archivos que no son relevantes para compartirlos en el grupo en conjunto, pero si para que lo aproveche un usuario que lo pueda necesitar en un determinado momento. Además hemos visto que en sistemas como Course Master que tienen como trabajo futuro la opción de envío automático de E-mails con los progresos que vaya obteniendo el estudiante al tutor, un aporte de gran utilidad. Sin embargo esta utilidad podría ampliarse mediante del envío de e-mails a los alumnos, es decir, los progresos de un participante del proyecto una vez registrados pudiesen ser enviados por e-mail a su vez a los demás participantes para ser avisados de los cambios realizados, en caso de que no se encontrasen conectado en el sistema en ese momento.

5 Conclusión

A lo largo de este estudio hemos hecho un análisis sobre el aprendizaje y enseñanza de lenguajes de programación de manera colaborativa. Nuestro trabajo en concreto ha sido el de mostrar las herramientas más relevantes que existen en el mercado en este campo, destacando sus principales ventajas y criticando también sus desventajas, además de mostrar aportes exclusivos y futuros trabajos cuyas funcionalidades son inexistentes en las demás aplicaciones. Este análisis de sistemas lo hemos dividido en dos grupos siguiendo el criterio de sistemas colaborativos enfocados al desarrollo y, en especial hincapié, sistemas colaborativos enfocados al aprendizaje de programación. Cada uno de estos entornos colaborativos se caracterizan por tener diferentes funcionalidades y métodos distintos para enseñar a programar, pero todos tienen el mismo objetivo en común: enseñar a programar de la manera más eficiente y sencilla posible, facilitando la tarea docente al profesor. Nuestra meta es enseñar a alumnos y, sobre todo, a profesores de programación de que existen estas herramientas y de que han sido exitosamente probadas en aulas, como hemos podido mostrar en los diferentes artículos que ha realizado estudios sobre la utilización de estos sistemas en la aulas. Todo son ventajas y los inconvenientes son casi inexistentes, únicamente son sugerencias de cómo mejorar esas ventajas.

Estas herramientas por si solas han demostrado ser efectivas en su objetivo particular y en cada etapa del aprendizaje. Además de haber clasificado las herramientas en cuanto a la orientación al desarrollo o a la educación, a su vez en los modos de comunicación para la colaboración hemos visto que pueden ser síncronos o asíncronos. En cuanto a las diferentes funcionalidades de los sistemas, hemos podido observar la gran variedad de utilidades de aprendizaje colaborativo que existen. Utilidades como el "combate entre estudiantes", edición y depuración de código por turnos, animaciones en la ejecución de código, decisiones mediante votaciones de los estudiantes, participación de un estudiante simulado, edición de código mediante privilegios, wiki, visualización de documentos en la misma pantalla, revisión del código por parte del programa o por los estudiantes, ayuda del sistema al alumno en todo momento… Todas ellas con la finalidad de hacer lo más agradable posible el difícil paso que es aprender a programar.

Referencias

[Amer, 2004] Amer Diwan, William M. Waite, Michele H. Jackson: PL-detective: a system for teaching programming language concepts. SIGCSE 2004: 80-84 (2004)

[Antunes, 2006] Ricardo Antunes, Benjamim Fonseca, Paulo Martins, Leonel Morgado, "Use of 3-D virtual environments to support the learning of programming", m-ICTE 2006, Sevilla, Spain, FORMATEX, (2006)

[Atramiz, 2003] Atramiz I., Villegas H., Perez E., Montilla G. "Diseño e Implementación del Ambiente Cooperativo de Aprendizaje de Programación

ECPASCAL". Memorias del Congreso Internacional Edutec 2003. Universidad Central de Venezuela. Venezuela. (2003)

[Bell, 1999] Bell, B. & Kaplan, D.. CourseMaster: Modeling A Pedagogy for On-line Distance Instruction. In Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 1999 (pp. 1720-1725). Chesapeake, VA: AACE. (1999)

