

## Ensayos

# Comparación de la programación individual y por pares en los cursos universitarios

### Resumen

En este artículo se presentan los resultados de un estudio que comparó la percepción de la programación por pares y la programación individual en el entorno universitario. El estudio se centró en 94 estudiantes de la carrera profesional de Licenciatura en Informática de la Facultad de Comercio, Administración y Ciencias Sociales de Nuevo Laredo en la Universidad Autónoma de Tamaulipas, México. Del total de los participantes, 54 programaron por pares y 40 individualmente. Todos desarrollaron un mismo programa bajo las mismas condiciones y al finalizar contestaron un cuestionario. Se utilizó la prueba estadística de Mann Whitney para buscar diferencias entre las percepciones recabadas. Comparados con los alumnos que trabajaron de manera individual, los alumnos que programaron por pares percibieron que les resultó más fácil su trabajo; así mismo, expresaron que detectaron errores más rápidamente y que estuvieron más satisfechos con el software final. Entre quienes trabajaron en parejas, se encontró que los alumnos de los últimos semestres tuvieron una mejor experiencia en este ejercicio y que quienes habían cursado materias de programación preuniversitarias tienen mayor preferencia para trabajar solos. Los resultados permiten establecer antecedentes contextualizados que derivarán en la implementación de estrategias de capacitación universitaria en el área de programación.

Ramón Ventura Roque Hernández,  
Juan Antonio Herrera Izaguirre,  
Adán López Mendoza, Diego  
Bravo Herrera

Universidad Autónoma de Tamaulipas,  
México.

### Abstract

This paper presents the results of a research project that compared the perceptions about pair and solo programming in university courses. The study focused on 94 students of the Computer Science undergraduate program at the Faculty of Commerce, Business Administration and Social Sciences at the Autonomous University of Tamaulipas in Nuevo Laredo, Mexico. 54 students worked in pairs, and 40 performed solo programming. All of them developed a program with the same requirements and under the same circumstances. A questionnaire was answered by the participants at the end of this task. Afterwards, the responses were analyzed, and Mann Whitney tests were performed to establish statistically significant differences. Students who worked in pairs perceived that their work was easier and that they were able to find errors faster; they were also more satisfied with their final software. Pair Programmers who were enrolled in the advanced semesters of their academic program reported a better experience while developing software. On the other hand, pair programmers with preuniversity experience in programming were more willing to work alone. These results will help by providing contextualized backgrounds that could lead to the implementation of teaching strategies for university programming courses.

**Palabras clave:** Desarrollo de software, educación, materias, metodologías, universidad.

### Résumé

Dans cet article, on présente les résultats d'une étude qui compare la perception de la programmation par paires et la programmation individuelle dans un contexte universitaire. L'étude est axée sur 94 étudiants de la filière professionnelle de Licence en Informatique de la Faculté de Commerce, Administration et Sciences sociales de Nuevo Laredo, Université Autonome de Tamaulipas, Mexique. Du total des 94 participants, 54 ont été programmés par paires et 40 individuellement. Tous ont réalisé le même programme dans les mêmes conditions et à la fin tous ont répondu à un questionnaire. On a utilisé le test statistique de Mann Whitney pour rechercher des différences entre les perceptions récoltées. Comparés avec les élèves qui ont travaillé de manière individuelle, les élèves ayant travaillé par paires ont perçus le travail comme étant plus facile. Ils ont aussi exprimé qu'ils avaient détecté les erreurs plus rapidement et qu'ils étaient plus satisfaits avec le logiciel final. Pour ceux qui ont travaillé par paires, on a trouvé que les élèves des 2 derniers semestres ont eu une meilleure expérience pour cet exercice et que ceux qui avaient suivi des matières de programmation préuniversitaires préfèrent travailler seuls. Les résultats permettent d'établir des antécédents contextualisés qui déboucheront sur la mise en place de stratégies de formation universitaire dans le domaine de la programmation.

## Introducción

La programación por pares es una práctica que se popularizó con la aparición de la Programación Extrema (Beck y Andres, 2004), un enfoque ágil que se encuentra entre los más usados en la ingeniería del software (Rizwan y Quershi, 2011), junto con Scrum y Kanban (Singh, Mishra, Singh, & Upadhyay, 2015). La programación por pares consiste en desarrollar y probar programas de cómputo con parejas de personas que comparten una misma computadora. La literatura sugiere que al trabajar con otra persona, la calidad del software aumenta, la solución de problemas puede ser más rápida y precisa, además de que se ahorra tiempo en el desarrollo y se fomenta la creatividad.

A pesar de que la programación por pares es muy conocida y ha sido abordada ampliamente en la literatura, aún hay discrepancia en las opiniones respecto a ella. Por otra parte, no ha sido investigada en la misma intensidad en entornos académicos como en ambientes de negocios. Se debe considerar además que no todos los supuestos existentes en la ingeniería de software están respaldados por evidencias empíricas (Juristo y M. Moreno, 2001) y que un enfoque como éste podría ser adecuado para un escenario particular y podría no serlo para otro diferente. Solamente los datos provenientes de la práctica en contextos específicos podrían brindar información confiable en este sentido.

En este artículo se presentan los resultados de una investigación que se realizó en el entorno universitario tomando en cuenta la percepción estudiantil con la finalidad de brindar una aproximación a la respuesta a las siguientes interrogantes:

1) ¿Qué diferencias existen entre el uso de la programación por pares y la programación individual en cursos universitarios relacionados directamente con el desarrollo de software?

2) ¿Qué diferencias existen en la aplicación de la programación por pares en algunos grupos de estudiantes universitarios? En este artículo se centra la atención en: a) Alumnos con y sin experiencia preuniversitaria en programación. b) Alumnos de los primeros y de los últimos semestres de su carrera profesional.

Los resultados obtenidos establecen antecedentes que permitirán definir lineamientos para mejorar las estrategias de capacitación en el área del desarrollo de software dentro de nuestras aulas universitarias. El trabajo está estructurado de la siguiente manera:

primero se presentan los antecedentes teóricos de la programación por pares. Después se encuentran los detalles del método seguido en esta investigación. Luego se muestran los resultados obtenidos, los cuales posteriormente son discutidos para dar paso a las conclusiones finales de este artículo.

