Docencia en Sistemas de Acceso á Información: detección de plaxios, emprego de tecnoloxías avanzadas para desenvolvemento software e achegamento da experiencia na industria á aula

Lopez-Otero, Paula; Valcarce, Daniel; Parapar, Javier; Barreiro, Alvaro

Departamento de Computación, Facultade de Informática, Universidade da Coruña.

RESUMO

Este artigo presenta as actividades desenvolvidas polo grupo de innovación educativa en Sistemas de Acceso á Información durante o curso 2017/2018. Este grupo, con docencia na Facultade de Informática da Universidade da Coruña, realizou accións en tres liñas de actuación diferentes. A primeira delas, dirixida á mellora da calidade nos métodos de avaliación, consiste no emprego dun protocolo para a detección de plaxios en prácticas de programación. A segunda actividade pretende mellorar a empregabilidade do alumnado e consiste en utilizar unha metodoloxía de aprendizaxe baseada en proxectos xunto cunha serie de ferramentas avanzadas para desenvolvemento software, permitindo recrear a actividade que deberán levar a cabo cando se incorporen ao mundo laboral. Por último, e de cara a aumentar o coñecemento das alternativas profesionais do alumnado, organizáronse unha serie de seminarios e charlas impartidas por profesionais dunha empresa internacional, unha empresa local multidisciplinar e un investigador da contorna académica. A experiencia obtida das diferentes actividades foi satisfactoria e enriquecedora tanto para o alumnado como para o profesorado, que xa baralla melloras de cara aos vindeiros cursos académicos.

PALABRAS CLAVE: detección de plaxio, ferramentas de desenvolvemento, aprendixaxe baseada en proxectos, profesionais na aula.

CITA RECOMENDADA:

López-Otero, P.; Valcarce, D.; Parapar, J; Barreiro, A. (2019): Docencia en Sistemas de Acceso á Información: detección de plaxios, emprego de tecnoloxías avanzadas para desenvolvemento software e achegamento da experiencia na industria á aula. En De la Torre Fernández, E. (ed.) (2019). *Contextos universitarios transformadores: construíndo espazos de aprendizaxe. III Xornadas de Innovación Docente*. Cufie. Universidade da Coruña. A Coruña (pág. 41-56).

DOI capítulo: https://doi.org/10.17979/spudc.9788497497121.041

DOI libro: https://doi.org/10.17979/spudc.9788497497121

ABSTRACT

This paper presents the activities performed by the educative innovation group in Information Access Systems during the academic year 2017/2018. This group, with teaching at the Faculty of Informatics of the University of A Coruña, carried out actions addressing three different topics. The first action was designed to improve the quality of the evaluation methods, and consisted in following a protocol for detecting plagiarism in programming exercises. The second activity aimed to improve the employability of the students and consisted in using a methodology based on project-based learning along with a series of advanced tools for software development, which recreated the activity that the students will carry out when they obtain their first job. Lastly, heading towards a better knowledge about the available professional alternatives, a series of seminars and talks were organized, which were performed by professionals from an international company, a local interdisciplinary company, and a researcher from an academic institution. The experience obtained from the different activities was satisfactory for both students and teachers, who are already considering improvements for the next academic years.

KEY WORDS: plagiarism detection, development tools, project-based learning, professionals in the classroom.

1. INTRODUCIÓN

Este artigo describe as tarefas desenvolvidas polo Grupo de Innovación Educativa en Sistemas de Acceso á Información (GIE-SAI) durante o curso académico 2017/2018. Os principais obxectivos destas accións son a mellora dos métodos de avaliación e da empregabilidade do alumnado nas materias impartidas polo GIE-SAI.