[Bravo, 2004] Bravo, Crescencio, Redondo, M.A., Ortega, M.; Aprendizaje en grupo de la programación mediante técnicas de colaboración distribuida en tiempo real. Actas del V Congreso Interacción Persona Ordenador, Lleida. (2004)

[Breeding, 2005] Breeding, Marshall. Unconventional and innovative: the Open Croquet Project. Medford NJ. November/December (2005)
Web oficial: http://www.croquetconsortium.org/index.php/Main_Page

[Chang-Hyun, 1999] Chang-Hyun Jo, Jea Gi Son, Younwoo Kang, Phill Soo Lim: The Distributed Programming Environment on the Internet. SAC 1999: 85-90 (1999)

[Chang-Hyun, 2003] Chang-Hyun Jo, Allen J. Arnold: A Portable and Collaborative Distributed Programming Environment. Software Engineering Research and Practice 2003: 198-203 (2003)

[Cheyer, 2006] Adam Cheyer and Joshua Levy. A Collaborative Programming Environment for Web Interoperability. Semantic Wiki 2006 workshop in Montenegro (2006)
Web oficial: http://www.wubhub.com:4110/webl/WubHub_DoIt?cmdline=home

[Chu-Carroll, 2000] Mark Chu-Carroll, Sara Sprenkle: Coven: brewing better collaboration through software configuration management. SIGSOFT FSE (2000)

[CourseMaster, 2000] CourseMaster, Vol. 2002 School of Computer Science & IT, The University of Nottingham, UK (2000)

[Esteves, 2003] M. Esteves and A. Mendes, OOP-Anim, a system to support learning of basic object oriented programming concepts in: CompSysTech' 2003 - International Conference on Computer Systems and Technologies (2003)

[Esteves, 2004] Esteves, M. e Mendes, A.J., A Simulation Tool to Help Learning of Object Oriented Programming Basics. In Proceedings of 34th ASEE / IEEE Frontiers in Education Conference, pp. F4C7-F4C12, Savannah, Estados Unidos, Octubre (2004)

[Esteves, 2006] Esteves M., Morgado L., Martins P., Fonseca B. The use of Collaborative Virtual Environments to provide student's contextualisation in programming. Proceedigns of m-ICTE 2006 (2006)

[Ezeiza, 2007] Ainhoa Ezeiza Ramos. MOODLE en la enseñanza de lenguas: proyecto Covcell. VirtualCampus (2007)

[Foxley, 2001] Foxley E., Higgins C., Symeonidis P., Tsintsifas A., The CourseMaster Automated Assessment System - a Next Generation Ceilidh, Conference on Computer Assisted Assessment to support the ICS disciplines, University of Warwick, April 5th - 6th, (2001)

[Hsu, 1993] Hsu, Lockwood. *Collaborative Computing.* Publicado en la Revista Byte. Edición de Marzo (1993)

[Kölling, 2003] Kölling, M., Quig, B., Patterson, A. and Rosenberg, J., The BlueJ system and its pedagogy, Journal of Computer Science Education, Special issue on Learning and Teaching Object Technology, Vol 13, No 4, (2003)

[Magnusson, 1993] Boris Magnusson, Ulf Asklund, Sten Minör: Fine-Grained Revision Control for Collaborative Software Development. SIGSOFT FSE (1993)

[Mendes, 2006] Mendes, A. "ProLearn, a platform to support programming learning" Methods, Materials and Tools for Programming Education, Tampere - Finlandia, (2006)

[Moreno, 2004] Moreno, A.; Myller, N.; Sutinen, E. JeCo, a Collaborative Learning Tool for Programming. Visual Languages and Human Centric Computing, 2004 IEEE Symposium on Volume , Issue , 30-30 (2004)