## Antecedentes

### *La programación en los cursos universitarios*

En la experiencia de los autores, las materias universitarias de programación que se ofertan en algunas de las carreras profesionales de informática con planes de estudio semestrales requieren que sus alumnos asistan a clases durante un periodo aproximado de 16 semanas hábiles. Durante este tiempo, se deben realizar prácticas y proyectos, así como también se deben desarrollar contenidos temáticos y evaluaciones. Tal es el caso de la carrera de Licenciatura en Informática en la Universidad Autónoma de Tamaulipas, México, en donde se condujo la presente investigación. En este contexto, es frecuente que se promueva más la obtención de software funcional a través de una programación individual y aislada y no a través de una participación activa que propicie la retroalimentación. Es comprensible, sin embargo, que esto ocurra debido a las restricciones de tiempo, a la gran cantidad de contenidos temáticos y a la naturaleza prescriptiva de los programas de estudio.

Las características que describen la situación actual sugieren que el desarrollo de este tipo de software debería realizarse con un enfoque ágil (Agile Alliance, 2015), con el que los alumnos construyan en corto tiempo programas de cómputo funcionales y apegados a los requerimientos del usuario, aplicando buenas prácticas. Si además, se contara con la retroalimentación de otro compañero, la interacción podría favorecer las competencias desarrolladas en el área de programación, el proceso de desarrollo podría ser más agradable y el producto final podría verse favorecido en su calidad. Estos planteamientos resultan particularmente relevantes si se les analiza desde diversas aristas:

a) Las materias de programación son percibidas como difíciles y suelen tener importantes índices de reprobación (Roque, Salinas, López, Mota, y Flores, 2014; Zhang, Zhang, Stafford, y Zhang, 2013). La programación por pares podría apoyar a que los estu-

diantes mejoren su rendimiento escolar y adquieran nuevas habilidades con mayor facilidad.

b) Elaborar software de buena calidad y con buenas prácticas en el entorno universitario debe ser una prioridad, ya que es aquí donde se gestan muchas soluciones de software plenamente funcionales para Pequeñas y Medianas Empresas (PYMES) y otros organismos.

c) Se debe tener presente que en el entorno universitario se prepara a los desarrolladores de software profesionales del futuro con conocimientos, habilidades y valores que les apoyarán en la creación de software de calidad una vez que hayan graduado y se desempeñen en su campo laboral.

d) Aunque en la literatura científica existen reportes de investigaciones que aluden a la programación por pares (Prabu y Duraisami, 2015; Salleh, Mendes, y Grundy, 2014; Swamidurai y Umprass, 2012), en su mayoría fueron conducidos en otros países. Es importante contextualizar esta técnica y estudiar si en realidad su uso resulta benéfico en nuestro entorno. Se debe tomar en cuenta que la infraestructura educativa, los contenidos de los programas de estudio, el tiempo para desarrollar los cursos, así como los antecedentes culturales y socioeconómicos de los estudiantes son distintos en otros países. Esto podría influir para que la programación por pares resultara inapropiada como estrategia didáctica en nuestras aulas ya que se fundamenta en aspectos de interacción y comunicación humana, cuyos estándares podrían ser diferentes en nuestros estudiantes debido a los factores mencionados anteriormente.

### *La programación por pares*

En el pasado, el desarrollo de software se realizaba siguiendo enfoques rígidos en donde existían fases prescriptivas, se producía mucha documentación, se privilegiaban el software y el proceso sobre el equipo de trabajo, no había flexibilidad para adaptarse a los cambios que surgían y el plazo de entrega al usuario final era muy extenso. Pronto quedó en evidencia que este tipo de enfoques no satisfacía las demandas de las aplicaciones informáticas emergentes que cada vez eran más complejas.

Los principios y prácticas ágiles surgieron para contribuir en la construcción rápida de programas de cómputo eficientes y útiles. Las prácticas ágiles son un conjunto de actividades orientadas a la obtención de

software funcional a través de un proceso en el que el cliente y el equipo de desarrollo trabajan conjuntamente (Pressman y Maxim, 2014). La agilidad privilegia a los individuos y sus relaciones sobre los procesos y herramientas; también pondera mayormente el software de calidad entregado rápidamente sobre la documentación que pueda generarse, al mismo tiempo que acepta los cambios en los requerimientos.

Existen distintos enfoques ágiles para la creación de aplicaciones informáticas. Uno muy conocido es la Programación Extrema, la cual adopta a la programación por pares como una práctica fundamental. Cuando se programa por pares, dos personas trabajan en un mismo equipo de cómputo; cada una utiliza el teclado en diferentes momentos. Durante el proceso de desarrollo, ambos participantes se comunican, toman decisiones y complementan su trabajo. En las parejas no existe el rol de jefe o de subordinado y ambos deben estar de acuerdo en trabajar juntos.

De acuerdo a Beck y Andres (2004) los programadores por pares se mantienen mutuamente en sus labores, aclaran ideas y toman la iniciativa cuando su pareja no puede avanzar. Sin embargo, esta modalidad también reconoce la necesidad de contar con momentos de trabajo individual; de tal manera que una persona puede apartarse para desarrollar ideas por sí misma y luego regresar para discutir las con su pareja. Este mismo tratamiento no puede darse al código, el cual siempre debe escribirse en pares. Las parejas pueden tomar repetidos descansos durante una jornada y los integrantes deben cambiar de pareja frecuentemente.

Se ha reconocido que la programación por pares puede ser cansada pero brinda buenos y rápidos resultados al proyecto completo ya que mejora la calidad del software, ahorra recursos y en general, hace que el proceso de desarrollo resulte más divertido (Kendall y Kendall, 2013). Sin embargo, estos beneficios podrían verse afectados por situaciones relacionadas con aspectos humanos de los programadores participantes. Por ejemplo, la incomodidad provocada por la invasión al espacio personal, la falta de higiene del compañero, la inmadurez emocional, la intolerancia a las diferencias, la desconsideración, e incluso, la atracción sexual.