A primeira das actividades que se poden destacar é o emprego dun detector automático de plaxios para a materia Sistemas Operativos. Nestas materias os estudantes deben realizar unha serie de exercicios prácticos de programación, e é habitual que caian na tentación de plaxiar o traballo dos compañeiros por diferentes motivos como a presión por non chegar a tempo para a data límite dunha entrega, ou porque non queren realizar o esforzo necesario para completar a tarefa asignada polos profesores da materia (Clough, 2000). Isto, sumado á percepción de que o plaxio non é un comportamento eticamente reprobable exhibido por unha cantidade representativa de estudantes, segundo algúns estudos (Joy & Luck, 1999; Aasheim et al., 2012), crea a necesidade de tratar de evitar este comportamento enganoso para non xerar desigualdades de cara á avaliación dos estudantes. Porén, evitar estes plaxios non é unha tarefa sinxela, xa que nalgúns casos o número de alumnos é moi elevado (por exemplo, hai case 300 alumnos matriculados na materia de Sistemas Operativos no segundo curso do Grao en Enxeñaría Informática) e comparar todos os pares de prácticas manualmente é inviable. Por este motivo, o desenvolvemento de sistemas automáticos de detección de plaxios en software é, a día de hoxe, un tema de investigación aberto (Luo et al., 2017; Mirza et al., 2017; Tian et al., 2018). Neste artigo describimos a ferramenta que empregamos para realizar esta detección de xeito automático, así como o protocolo seguido polo profesorado do GIE-SAI para obter o maior rendemento na detección de plaxios.

A segunda actividade está destinada á mellora da empregabilidade do alumnado mediante o uso de ferramentas avanzadas para desenvolvemento software. Esta actividade, que se levou a cabo na materia Ferramentas de Desenvolvemento (Grao en Enxeñaría Informática), consiste no emprego de ferramentas de desenvolvemento software amplamente empregadas na

empresa. Especificamente, empregáronse ferramentas para xestión de proxectos, integración continua e inspección continua. A experiencia de empregar ferramentas colaborativas para xestión de proxectos adoita ter un impacto positivo entre os alumnos, que valoran positivamente a posibilidade de complementar as reunións presenciais con outras virtuais, así como a posibilidade de interactuar co resto do equipo sempre que o necesiten (Medeiros et al., 2017). De acordo con algúns estudos, o emprego de técnicas profesionais de desenvolvemento software axuda os estudantes a comprender a necesidade e o uso da integración continua (Eddy et al., 2017), e esta reduce o salto entre as prácticas na aula e a programación profesional (Süß & Billingsley, 2012). No relativo á inspección continua, o uso destas ferramentas resulta de gran axuda de cara a desenvolver software de calidade ao permitir visualizar unha análise dos problemas encontrados no código (Andrade Gomes et al., 2017) . O uso destas ferramentas integrouse nunha metodoloxía de aprendizaxe baseada en proxectos: esta metodoloxía adoita ter moito éxito entre os estudantes, pois consideran que aprenden máis cando traballan en grupo e ademais teñen certa preferencia por este formato máis dinámico e interactivo (Caceffo et al., 2018; Yagci, 2018). A metodoloxía proposta polo GIE-SAI permite que os estudantes aprendan a empregar tecnoloxías profesionais para desenvolveren software de calidade, o que favorece directamente a súa empregabilidade.

Por último, a terceira actividade tivo como obxectivo ampliar o coñecemento do mundo laboral dos estudantes das materias Recuperación da Información (Grao en Enxeñaría Informática), Recuperación da Información e Web Semántica (Mestrado Universitario en Enxeñaría Informática) e Xestión de Coñecemento Biomédico (Mestrado Universitario en Bioinformática para Ciencias da Saúde). Con este propósito, organizáronse seminarios e conferencias impartidos por profesionais do sector: este tipo de actividades axudan a atopar un equilibro entre a teoría e a realidade do mundo laboral (Callahan and Pedigo, 2002). De cara a ofrecer aos estudantes unha visión o ampla das diferentes alternativas do mundo laboral, os seminarios foron impartidos por traballadores de diferentes eidos: a empresa internacional, a empresa local e o mundo académico.