[Newman, 1989] Newman, D., Goldman, S.V., Brienne, D., Jackson, I. & Magzamen, S. (1989) Computer mediation of collaborative science investigations. Journal of Educational Computing Research, 5 (2), pp. 151-166.(1989)

[Phprojekt] Web Oficial Phprojekt: http://www.phprojekt.com

[Posner, 1993] Posner, I.R. and Baecker, R.M. How People Write Together, Proceedings of the Twenty-fifth Annual Hawaii International Conference on System Sciences, 1992, 127-138. Reprinted, with slight modification, in Baecker, R.,M. Readings in Groupware and Computer-supported Cooperative Work: Facilitating Human-Human Collaboration, Morgan Kaufmann, (1993)

[Rigby, 2005] Rigby, P.C., S. Thompson, "Study of novice programmers using Eclipse and Gild," Eclipse Technology eXchange (eTX) Workshop at OOPSLA 2005, San Diego, California, (2005)

[Satratzemi, 2001] Satratzemi, Maria; Dagdilelis, Vassilios; Evagelidis, Georgios A system for program visualization and problem-solving path assessment of novice programmers. ITiCSE '01: Proceedings of the 6th annual conference on Innovation and technology in computer science education. (2001)

[Shen, 2000] H. Shen, C. Sun, RECIPE: a prototype for Internet-based real-time collaborative programming, In Proceedings of the 2nd International Workshop on Collaborative Editing Systems in conjunction with ACM CSCW Conference, Philadelphia, Pennsylvania, USA, (2000)

[Soller, 2005] A. Soller, A. Martinez, P. Jermann, and M. Muehlenbrock. From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. International Journal of Artificial Intelligence in Education, 15:261-290, (2005)

[Storey, 2003] Storey, M.-A., M. Sanseverino, D. German, D. Damian, A. Damian, J. Michaud, A. Murray, R. Lintern, J. Chisan, M. Litoiu, and D. Rayside, "Adopting GILD: an integrated learning and development environment for programming," ACSE 2003: 3rd Int. Workshop on Adoption-Centric Software Engineering ICSE 2003, the 25th Int. Conf. on Software Engineering, Portland, USA, (2003)

[Suzuki, 2002] Hideyuki Suzuki and Hiroshi Kato:' Identity formation /transformation as the process of collaborative learning through AlgoArena', CSCL2: Carrying Forward the Conversation, Lawrence Erlbaum Assoc., pp.275-296 (2002)
Web de Hiroshi Kato: http://pa.nime.ac.jp/kato/index.php?Hiroshi%20Kato%20Lab

[Teague, 2007] Teague, Donna M. and Roe, Paul Learning to Program: Going Pair-Shaped. ITALICS 6(4). (2007)

[Truong, 2003] Nghi Truong, Peter Bancroft, Paul Roe, A Web Based Environment for Learning to Program, 26 Australasian Computer Science Conference, Vol 25, No 1, Australian Computer Science Communications, (2003)

[Villegas, 2003] Villegas H., Atramiz I., Delgado M., Dorta R. "ECPASCAL Una Aplicación de Programación Colaborativa". Resúmenes de la XLXIII Convención Anual de ASOVAC. Acta Científica Venezolana: 54(sup. 1) pp. 315. Venezuela. (2003)

[Vizcaíno, 2000] Aurora Vizcaíno, Juan Contreras, Jesús Favela, Manuel Prieto: An Adaptive, Collaborative Environment to Develop Good Habits in Programming. Intelligent Tutoring Systems (2000)

[Vizcaíno, 2007] Vizcaíno, A. Un Estudiante Simulado que Detecta y Corrige Situaciones Negativas en Entornos Colaborativos. páginas 149-156 (2007)

[Williams, 2000] Williams, Laurie, Kessler, Robert R.. (2000) Experimenting with Industry's "Pair-Programming" Model in the Computer Science Classroom. Journal on Software Engineering Education, (2000)

[Zeller, 2000] Zeller, A. Making students read and review code. In Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, ACM (2000)