### *Trabajo previo*

En la literatura se reconoce que la programación por pares ha sido el tema central de muchas investiga-

ciones, sin embargo, poco ha sido abordada desde el entorno académico (Prabu y Duraisami, 2015); en este trabajo, Prabu y Duraisami encontraron que aunque esta técnica de programación tiene algunas desventajas, puede ser altamente benéfica ya que promueve el deseo de trabajar de manera colaborativa y aumenta la percepción sobre la buena organización y el ahorro de tiempo. En el trabajo de Coman, Robillar, Silliti y Succi (2014), se menciona la falta de consenso en los resultados de investigaciones relacionadas con la programación por pares; el objetivo de su estudio fue encontrar los mejores usos de esta práctica. Por su parte, Salleh et al. (2014) también identifican la discrepancia entre los hallazgos reportados por distintas investigaciones en esta área y profundizan en los efectos de los rasgos de la personalidad de los estudiantes de educación superior en la programación por pares. En la investigación de Plonka, Sharp, van der Linden y Dittrich (2015) se destaca la programación por pares como una actividad muy relevante y se estudia la manera en la que el conocimiento se transfiere entre las parejas.

Kropp y Meier (2013) destacan la enseñanza de los enfoques y prácticas ágiles en la universidad debido a su importancia y utilidad en el mundo empresarial mientras que Schroeder, Klarl y Kroib (2012) ponen de manifiesto la importancia del lado humano del desarrollo de software y presentan los cursos prácticos en laboratorio como medio para enseñar las prácticas ágiles en las escuelas. Por otra parte, de acuerdo a McDowell, Werner, Bullock y Fernald (2006), la programación por pares produce programadores más capaces y con más confianza y al utilizarla en el entorno universitario, se promueve el aprendizaje colaborativo

así como también se incrementa el número de estudiantes que continúan con la intención de graduarse de un programa universitario en informática.

## Método

### Participantes y muestreo

Esta investigación se realizó con alumnos matriculados en los semestres primero, tercero, quinto y séptimo de la carrera de “Licenciatura en Informática” en la Facultad de Comercio, Administración y Ciencias Sociales de la Universidad Autónoma de Tamaulipas, México. No se incluyeron alumnos de los semestres pares porque al momento de la realización de este trabajo, esos grados no estaban siendo ofertados por la universidad.

Para este estudio se tomaron en cuenta las respuestas de 94 estudiantes, de los cuales, 54 programaron por pares y 40 de manera individual. Ninguno de los participantes había desarrollado software anteriormente siguiendo los principios de la programación por pares, pero todos conocían la programación individual, ya que todos sus trabajos hasta ese momento habían sido realizados en solitario.

Las principales características de la muestra se presentan en la Tabla 1. Durante el desarrollo de esta investigación, se descartaron dos cuestionarios por no proveer respuestas para un alto porcentaje de preguntas: la de un alumno de tercer semestre y la de un alumno de quinto semestre. Ambos programaron por pares. Por otra parte, en el grupo de séptimo semestre se decidió que los alumnos no trabajaran individualmente, ya que era un número reducido de estudiantes y el énfasis de esta investigación era sobre la programación por pares. Tomando en cuenta

Tabla 1. Características de los participantes y de los tipos de proyectos tomados en cuenta para esta investigación.

Semestre	Total de alumnos participantes	Alumnos registrados en la materia	Alumnos que programaron en solitario	Alumnos que programaron por pares	Materia que cursaban	Lenguaje utilizado	Tipo de proyecto realizado
1	34	43	18	16	Fundamentos de la informática y metodología de la programación	Visual Basic .Net	Consola
3	35	47	16	19	Programación básica	Visual Basic. Net	Formas de Windows
5	15	23	6	9	Programación avanzada	C#	Formas de Windows
7	10	19	0	10	Diseño y desarrollo de aplicaciones	C#	Formas de Windows

estas consideraciones, la Tabla 1 muestra únicamente los datos de los participantes que se incluyeron en el análisis de los datos de este trabajo.

La organización de los equipos de trabajo se realizó de manera aleatoria. Se utilizó la tabla de números aleatorios uniformemente distribuidos que se encuentra en (Coss Bu, 1995). El proceso se llevó a cabo en dos etapas: primero, con los números de la tabla se asignaron alumnos para trabajar en una de dos modalidades: individualmente o por pares. Luego, para los alumnos que trabajarían en parejas, se realizó un segundo proceso en el que se utilizaron pequeñas tarjetas de papel extraídas de un contenedor. Las tarjetas contenían números distintos que representaban los equipos que se formarían. Cada tarjeta estaba duplicada, de tal manera que siempre había dos alumnos que obtenían el mismo número y por lo tanto conformaban una pareja de trabajo.

### Escenario

Se utilizó un laboratorio ubicado dentro de las instalaciones de la universidad, en donde regularmente los alumnos toman sus clases de programación. Este laboratorio cuenta con treinta computadoras con procesador i5, 8 Giga bytes de memoria RAM y 1 Tera Byte de disco duro. En cada equipo había acceso a internet y estaba instalado el entorno de desarrollo Visual Studio 2013 en español, el cual proporcionaba acceso a los lenguajes Visual Basic .Net y C#. Se cuidó que las condiciones de tiempo de desarrollo, distribución física de los participantes, ventilación e iluminación de la sala fueran las mismas en todos los casos.

### Instrumento

Para la recolección de datos, se utilizó el cuestionario mostrado en la Tabla 2. Antes de aplicar este instrumento, fue revisado por dos expertos y fue sometido a una sesión de lluvia de ideas entre los autores de este artículo, quienes lo crearon durante el otoño de 2015 con el objetivo de recabar percepciones de los estudiantes acerca de: desarrollo de software en parejas, su experiencia en este ejercicio y sus preferencias personales sobre la programación. Actualmente aún se encuentra en proceso de validación y refinamiento.

Se utilizó una escala de Likert en las primeras trece preguntas y una escala de 0 a 10 en las cuatro últimas.