2. DETECCIÓN DE PLAXIOS

Como se mencionou na Sección 1, empregouse un detector de plaxios na materia Sistemas Operativos para tratar de detectar posibles casos de copia entre alumnos. Nesta materia, os alumnos debían realizar unhas prácticas de programación na linguaxes C, dirixidas a afianzar os coñecementos adquiridos nas clases teóricas.

A detección de plaxios, tanto en traballos escritos como en software, é unha tarefa complexa que está a ser obxecto de investigación nas últimas décadas (Clough et al., 2000; Flores et al., 2011; Luo et al., 2017; Mirza et al., 2017). Das ferramentas dispoñibles, a empregada polo GIE-SAI é Moss[i] (*A Measure Of Sofware Similarity*), desenvolvida pola Universidade de Stanford (Schleimer et al., 2003). Moss é unha ferramenta gratuíta e está dispoñible como servizo web, de xeito que o usuario simplemente ten que descargar un pequeno código que lle permite acceder ao servizo.

Moss Results

Sat Nov 17 03:04:59 PST 2018

Options -l c -m 10

[How to Read the Results | Tips | FAQ | Contact | Submission Scripts | Credits]

File 1	File 2	Lines Matched
[redacted]/all-P2-so-18-19.c (93%)	[redacted]/all-P2-so-18-19.c (60%)	1388
[redacted]/all-P2-so-18-19.c (96%)	[redacted]/all-P2-so-18-19.c (96%)	1100
[redacted]/all-P2-so-18-19.c (82%)	[redacted]/all-P2-so-18-19.c (74%)	1193
[redacted]/all-P2-so-18-19.c (28%)	[redacted]/all-P2-so-18-19.c (30%)	272
[redacted]/all-P2-so-18-19.c (19%)	[redacted]/all-P2-so-18-19.c (19%)	219
[redacted]/all-P2-so-18-19.c (15%)	[redacted]/all-P2-so-18-19.c (13%)	172
[redacted]/all-P2-so-18-19.c (14%)	[redacted]/all-P2-so-18-19.c (11%)	159
[redacted]/all-P2-so-18-19.c (12%)	[redacted]/all-P2-so-18-19.c (13%)	156
[redacted]/all-P2-so-18-19.c (12%)	[redacted]/all-P2-so-18-19.c (10%)	129
[redacted]/all-P2-so-18-19.c (12%)	[redacted]/all-P2-so-18-19.c (11%)	148
[redacted]/all-P2-so-18-19.c (9%)	[redacted]/all-P2-so-18-19.c (7%)	97
[redacted]/all-P2-so-18-19.c (6%)	[redacted]/all-P2-so-18-19.c (8%)	132
[redacted]/all-P2-so-18-19.c (7%)	[redacted]/all-P2-so-18-19.c (6%)	109

Moss ten unha serie de parámetros configurables, entre os que podemos destacar a linguaxe de programación, ou o número de veces que se debe repetir un fragmento de código en diferentes prácticas para deixar de consideralo no cálculo da similitude (por exemplo, se se lles subministra aos alumnos algunha función para empregaren no seu código, esta vai ser igual en todos os programas e non debe ser considerada).

