Importancia de roles simples en proyectos complejos

Meaning of simple roles in complex projects

Yadiel Ramos Rodríguez, Dayana Daniel Hernández

Universidad de las Ciencias Informáticas

yramosr@uci.cu, ddaniel@estudiantes.uci.cu

Resumen

La historia de los proyectos informáticos a escala mundial ha demostrado que los roles simples en la producción son infravalorados. Delegándose a planos secundarios y debido a esto muy pocas personas desean enfrascarse en ellos. El objetivo de este trabajo es demostrar a través de la práctica y apoyándose en el ciclo de desarrollo de un proyecto en la Universidad de las Ciencias Informáticas (UCI) la importancia que tienen los "roles simples" en la producción de software, y lo que pasaría si no contáramos con los mismos.

Palabras clave: Producción de software, proyectos, roles simples.

Abstract

The history of informatics projects worldwide has shown that simple roles in the production are undervalued. Delegating to secondary levels and because of that just few people want to engage in them. The objective of this study is to demonstrate through practice and taking the development cycle of a project at the University of Computer Science (UCI) as example, the importance of "simple roles" in the production of software, and what will happen if we doesn't have them.

***Key words**: Production of software, projects, simple roles.*

Introducción

Desde hace algunas décadas, el mundo de la informática ha estado visto a nivel de roles debido al amplio crecimiento que dicha rama ha tenido, y se quiera o no, ha surgido una amplia competencia entre estos, dándose a conocer ampliamente y versando sobre dicha competencia el criterio a manera general de los mal llamados roles simples.

El desarrollo de software no escapa a esto, y dentro de este se consideran roles simples los de montadores de interfaz, gestores de configuración, administradores de sistemas, documentadores, arquitectos de información y planificadores.

Materiales y métodos

Para el desarrollo de este trabajo se emplearon métodos empíricos como entrevistas a expertos, encuestas, cuestionarios y consulta de bibliografías que conllevaron a desarrollar un análisis exhaustivo que tiene como soporte los métodos de investigación científica.

Para evaluar cómo se manifestaba en la UCI dicha problemática internacional, se publicó una encuesta y se tomó un espacio muestral que incluía varios proyectos de desarrollo e investigación de software para darle respuesta.

La estrategia en la que se enfocó dicho trabajo fue exploratoria, y la muestra se tomó teniendo en cuenta el método no probabilístico intencional.

Desarrollo

El rápido avance de la tecnología ha traído consigo que cada día se vuelva más complejo el desarrollo del software pues son mucho más versátiles las posibilidades informáticas, por lo que las cosas que antes se podían realizar por un único rol hoy se ve dividido en varios. De aquí han surgido en dependencia de la metodología elegida, de lo que se desee, y del personal con que se cuente los llamados roles.

Lo problemático de esto es que la integración de dichos roles a la producción de software no se ve de la misma manera, pues para muchos los roles en verdad relevantes son los de arquitectos, programadores, analistas y diseñadores, y se sienten subvalorados cuando los colocan a desempeñarse en otros roles, ya que no los creen importantes o simplemente los ven como "poco serios".

El problema de esto radica en la carencia de información con que se cuenta respecto al verdadero valor de otros roles como los de montador de interfaz, gestor de configuración, administrador de sistemas, documentador, arquitecto de información y planificador, los que son considerados como "roles simples" y que sin embargo sin alguno de ellos sería bien difícil llegar a desarrollar un software con la calidad que el cliente espera.

Lo que si sucede en varios casos es que los lideres de proyectos no le dedican el mismo empeño a velar porque las personas que se encargan de estos roles sean utilizadas al máximo de sus capacidades, pues no tienen bien definidas las tareas que pueden desarrollar, y estos a su vez se limitan a desarrollar solo las tareas que le son asignadas, sin preocuparse por investigar las responsabilidades de su papel para hacer más eficiente su trabajo y garantizar que su parte en el software quedará 100% cubierta, que a fin de cuentas es su responsabilidad pues para esto surgió su rol.

De aquí les surgiría una incógnita:

¿Qué pasaría en un proceso de desarrollo de software sin personal encargado de desempeñar los roles simples?

En un proyecto de pequeña o mediana complejidad, suelen solaparse roles, o sea, una misma persona en momentos determinados puede jugar uno u otro rol, e incluso hacerlo al mismo tiempo, por lo cual no habría claridad en la definición de roles pero esto no sería un gran problema pues como el proyecto no es de gran complejidad tampoco son de gran complejidad las tareas a desempeñar, aunque esto traería consigo una muy baja identificación de las personas con el trabajo que puedan estar desarrollando. Ahora, en un proyecto de elevada complejidad, si esto sucediera, sería el caos, pues cada rol aumentaría drásticamente las funciones a desarrollar, y sería prácticamente imposible llevar a cabo las tareas de más de un rol a la vez por una misma persona, pues, o desocuparía parte de sus funciones, o, en el peor de los casos, no desarrollaría completamente ninguna de ellas; por lo tanto, sería inminente la necesidad de incluir personas que realicen las tareas de estos "roles simples", y que a su vez estas personas estén bien preparadas y se especialicen en los mismos, pues una incongruencia en cualquiera de las áreas de trabajo del proceso, podría llevar el mismo al fracaso.

Si un planificador no fuera capaz de concebir en cuanto a tiempo, recurso e iteraciones se refiere cómo será el ciclo de desarrollo, teniendo una mente futurista, se comenzaría a hacer un proyecto sin pensar en un producto, se caería en un proceso de iteraciones infinitas e indefinidas enfocado en el desarrollador y no en el cliente y que además desde su comienzo estaría muy lejos de cumplir con una fecha de entrega.

Si un gestor de configuración y cambios no fuera capaz de controlar adecuadamente todo el flujo de órdenes de cambio y sus respectivas respuestas, además de las distintas versiones en el código fuente, y cuando y como se hace cada cosa, se estaría cayendo en la aplicación de antipatrones o malas prácticas, donde cada desarrollador tendría su propia versión del componente que esté implementando, y al final sería muy difícil la tarea de solapar el trabajo de todos dando como resultado en el mejor de los casos un producto incompleto y con una mala interrelación entre sus partes.

Y así podría seguir demostrándose la vital importancia que tienen en un proyecto de desarrollo todos y cada uno de los roles, sin menospreciar ninguno de ellos, y no solo los roles, sino el correcto desempeño en los mismos.

Conclusiones

Todos los roles en un proyecto son importantes, tanto así, que el mal desempeño en cualquiera de estos podría desencadenar un fracaso, por lo que documentarse sobre las funciones y habilidades necesarias para los mismos es una tarea de primer orden al plantearse la ardua tarea de hacer un software con calidad. El merito está en desarrollar cada tarea y desarrollarla bien, cada cual sabe en qué esfera de la informática es en la que mejor se desenvuelve, sin importar si es analista, programador, documentador, planificador o cualquier otro rol.

Es evidente que todos son necesarios, más aún imprescindibles, por lo que no se deben enfocar los esfuerzos en un rol en particular, y menos aún en criticar otros, sino en formar un equipo con bases sólidas, para enfrentar cualquier tarea que necesite el concurso de los modestos esfuerzos de un equipo en general.

Bibliografía Consultada

JACOBSON, Ivar; RUMBAUGH, James; BOOCH, Grady, “El Proceso Unificado de Desarrollo”. Addison Wesley. 2000.

Rational Unified Process (de la Suite de Rational 2003).

RUMBAUGH, James, JACOBSON, Ivar; BOOCH, Grady, “El lenguaje unificado de modelado. Manual de referencia”. Addison Wesley. 2000