Tabla 2. Cuestionario aplicado a los participantes

1.	La metodología que seguí es adecuada para ser usada en el ámbito de negocios.
2.	Resultó fácil trabajar con la metodología utilizada.
3.	Resulta mejor utilizar otra metodología diferente a la que utilicé.
4.	Me es más cómodo trabajar solo.
5.	Es importante aprender a desarrollar software en equipo.
6.	Trabajar con esta metodología reduce el tiempo de desarrollo.
7.	Con esta metodología se detectaron errores rápidamente.
8.	La calidad del software final resulta mejor utilizando esta metodología.
9.	Cuando se programa por pares, el nivel de confianza de los programadores es mayor que cuando se trabaja individualmente.
10.	Programar individualmente es más fácil que programar en pares.
11.	Me motiva más programar por pares que individualmente.
12.	El nivel de satisfacción de los programadores es mayor cuando se programa por pares que cuando se programa individualmente.
13.	Basándome en esta experiencia, me gustaría volver a programar de esta manera.
14.	¿Qué tanto le gusta programar? (De 0 a 10)
15.	¿Qué tanto le gustó su experiencia de desarrollo de software en este ejercicio?(De 0 a 10)
16.	¿Qué tan satisfecho está con el programa obtenido? (De 0 a 10)
17.	¿Cuál fue su nivel de estrés durante este ejercicio? (De 0 a 10)

Las posibles respuestas para la escala de Likert se identificaron numéricamente de la siguiente manera:

1. Completamente en desacuerdo,
2. En desacuerdo,
3. Neutral,
4. De acuerdo,
5. Completamente de acuerdo.

### Procedimiento

El experimento se llevó a cabo durante sesiones regulares de clase con cada uno de los grupos estudiados. Los alumnos asistieron a una sola sesión regular de dos horas programada en su horario de clases. No se les mencionó con antelación ningún detalle del experimento ni de la fecha en la que se realizaría. Tampoco se les ofreció ningún tipo de estímulo académico o económico por su participación.

Los alumnos que programaron individualmente para este estudio no hicieron nada diferente a lo que estaban acostumbrados: se sentaron frente a una computadora e intentaron desarrollar por ellos mismos un programa durante el tiempo que se les indicó. Sólo que en este caso, igual que los programadores en parejas, no pudieron interactuar con otras personas por ningún medio, pero sí pudieron consultar notas, ejercicios previos y materiales de internet.

La distribución del tiempo de la sesión de trabajo fue la siguiente: durante los primeros veinte minutos se esperó a que todos los alumnos llegaran al laboratorio. Quienes llegaron después fueron excluidos de esta investigación. Posteriormente se procedió a identificar el número de alumnos que estaban presentes y se les dio una breve explicación técnica de cómo trabajarían de manera individual o por pares. Se cuidó en todo momento de no brindar información tendenciosa a favor de alguno de ambos enfoques. Posteriormente se les presentaron los requerimientos del programa que deberían realizar y se formaron los equipos de trabajo. Este proceso expositivo tuvo una duración de veinte minutos. Inmediatamente después, los alumnos dispusieron de 70 minutos para concluir el programa que se les solicitó y de diez minutos más para completar un cuestionario que se contestó de manera individual.

En el laboratorio de cómputo, los estudiantes se distribuyeron alternadamente en cada computadora de acuerdo a su modalidad de trabajo. Es decir, había un alumno programando individualmente, luego una pareja, en seguida otro alumno solo, y así sucesivamente. Todos los alumnos podían realizar consultas en internet y revisar notas personales o programas realizados en clase con anterioridad. Los alumnos de diferentes equipos no podían conversar entre sí por ningún medio; tampoco podían copiar o revisar el trabajo de otras personas. Los alumnos que trabajaron por pares tenían que utilizar una sola computadora y alternar el teclado con su compañero cada cinco minutos. Los investigadores se encargaron de cuidar todos estos aspectos durante las sesiones: organizaron y supervisaron a los participantes, así como también les indicaron los tiempos para cada actividad.

Los participantes guardaron su trabajo en la plataforma universitaria en línea denominada Blackboard (UAT, 2015) en cuanto finalizaron su programa, o bien, en cuanto el tiempo de desarrollo terminó. Posteriormente, respondieron el cuestionario que los investigadores les proporcionaron; luego, abandonaron la sala.

### *Tipo de estudio y diseño de investigación*

Este artículo presenta un estudio de tipo experimental en un laboratorio en donde se sometió a los alumnos de un curso universitario al desarrollo de software con condiciones controladas con el objetivo

de observar los resultados producidos en los participantes. Este tipo de investigación puede aplicarse exitosamente en el campo de la educación cuando se pretende probar los beneficios de un método o material didáctico (Niño, 2011). El diseño de investigación fue experimental sólo con prueba posterior y grupo de control (Salkind, 1999).

### *Definición conceptual y operacional de variables*

Las definiciones conceptuales y operacionales de las variables de interés en esta investigación se encuentran en la Tabla 3 y en la Tabla 4. Todas estas variables en escala ordinal fueron estudiadas a través de una comparación de acuerdo al tipo de programación desarrollado. El tipo de programación es una variable nominal con dos categorías: “por pares” (para el grupo experimental), o “individual” (para el grupo de control) y se define conceptualmente como la metodología que los participantes siguieron para la creación del programa que se les planteó.

### *Intervención*

A todos los alumnos se les pidió realizar un programa con los requerimientos mostrados en la Tabla 5 y se les presentó también un ejemplo con los resultados que se deberían mostrar en cada ejecución. Toda esta información fue expuesta en dos diapositivas con el apoyo de un proyector. Los alumnos del grupo experimental programaron por pares y los del grupo de control programaron de manera individual, tal como se indica en la descripción del procedimiento de este artículo.

A los alumnos se les dijo que cada quien podría implementar la interfaz de usuario que desearan siempre y cuando el programa proporcionara los resultados solicitados. Los participantes utilizaron los lenguajes de programación y los tipos de proyectos mencionados previamente en la Tabla 1.