Os profesores da materia Sistemas Operativos estableceron un protocolo a seguir para detectar posibles plaxios entre os alumnos. Primeiro, os alumnos deben subir o código das súas prácticas ao servidor SVN da Facultade de Informática, coa finalidade de asegurar que todos os pares de prácticas serán comparados entre si. Despois, emprégase a ferramenta Moss para realizar unha comparación de todos os pares de prácticas: esta devolve unha listaxe ordenada como a mostrada na Figura 1, na que aparecen os pares de prácticas xunto coa súa correspondente porcentaxe de similitude. O profesorado estableceu que todos aqueles pares de prácticas cunha porcentaxe de similitude superior ao 10% son sospeitosas de ser plaxios. Porén, Moss detecta que un par de códigos son similares, pero non é quen de saber por que. Por exemplo, os dous primeiros pares de prácticas mostrados na Figura 1 constitúen falsos positivos, xa que a alta similitude débese a que os dous alumnos da parella subiron a práctica ao servidor SVN cando só debería tela subido un deles. Polo tanto, o profesorado debe realizar unha revisión manual dos pares de prácticas con alta similitude para avaliar se o parecido débese a plaxio ou a outras situacións como a mencionada. As Figuras 2 e 3 mostran dous códigos reais entregados por alumnos da materia Sistemas Operativos durante o curso 2018/2019. As prácticas destes alumnos resultaron sospeitosas de copia por teren unha similitude do 43% segundo a ferramenta Moss. A inspección manual levada a cabo polos profesores da materia concluíu que esta similitude debíase a un caso de plaxio por diferentes motivos:

- A estrutura de ambos códigos é idéntica, a diferencia máis relevante entre eles é o nome das variables (*listado_corto* fronte a *corto*, *listar_ocultos* fronte a *ocultos*, etc.).

```
void muestraLinea(int listado_corto, int listar_ocultos, int modo_recursivo, char* nombre){

        struct stat buf;
        char info[MAX_ENTRADA];
        char completo[MAX_ENTRADA];
        DIR *contenido;
        struct dirent *entrada;

        if(lstat(nombre,&buf)){
                printf("Error con peticion de la entrada:%s\n", nombre);
                perror("Error al ejecutar stat");
                return;
        }

        if(S_ISDIR(buf.st_mode)){
                if((contenido = opendir(nombre)) == NULL){
                        perror("No se puede abrir el directorio");
                        return;
                }
                while( (entrada=readdir(contenido)) != NULL){
                        if(entrada->d_name[0] != '.' || listar_ocultos){
                                completo[0] = '\0';
                                strcat(completo,nombre);
                                strcat(completo,"/");
                                strcat(completo,entrada->d_name);
                                printf("%s\n", info_entrada(completo, listado_corto, info));
                        }
                }
```

Figura 2: Fragmento de código do alumno A sospeitoso de copia.

```
void muestraLinea(int corto, int ocultos, int recursivo, char* nombre){

        struct stat buf; //Estructura de tipo stat
        char salida[MAX_ENTRADA];
        char completo[MAX_ENTRADA];
        DIR *contenido; //Puntero a directorio.
        struct dirent *info;    //Puntero a una estrucuta de tipo dirent.

        if(lstat(nombre,&buf)){
                printf("Error con peticion de la entrada:%s\n", nombre);
                perror("Error al ejecutar stat");
                return;
        }

        if(S_ISDIR(buf.st_mode)){       //Si es un directorio.
                if((contenido = opendir(nombre)) == NULL){      //Abrimos el directorio para recorrer su conteni
                        perror("No se puede abrir el directorio");
                        return;
                }
                while( (info=readdir(contenido)) != NULL){      //Vamos leyendo cada una de las entradas del dir
                        if(info->d_name[0] != '.' || ocultos){
                                completo[0] = '\0';
                                strcat(completo,nombre);
                                strcat(completo,"/");
                                strcat(completo,info->d_name);
                                printf("%s\n", entrada(completo, corto, salida,info->d_name));
                        }
                }
```

Figura 3: Fragmento de código do alumno B sospeitoso de copia.

- O código da Figura 3 ten comentarios mentres que o da Figura 2 non ten ningún. Neste caso foi o alumno B o que realizou o plaxio, e posiblemente engadiu os comentarios para ter un apoio de cara a responder as preguntas dos profesores.
- A expresión *if(entrada->d_name[0] != '.' || listar_ocultos)*, que aparece en ambos fragmentos de código, é rechamante debido á súa corrección (trátase de alumnos pouco experimentados nesta linguaxe de programación). Ademais, dada a ampla cantidade de alternativas para realizaren a mesma función, é rechamante que dúas persoas teñan optado por exactamente a mesma solución.