### *Análisis de datos*

Las respuestas proporcionadas por los participantes fueron capturadas en el paquete estadístico SPSS (Wagner, 2014), en donde se condujo un análisis preliminar exploratorio. No se cumplieron los supuestos estadísticos recomendados para realizar pruebas paramétricas (Kuanli, Pavur, y Keeling, 2006). Las variables eran de tipo ordinal y no tenían una distribución normal; por estas razones, se realizaron varias pruebas



Tabla 3. Variables de este estudio y sus definiciones (Dimensión "Percepción sobre la metodología durante la intervención").

Dimensión	VARIABLES	Definición conceptual	Definición operacional y Pregunta
Percepción sobre la metodología durante la intervención	Pertinencia para el mundo empresarial.	Grado en el que la metodología puede ser usada en escenarios reales del mundo empresarial.	Valoración de la pertinencia de la metodología al mundo empresarial (P1).
	Facilidad para trabajar.	Grado en el que la metodología facilita el trabajo de los participantes.	Valoración de la facilidad para trabajar con la metodología (P2).
	Preferencia de utilización.	Grado en el que los participantes eligen esta metodología sobre otras.	Valoración de la preferencia de utilizar esta metodología (P3).
	Tiempo de desarrollo.	Grado en el que la metodología facilita la reducción del tiempo de desarrollo.	Valoración sobre el tiempo de desarrollo invertido en el proyecto (P6).
	Detección de errores.	Grado en el que la metodología facilita la detección de errores en el software.	Valoración sobre la cantidad de errores detectados durante el proceso (P7).
	Calidad del software.	Grado en el que la metodología facilita la obtención de un software concordante con los requerimientos.	Valoración de la calidad del software final (P8).
	Uso recurrente.	Grado en el que los participantes buscarán utilizar nuevamente la metodología.	Probabilidad de volver a utilizar la metodología (P13).
	Percepción global de la experiencia.	Valoración global de la vivencia en el ejercicio.	Valoración de la experiencia de trabajo (P15).
	Satisfacción con el programa realizado.	Grado en el que los participantes están satisfechos con el software que desarrollaron.	Valoración del nivel de satisfacción con el producto final (P16).
	Estrés provocado por la metodología.	Nivel de tensión nerviosa experimentado por los participantes.	Valoración del nivel de estrés experimentado durante el desarrollo (P17).

Tabla 4. Variables de este estudio y sus definiciones (Dimensiones "Preferencias personales sobre el desarrollo de software" y "Percepciones generales sobre el desarrollo de software en equipo").

Dimensión	VARIABLES	Definición conceptual	Definición operacional y Pregunta
Preferencias personales sobre el desarrollo de software	Comodidad al trabajar en pareja.	Grado en el que los participantes se sienten cómodos al desarrollar software con alguien más.	Valoración de la preferencia para trabajar en pareja (P4).
	Motivación para programar en parejas.	Grado en el que los participantes están motivados para realizar programación por pares.	Valoración del nivel de motivación para programar por pares (P11).
	Gusto por la programación.	Nivel de agrado que tienen los participantes hacia la programación.	Valoración del agrado percibido hacia la programación (P14).
Percepciones generales sobre el desarrollo de software en equipo	Importancia de aprender a desarrollar software colaborativamente.	Grado en el que los participantes perciben la importancia de aprender a desarrollar software en equipo.	Valoración de la importancia de aprender a desarrollar software en equipo (P5).
	Confianza al programar en pares	Grado en el que los participantes perciben la confianza de los programadores que trabajan en parejas.	Valoración de la confianza cuando se programa en parejas (P9).
	Facilidad para programar en pares.	Grado en el que los participantes perciben la facilidad de programar en parejas.	Valoración del nivel de facilidad para programar en parejas (P10).
	Satisfacción de los programadores por pares.	Grado en el que los participantes perciben la satisfacción general de los programadores en parejas.	Valoración del nivel de satisfacción de quienes programan por pares (P12).

Tabla 5. Descripción del programa y ejemplo de su ejecución que fue presentado a los alumnos.

<p><b>Descripción del programa:</b>                  Realizar un programa que solicite al usuario un número y realice las siguientes operaciones con él:</p> <ol style="list-style-type: none"> <li>1) Sumarle el mismo número.</li> <li>2) Multiplicarlo por el mismo número.</li> <li>3) Dividirlo entre (el mismo número más 1).</li> <li>4) Restarle (el mismo número menos 1).</li> </ol> <p>Mostrar los resultados de cada operación, además de la suma de todos los resultados anteriores (incluyendo el número capturado).</p> <p>Si la suma de todo es menor a 30, mostrar un mensaje indicando que el número es muy pequeño. Si la suma es mayor a 50, mostrar un mensaje de que el número es muy grande. Por último almacenar de algún modo la suma de todas las cantidades para posteriormente listar la bitácora de las operaciones realizadas con la fecha y hora en que se realizaron.</p>	<p><b>Ejemplo de la ejecución del programa:</b></p> <p>Número introducido: <u>  4  </u></p> <p>Suma: <math>4 + 4 = 8</math></p> <p>Multiplicación: <math>4 * 4 = 16</math></p> <p>División: <math>4 / (4 + 1) = 0.8</math></p> <p>Resta: <math>4 - (4 - 1) = 1</math></p> <p>Suma Total =29.8</p> <p>“La suma final es un número muy pequeño”</p> <table> <tr> <td>01/08/2016</td> <td>6:16</td> <td>Suma total: 19.75</td> </tr> <tr> <td>01/08/2016</td> <td>6:17</td> <td>Suma total: 41.83</td> </tr> <tr> <td>01/08/2016</td> <td>6:18</td> <td>Suma total: 29.8</td> </tr> </table>	01/08/2016	6:16	Suma total: 19.75	01/08/2016	6:17	Suma total: 41.83	01/08/2016	6:18	Suma total: 29.8
01/08/2016	6:16	Suma total: 19.75								
01/08/2016	6:17	Suma total: 41.83								
01/08/2016	6:18	Suma total: 29.8								

no paramétricas de Mann Whitney para buscar diferencias entre grupos de respuestas. En todos los casos se tomó como referencia el 95% de confianza para concluir resultados estadísticamente significativos.