Cando o profesorado detecta un posible caso de plaxio, realiza unha entrevista cos estudantes para pedirlles explicacións sobre a alta similitude da súa práctica coa dos compañeiros: en caso de que estes non sexan quen de ofrecer unha explicación razoable para esta alta similitude, reciben un suspenso en todas as prácticas da materia de acordo co establecido nas normas comunicadas ao alumnado no comezo do curso académico.

O emprego da ferramenta de detección de plaxios xunto co protocolo descrito nesta sección proporcionou os resultados esperados: o número de estudantes que se arrisca a copiar é moi reducido porque coñecen a posibilidade de ser descubertos, e ademais axudou a detectar algún caso de fraude nas prácticas de programación, o que contribúe a avaliar os alumnos de xeito máis xusto.

3. TECNOLOXÍAS AVANZADAS PARA DESENVOLVEMENTO SOFTWARE

A medida que pasan os anos xorden novas ferramentas de desenvolvemento software que substitúen a outras xa existentes por seren máis eficientes, melloraren a usabilidade ou teren máis funcionalidades. Polo tanto, na docencia relacionada con desenvolvemento software, é interesante realizar unha revisión anual dos contidos das materias para adaptalos á realidade do mercado laboral en cada momento. No caso da materia Ferramentas de Desenvolvemento do Grao en Enxeñaría Informática, esta actualización dos contidos é crucial, xa que é importante que os alumnos aprendan a desenvolver software de xeito profesional para teren

unha transición menos traumática ao mundo laboral. O GIE-SAI propuxo, en concreto, o emprego de tres ferramentas profesionais que se describen no resto desta sección. Esta actividade foi valorada positivamente polo alumnado, xa que a percibiron como un reforzo das súas competencias en desenvolvemento software que vai supoñer un beneficio directo na súa empregabilidade.

3.1. XESTIÓN DE TAREFAS

Cando se desenvolve un proxecto software é importante dispoñer de ferramentas para xestionar as tarefas e facer un seguimento das incidencias (*issues*). Estas ferramentas permiten levar un control de quen é responsable de cada tarefa, onde está a documentación de cada versión ou que características inclúe unha determinada versión, entre outras funcionalidades.

Redmine[ii] é unha aplicación web de xestión de tarefas. O núcleo de Redmine é a incidencia, que pode ser definida como unha tarefa discreta que implica unha acción. Este tipo de xestión permite dividir un proxecto grande en compoñentes máis pequenos; organizar tarefas e facer un seguimento do progreso; identificar, investigar e resolver *bugs;* e mellorar a comunicación entre os desenvolvedores do equipo mediante un taboleiro de novas, un foro ou un repositorio de documentos. Estas características motivaron o emprego de Redmine na materia Ferramentas de Desenvolvemento, xa que é moi completa e apropiada para o traballo en equipo.

3.2. INTEGRACIÓN CONTINUA

A integración continua é unha práctica de desenvolvemento software na que os membros do equipo integran o seu traballo de xeito frecuente, de maneira que haxa varias integracións diarias. A principal vantaxe desta práctica é que a integración convértese nun proceso máis sinxelo, xa que os cambios intégranse con moita frecuencia. O feito de que todos os membros do equipo fagan integracións sobre o mesmo proxecto obrígalles a estandarizar moitos

procesos do desenvolvemento software. Ademais, permite coñecer en todo momento o estado de saúde do software.

En Ferramentas de Desenvolvemento empregouse a ferramenta Jenkins[iii] para integración continua. Jenkins proporciona estes servizos en forma de *servlet*, é moi sinxelo de configurar, e inclúe diversas ferramentas que permiten estandarizar o proceso software seguido polo equipo de traballo.

3.3. INSPECCIÓN CONTINUA

A inspección continua é un proceso deseñado para facer que a calidade interna do código sexa unha parte integral do ciclo de vida de desenvolvemento software. Esta calidade está relacionada con atributos chave do código tales como a robustez, a conformidade cos estándares e a mantebilidade. O obxectivo da inspección continua é o desenvolvemento software de calidade dende o inicio mediante a realización de iteracións cortas nas que se garante a rápida resolución de problemas de calidade interna.

A ferramenta seleccionada para inspección continua foi SonarQube[iv]. Esta plataforma permite acceder a diferentes análises do código tales como erros de estilo, *bugs* potenciais, defectos do código, ineficiencias no deseño, duplicado de código, falta de cobertura dos tests, ou excesiva complexidade. Todo o que afecta ao código do proxecto, dende pequenos detalles de estilo até erros críticos de deseño, é inspeccionado e avaliado por SonarQube. para integración continua. Jenkins proporciona estes servizos en forma de servlet, é moi sinxelo de configurar, e inclúe diversas ferramentas que permiten estandarizar o proceso software seguido polo equipo de traballo.

4. PROFESIONAIS NA AULA

Como xa se mencionou na Sección 1, levar profesionais á aula permite que os estudantes teñan unha visión máis clara do mundo laboral que se van atopar cando rematen a súa titulación. O GIE-SAI organizou unha serie de charlas e seminarios para o alumnado das súas

materias, tentando abranguer diversos perfís dentro do mundo laboral, tal e como se describe no resto desta sección.

4.1. EMPRESA INTERNACIONAL: ELASTIC

Elastic[v] é unha empresa internacional moi destacada no eido da Recuperación de Información, sendo o seu produto máis popular a ferramenta Elasticsearch, amplamente empregada en contornas industriais. Esta empresa exhibe certas peculiaridades no relativo á súa estrutura, xa que se trata dunha compañía fundada por persoas de diferentes países, con sedes deslocalizadas en diferentes puntos do planeta. Esta estruturación empresarial responde á premisa de que "formar un consenso entre mentes similares é sinxelo, pero acadar este obxectivo cando o equipo de traballo está formado por persoas de diferentes nacionalidades, culturas e tradicións leva a solucións máis robustas e duradeiras no tempo". O seu produto máis popular, a ferramenta Elasticsearch, é un motor de procura de texto baseado na librería Lucene. Dispón dunha REST API que permite realizar consultas nos datos almacenados nos índices de xeito cómodo e flexible.

En vista das características da empresa Elastic, así como a popularidade do seu produto Elasticsearch, resultou interesante convidar ao seu traballador Ismael Hasan, Enxeñeiro Informático e antigo alumno da Universidade da Coruña, a impartir un seminario sobre Elasticsearch aos alumnos de Recuperación da Información e Web Semántica.

4.2. A EMPRESA LOCAL: HEALTH IN CODE

Health In Code[vi] é unha empresa biotecnolóxica, xurdida da Universidade da Coruña e da Universidade de Santiago de Compostela, que se dedica ao desenvolvemento software con aplicacións na clínica e investigación en medicina. Dita empresa, con sede na Coruña, ten actividade internacional e un equipo interdisciplinar dividido en tres áreas: clínica, composta por diversos traballadores do ámbito de ciencias da saúde; laboratorio, con persoal de eidos

relacionados coa bioloxía e a biotecnoloxía; e bioinformática, composta principalmente por enxeñeiros informáticos.

O carácter marcadamente interdisciplinar desta empresa, así como a súa proximidade xeográfica, pareceron de interese para mostrar aos alumnos como é a vida nunha empresa local e de menor tamaño, en contraste con Elastic. Jorge Rodríguez e Pablo Iglesias impartiron unha charla sobre sistemas de xestión de coñecemento biomédico aos alumnos da materia Xestión de Coñecemento Biomédico (Mestrado Universitario en Bioinformática para Ciencias da Saúde), na que aportaron a súa visión como expertos na área. Como engadido á experiencia dos oradores neste campo, é destacable a importancia que tivo para o alumnado escoitar a experiencia de investigadores que traballan nun equipo multidisciplinar, xa que supón unha dificultade engadida á que se poderían ter que enfrontar cando se incorporen ó mundo laboral.