Primero se compararon las respuestas de los alumnos que programaron en solitario con las de los alumnos que programaron por pares en cada una de las preguntas del cuestionario, para lo cual se plantearon las hipótesis indicadas por el conjunto de pruebas número 1 de la Tabla 6.

Posteriormente, la atención se centró únicamente en los participantes que programaron por pares y se buscaron diferencias en relación a su grado de avance en la carrera y a su experiencia en materias de progra-

mación cursadas antes de ingresar a la universidad. Ésto se realizó para cada una de las preguntas del cuestionario, para lo cual se plantearon las hipótesis indicadas por los conjuntos de pruebas 2 y 3 de la Tabla 6.

Para los resultados significativos, se obtuvo el tamaño del efecto  $r$ , dividiendo el valor de  $Z$  entre la raíz cuadrada del número total de datos de ambos grupos (Tomczak y Tomczak, 2014). Estos resultados se comprobaron con la calculadora en línea disponible en (AI-Therapy.com, 2016). El tamaño del efecto se presenta en los siguientes apartados con el valor absoluto de  $r$ , y puede interpretarse de igual manera que el coeficiente de correlación de Pearson. Un acercamiento a esta interpretación, basada en los

Tabla 6. Hipótesis planteadas en la investigación.

Conjunto de pruebas	Hipótesis nula	Hipótesis de investigación
1	$H_0$ : No existen diferencias estadísticas significativas entre las respuestas a la pregunta x que fueron proporcionadas por los alumnos que programaron individualmente y los alumnos que programaron por pares.	$H_a$ : Existen diferencias estadísticas significativas entre las respuestas a la pregunta x que fueron proporcionadas por los alumnos que programaron individualmente y los alumnos que programaron por pares.
2	$H_0$ : No existen diferencias estadísticas significativas entre las respuestas a la pregunta x que fueron proporcionadas por los alumnos programadores en parejas que tuvieron experiencia preuniversitaria en programación y aquellos que no la tuvieron.	$H_a$ : Existen diferencias estadísticas significativas entre las respuestas a la pregunta x que fueron proporcionadas por los alumnos programadores en parejas que tuvieron experiencia preuniversitaria en programación y aquellos que no la tuvieron.
3	$H_0$ : No existen diferencias estadísticas significativas entre las respuestas a la pregunta x que fueron proporcionadas por los alumnos programadores en parejas de los primeros semestres y de los últimos semestres.	$H_a$ : Existen diferencias estadísticas significativas entre las respuestas a la pregunta x que fueron proporcionadas por los alumnos programadores en parejas de los primeros semestres y de los últimos semestres.

Pregunta x: Cualquier pregunta de la 1 a la 17 del cuestionario de la tabla 2



criterios de Hernández, Fernández y Baptista (2014) es: de 0 a 0.25, débil; de 0.25 a 0.50, media; de 0.50 a 0.75, considerable; de 0.75 a 1, muy fuerte.

## Resultados

### Comparación de la Programación por pares y Programación individual

En la Tabla 7 se muestran los resultados significativos de la prueba no paramétrica de Mann Whitney realizada a las respuestas obtenidas por todos los participantes (n=94). Se observaron diferencias en los siguientes aspectos: facilidad de trabajar con la metodología (U=787.5, Z=-2.48, p=.013, r=.26), detección rápida de errores (U=829.5, Z=-2.08, p=.037, r=.21) y satisfacción con el programa obtenido (U=597.0, Z=-2.56, p=.010, r=.28). Los alumnos que programaron por pares percibieron que les resultó más fácil trabajar en parejas que los alumnos que programaron individualmente. También expresaron que detectaron errores más rápidamente que los alumnos que programaron solos, así como también estuvieron más satisfechos con el programa que desarrollaron en comparación con los alumnos que programaron de manera individual.

### Análisis de las respuestas de los participantes que programaron por pares

La prueba no paramétrica de Mann Whitney realizada a las respuestas obtenidas únicamente por los participan-

tes que programaron en parejas (n=54), categorizadas de acuerdo a su experiencia en cursos de programación previos al ingreso a la universidad (Tabla 8), reveló diferencias en cuanto a la preferencia personal para trabajar solo o en parejas (U=146.0, Z=-2.48, p=.013, r=.35). Los estudiantes que cursaron materias de programación antes de ingresar a su carrera universitaria expresaron sentirse más cómodos trabajando solos que los estudiantes que no habían cursado materias de programación al entrar a la universidad.

Al realizar la prueba de Mann Whitney a las respuestas obtenidas únicamente por los participantes que programaron en parejas (n=54), categorizadas de acuerdo al nivel de avance en su carrera profesional, se identificaron diferencias significativas en relación a: la detección rápida de errores (U=224.0, Z=-2.16, p=.031, r=.29), la satisfacción de la experiencia de programar por pares (U=169.5, Z=-2.05, p=.040, r=.29), la satisfacción con el programa obtenido (U=152.0, Z=-2.44, p=.014, r=.35) y el nivel de estrés experimentado (U=167, Z=-2.20, p=.028, r=.31). Estos resultados se muestran en la Tabla 9, en donde puede observarse que los alumnos de semestres avanzados (quinto y séptimo) estuvieron más de acuerdo en haber detectado errores rápidamente al programar por pares que los alumnos de los primeros semestres (primero y tercero). También los alumnos avanzados calificaron la experiencia con puntuaciones más altas y se mostraron más satisfechos con el

Tabla 7. Resultados significativos de la Prueba de Mann-Whitney para comparar la programación por pares y la programación en solitario.

Planteamiento	Resultado de la prueba	Rango promedio Programación en solitario	Rango promedio Programación por pares
Resultó fácil trabajar con la metodología utilizada	U=787.5, Z=-2.48, p=.013, r=.26	40.19 (n=40)	52.92 (n=54)
Se detectaron errores rápidamente con la metodología utilizada.	U=829.5, Z=-2.08, p=.037, r=.21	41.24 (n=40)	52.14 (n=54)
¿Qué tan satisfecho está con el programa obtenido?	U=597.0, Z=-2.56, p=.010, r=.28	35.08 (n=36)	48.82 (n=49)

Tabla 8. Resultados significativos de la Prueba de Mann-Whitney para comparar las respuestas obtenidas por los participantes que programaron por pares de acuerdo a su experiencia preuniversitaria en programación.

Planteamiento	Resultado de la prueba	Rango promedio (Sí cursaron materias de programación antes de la universidad)	Rango promedio (No cursaron materias de programación antes de la universidad)
Me es más cómodo trabajar solo	U=146.0, Z=-2.48, p=.013, r=.35	32.27 (n=15)	21.79 (n=34)

programa desarrollado que los alumnos de primer y tercer semestre. Por otra parte, los alumnos de los primeros semestres registraron niveles de estrés más altos que los alumnos de los últimos semestres.

## Discusión

Un panorama completo de los resultados abordados en este artículo se muestra en la Tabla 10.

De acuerdo a esta evaluación realizada a ambos enfoques, la programación por pares podría ser una excelente alternativa para que los alumnos desarrollen sus proyectos de programación en los cursos universitarios, ya que fue mejor evaluada por los participantes en relación a la facilidad para trabajar con ella, la detección rápida de errores y la satisfacción con el programa obtenido.

Al profundizar en las respuestas de los alumnos que programaron por pares, se encontró que este enfoque de desarrollo resultaría más adecuado para los alumnos de los últimos semestres de la carrera ya que fueron ellos quienes se mostraron más satisfechos con la experiencia de trabajar en parejas y con el programa que obtuvieron. También fueron ellos quienes reportaron en mayor grado la detección rápida de errores y presentaron niveles de estrés más bajos en comparación con los alumnos de los primeros semestres.

Por otra parte, también se encontró que los estudiantes que trabajaron en parejas y que habían cursado materias de programación antes de comenzar su carrera profesional manifestaron sentirse más cómodos para trabajar de manera individual que aquellos

Tabla 9. Resultados significativos de la Prueba de Mann-Whitney para comparar las respuestas obtenidas por los participantes que programaron por pares agrupadas categorizadas de acuerdo al nivel de avance en su carrera profesional.

Planteamiento	Resultado de la prueba	Rango promedio (Materias básicas de programación. Primeros semestres de la carrera: primero y tercero)	Rango promedio (Materias avanzadas de programación. Últimos semestres de la carrera: quinto y séptimo)
Se detectaron errores rápidamente	U=224.0, Z=-2.16, p=.031, r=.29	24.40 (n=35)	33.21 (n=19)
Satisfacción de la experiencia de programar por pares	U=169.5, Z=-2.05, p=.040, r=.29	22.14 (n=33)	30.91 (n=16)
Satisfacción con el programa obtenido	U=152.0, Z=-2.44, p=.014, r=.35	21.61 (n=33)	32 (n=16)
Nivel de estrés durante la programación por pares.	U=167, Z=-2.20, p=.028, r=.31	28.59 (n=34)	18.94 (n=16)

Tabla 10. Resumen de resultados estadísticamente significativos presentados en este trabajo.

Aspecto evaluado	Puntuaciones más bajas	Puntuaciones más altas
<b>Resultados de todos los participantes</b>		
Facilidad de trabajar	Programación individual	Programación por pares
Satisfacción con el programa desarrollado	Programación individual	Programación por pares
Detección rápida de errores	Programación individual	Programación por pares
<b>Resultados de alumnos que programaron por pares</b>		
Detección rápida de errores	Primeros semestres de la carrera	Últimos semestres de la carrera
Satisfacción de la experiencia de programar por pares	Primeros semestres de la carrera	Últimos semestres de la carrera
Satisfacción con el programa obtenido	Primeros semestres de la carrera	Últimos semestres de la carrera
Nivel de estrés durante la programación por pares.	Últimos semestres de la carrera	Primeros semestres de la carrera
Comodidad para trabajar individualmente	Estudiantes que nunca habían cursado materias de programación antes de ingresar a la universidad	Estudiantes que cursaron materias de programación antes de ingresar a la universidad

que cursaron materias de programación por primera vez en la universidad. Es posible como consecuencia de esta experiencia preuniversitaria, que ellos hayan tenido que adoptar un rol de guía o tutor en el equipo en el que trabajaron. Esto podría generar en ellos la percepción de que sería más cómodo trabajar individualmente, pues así no tendrían que asesorar, enseñar o tutorar a alguien más. Tomando esto en consideración, al inicio de cualquier curso universitario en donde se adopte la programación por pares resultaría relevante identificar a los alumnos con experiencia previa para focalizar en ellos mecanismos de motivación para realizar trabajo colaborativo, así como también considerarlos estratégicamente en la organización de parejas.


Los resultados de la presente investigación sugieren que la programación por pares puede ser favorable si se adopta en el entorno universitario pues promueve el trabajo en equipo y produce una buena percepción del proceso y del producto final entre los participantes. Sin embargo, se debe cuidar el perfil de los alumnos con quienes se trabaja de esta manera. A pesar de esto, es importante reflexionar que las prácticas de clase en la universidad pueden ser aprovechadas para que los alumnos experimenten los enfoques ágiles, ya que les brindan un panorama actualizado del desarrollo real de aplicaciones. Estos hallazgos son concordantes con las conclusiones de Prabu y Duraisami (2015), de Kropp y Meier (2013) y de Schroeder et al. (2012).

Por otra parte, es importante tomar en cuenta que al introducir una nueva metodología hay un periodo de adaptación entre los participantes. Para esta investigación, los estudiantes tuvieron su primer encuentro práctico con la programación por pares a través del ejercicio de desarrollo de software descrito en este trabajo, lo que podría afectar su percepción acerca de esta experiencia. Sin embargo, los resultados proporcionan una primera perspectiva orientadora y motivante para continuar la investigación en esta línea, ya que se han encontrado posturas favorables hacia la programación por pares aún en el periodo inicial de asimilación de la metodología.