4.3. O MUNDO DA ACADEMIA

Unha saída laboral pouco coñecida entre o alumnado é o mundo da investigación, tanto en institucións públicas como privadas. Polo tanto, considerouse de grande interese convidar a un investigador con certa traxectoria a falarlles aos alumnos sobre a súa investigación. O elixido foi un antigo alumno da Facultade de Informática da Universidade da Coruña, David E. Losada Carril, que ten un posto de profesor titular na Escola Técnica Superior de Enxeñaría Informática da Universidade de Santiago de Compostela. Este investigador é experto no eido da recuperación de información, especialmente en temáticas como a xeración automática de resumos, clasificación de textos, procura de patentes ou minaría de opinións. A charla impartida por este profesor tiña como nome “Aprendizaxe por reforzo para a construción de bancos de proba en Recuperación de Información” e foi impartida, como actividade complementaria e opcional, aos alumnos das materias Recuperación da Información (Grao en Enxeñaría Informática) e Recuperación da Información e Web Semántica (Mestrado Universitario en Enxeñaría Informática).

5. CONCLUSIÓNS E TRABALLO FUTURO

Este artigo describe as actividades levadas a cabo polo GIE-SAI no curso 2017/2018. A experiencia foi moi positiva tanto para o alumnado como para o profesorado, pero aínda hai unha marxe de mellora que tratará de ser abordada nos vindeiros cursos académicos.

O emprego dun detector de plaxios nunha materia con prácticas de programación resultou ser unha efectiva ferramenta de cara a realizarmos unha avaliación máis xusta para todo o alumnado, e como método disuasorio para que os alumnos non consideren a opción de copiar os exercicios de programación. Porén, aínda hai casos de plaxio entre o alumnado. Polo tanto, o obxectivo para os vindeiros cursos é continuar co emprego desta ferramenta para detección de plaxios, mellorar o protocolo establecido e concienciar ao alumnado da importancia de traballar para acadar os seus obxectivos sen recorrer a enganos. Tamén se comezará a empregar esta estratexia na materia Recuperación da Información.

A metodoloxía de aprendizaxe baseada en proxectos incluíndo o uso de ferramentas de desenvolvemento profesionais tivo unha moi boa acollida entre o alumnado, xa que aprenderon a programar tal e como o farán no seu futuro profesional, e ademais esta metodoloxía de traballo ten unha forte compoñente de traballo en equipo, unha competencia moi importante no mundo laboral. No curso 2017/2018 empregáronse ferramentas para xestión de tarefas, inspección continua e integración continua. O obxectivo de cara ao curso 2018/2019 é engadir unha ferramenta de control de versións como GitLab, e dar acceso ao alumnado a unha máquina de despregue de aplicacións.

A organización de charlas e conferencias na aula foi unha experiencia enriquecedora para o alumnado, xa que foron quen de coñecer de primeira man as diferentes formas de traballar nunha empresa internacional, nunha empresa multidisciplinar e na contorna académica. En futuras charlas que se organicen no eido das actividades do GIE-SAI, sería interesante incluír a algún orador que traballe nun centro de investigación privado para completar o abanico de posibilidades laborais. Outro obxectivo a futuro é convidar a mulleres para impartir estes seminarios de cara a dar visibilidade ao papel da muller no eido das TIC e para aportar

referentes femininos ás alumnas dunha carreira maioritariamente masculina como é a Enxeñaría Informática.

6. REFERENCIAS

Aasheim, C. L., Rutner, P. S., Li, L. & Williams, S. R. (2012). Plagiarism and Programming: A Survey of Student Attitudes. *Journal of Information Systems Education*, 23 (3), pp. 297-313.

Andrade Gomes, P. H., Garcia, R. E., Spadon, G., Medeiros Eler, D., Olivete Júnior, C., & Messias Correia, R. C. (2017). Teaching Software Quality via Source Code Inspection Tool. *IEEE Frontiers in Education Conference (FIE)*, pp. 1-8.

Caceffo, R., Gama, G. & Azevedo, R. (2018). Exploring Active Learning Approaches to Computer Science Classes. *Proc. SIGCSE'18*, pp. 922-927.

Callahan, D. & Pedigo, B. (2002). Educating experienced IT professionals by addressing industry's needs. *IEEE Software*, 19 (5), pp. 57-62.

Clough, P. (2000). Plagiarism in natural and programming languages: an overview of current tools and technologies. *Research memoranda: CS-00-05, Department of Computer Science, University of Sheffield, UK*, pp. 1-31. https://ir.shef.ac.uk/cloughie/papers/plagiarism2000.pdf

Eddy, B. P., Wilde, N., Cooper, N. A., Mishra, B., Gamboa, V. S., Shah, K. M., Deleon, A. M. & Shields, N. A. (2017). A Pilot Study on Introducing Continuous Integration and Delivery into Undergraduate Software Engineering Courses. *Proc. 30th IEEE CSEET*, pp. 47-56.

Flores, E., Barrón-Cedeño, A., Rosso, P. & Moreno, L. (2011). Detección de Reutilización de Código Fuente entre Lenguajes de Programación en base a la Frecuencia de términos. *Proc. IV Jornadas PLN-TIMM*, pp. 21-26.

Joy, M. & Luck, M. (1999). Plagiarism in Programming Assignments. *IEEE Transactions on Education*, 42 (2), pp. 129-133.

Luo, L., Ming, J., Wu, D., Liu, P. & Zhu, S. (2017). Semantics-Based Obfuscation-Resilient Binary Code Similarity Comparison with Applications to Software and Algorithm Plagiarism Detection. *IEEE Transactions on Software Enigeering*, 43 (12), pp. 1157-1177.

Medeiros, F., Júnior, P. Bender, M., Menegussi, L. & Curcher, M. (2017). A blended learning experience applying project-based learning in an interdisciplinary classroom. *Proc. ICERI*, pp. 8665-8672.

Mirza, O.M., Joy, M. & Cosma, G. (2017). Style Analysis for Source Code Plagiarism Detection – An Analysis of a Dataset of Student Coursebook. Proc. *ICALT*, pp. 296-297.

Schleimer, S., Wilkerson, D.S. & Aiken, A. (2003). Winnowing: Local Algorithms for Document Fingerprinting. *Proc. SIGMOD 2003*, pp. 76-85.

Süß, J. & Billingsley, W. (2012). Using Continuous Integration of Code and Content to Teach Software Engineering with Limited Resources. *Proc. ICSE*, pp. 1175-1184.

Tian, Z., Liu, T., Zheng, Q., Zhuang, E., Fan, M. & Yang, Z. (2018). Reviving Sequential Program Birthmarking for Multithreaded Software Plagiarism Detection. *IEEE Transactions on Software Engineering*, 44 (5), pp. 491-511.

Wagner, N. R. (2002). Plagiarism by Student Programmers. http://www.cs.utsa.edu/~wagner/pubs/plagiarism.pdf.

Yagci, M (2018). Web-Mediated Problem-Based Learning and Computer Programming: Effects of Study Approach on Academic Achievement and Attitude. *Journal of Educational Computing Research*, 56 (2), pp. 272-292.

Yang, L., McKeown, N, Sahami, M. & Piech, C. (2018). TMOSS: Using Intermediate Assignment Work to Undestand Excessive Collaboration in Large Classes. *Proc. SIGCSE*, pp. 110-115.

[i] theory.stanford.edu/~aiken/moss/

[ii] www.redmine.org

[iii] jenkins.io

[iv] www.sonarqube.org

[v] elastic.io

[vi] www.healthincode.com