También es importante destacar que esta investigación se realizó en un ámbito académico, en el cual todos los estudiantes, sin importar si trabajan solos o con una pareja, deben dedicar tiempo a sus

proyectos escolares para cumplir con sus deberes oportunamente. En este escenario, no existe un costo asociado a la cantidad de personas que trabajan simultáneamente en un mismo proyecto, es decir, al total de horas-hombre requeridas. Pero, incluso si ese costo existiera, trabajar en parejas probablemente no representaría una inversión significativamente mayor, ya que existen estudios que indican que la programación por pares produce mejores programas en menos tiempo que la programación individual (Gómez, Batún, y Aguilar, 2013).

## Conclusiones

En este artículo se presentaron los resultados de una investigación que comparó la percepción de la programación por pares y la programación individual en el entorno universitario. Al analizar los resultados de las pruebas estadísticas de Mann Whitney que se aplicaron, se encontró que la programación por pares obtuvo mejores evaluaciones en relación a la facilidad para trabajar, detección de errores y la satisfacción con el programa realizado. Sin embargo, este enfoque podría ser más adecuado para los alumnos de los últimos semestres y menos recomendable para los alumnos de los primeros semestres: al programar por pares, se observó que los alumnos de los últimos semestres tuvieron una mejor experiencia en este ejercicio y presentaron menores niveles de estrés que los alumnos principiantes. Por otra parte, los resultados revelaron que los alumnos que trabajaron en parejas y que habían cursado materias de programación preuniversitarias tienen mayor preferencia por trabajar solos. De esta manera, los alumnos con experiencia preuniversitaria deberían recibir motivación para interactuar académicamente con otros estudiantes y deben considerarse estratégicamente en la organización de las parejas que se formen para programar 

## Referencias

- Agile Alliance. (01 de marzo de 2015). *Manifiesto por el Desarrollo Ágil de Software*. Obtenido de <http://www.agilemanifesto.org/>
- AITherapy.com. (2016). *AI-Therapy Statistics*. Obtenido de Comparing two sets of data: <https://www.ai-therapy.com/psychology-statistics/hypothesis-testing/two-samples?groups=0&parametric=1>

- Beck, K., & Andres, C. (2004). *eXtreme Programming explained. Embrace Change*. Estados Unidos: Addison Wesley.
- Coman, I. D., Robillard, P. N., Silliti, A., and Succi, G. (2014). Cooperation, collaboration and pair-programming: Field studies on backup behavior. *The Journal of Systems and Software*, 124-134.
- Coss Bu, R. (1995). *Simulación: Un enfoque práctico*. Cd. de México, México: Limusa Noriega Editores.
- Gómez, O., Batún, J., & Aguilar, R. (2013). Pair versus solo programming – An experience report from a course on design of experiments in software engineering. *International Journal of Computer Science Issues*, 10(1), 734-742.
- Hernández, S. R., Fernández Collado, C., & Baptista Lucio, P. (2014). *Metodología de la investigación*. Ciudad de México: McGrawHill.
- Juristo, N., & M. Moreno, A. (2001). *Basics of Software Engineering Experimentation*. The Netherlands: Springer.
- Kendall, K., & Kendall, J. (2013). *Systems Analysis and Design*. United States: Prentice Hall.
- Kropp, M., & Meier, A. (2013). Teaching Agile Software Development at University Level. *IMVS Fokus Report*, 15-20.
- Kuanli, Pavur, & Keeling. (2006). *Introduction to Business Statistics*. Estados Unidos: Thomson South Western.
- McDowell, C., Werner, L., Bullock, H., and Fernald, J. (2006). Pair Programming Improves Student Retention, Confidence, and Program Quality. *Communications of the ACM*, 49(8), 90-95.
- Niño Rojas, V. (2011). *Metodología de la investigación*. Colombia: Ediciones de la U.
- Plonka, L., Sharp, H., van der Linden, J., and Dittrich, Y. (2015). Knowledge transfer in pair programming: An in-depth analysis. *International Journal of Human-Computer Studies*, 66-78.
- Prabu, P., & Duraisami, S. (2015). Impact of Pair Programming for Effective Software Development Process. *International Journal of Applied Engineering Research*, 10(8), 18969-18986.
- Pressman, R. & Maxim, B. (2014). *Software Engineering: A Practitioner's approach* (8 ed.). United States: McGrawHill.
- Rizwan, M., & Qureshi, J. (2011). Agile software development methodology for medium and large projects. *IET Software*, 6(4), 358-363.
- Roque Hernández, R. V., Salinas Escandón, J. M., López Mendoza, A., Mota Martínez, S., & Flores Rosales, O. (2014). Análisis de proyectos de Programación Orientada a Objetos utilizando métricas de software como medio para determinar necesidades didácticas. *Investigación y Ciencia*, 61, 20-26.
- Salkind, N. (1999). *Métodos de investigación*. Ciudad de México: Pearson.
- Salleh, N., Mendes, E., and Grundy, J. (2014). Investigating the effects of personality traits on pair programming in a higher education setting through a family of experiments. *Empirical Software Engineering*, 714-752.
- Schroeder, A., Klarl, A., Mayer, P., and Kroib, C. (2012). Teaching agile software development through lab courses. *Global Engineering Education Conference (EDUCON) 2012* (págs. 1177-1186). Marrakech: IEEE.
- Singh, G., Mishra, A., Singh, H., & Upadhyay, P. (2015). Empirical study of agile software development methodologies: a comparative analysis. *ACM SIGSOFT Software Engineering Notes*, 40(1).
- Swamidurai, R., and Umprass, D. A. (2012). Collaborative-adversarial pair programming. *International Scholarly Research Network Software Engineering*.
- Tomczak, M., & Tomczak, E. (2014). The need to report effect size estimates revisited. An overview of some recommended measures of effect size. *Trends in Sport Sciences*, 1(21), 19-25.
- UAT, B. (2015). *Campus en Línea de la Universidad Autónoma de Tamaulipas*. Ciudad Victoria, Tamaulipas: UAT.
- Wagner, W. (2014). *Using IBM SPSS Statistics for Research Methods and Social Science Statistics*. SAGE Publications.
- Zhang, X., Zhang, C., Stafford, T. F., & Zhang, P. (2013). Teaching introductory programming to IS students: the impact of teaching approaches on learning performance. *Journal of information systems education*, 24(2), 147-155.