INGENIERÍA DE SISTEMAS

Desafíos y estrategias prácticas de los estudios empíricos sobre las técnicas de prueba del software

Edgar Serna[1§], Fernando Arango[2]

[1§] *Facultad de Ingenierías Fundación Universitaria Luis Amigó, Medellín Colombia, edgar.sernamo@amigo.edu.co*

[2] *Facultad de Minas Universidad Nacional de Colombia Medellín, farango@unal.edu.co*

(Recibido: Junio 15 de 2010 - Aceptado: Junio 14 de 2011)

Resumen

Este artículo de reflexión tiene como objetivo discutir una serie de cuestiones que típicamente se plantean cuando se realizan estudios empíricos con técnicas de prueba del software. Aunque algunos problemas son generales a todas las disciplinas empíricas, los estudios de prueba del software enfrentan una serie de desafíos específicos. Algunos de los más importantes se discuten a continuación.

Palabras Claves: Pruebas del software, Fallas del software, Técnicas de prueba, Estudios empíricos.

SYSTEMS ENGINEERING

Challenges and practical strategies of the empirical studies on software testing techniques

Abstract

This reflexion paper aims at discussing a number of issues that typically arise when performing empirical studies with software testing techniques. Though some problems are general to all empirical disciplines, software testing studies face a number of specific challenges. Some of the main ones are discussed below.

Keywords: Software testing, Software faults, Testing techniques, Empirical studies.

1. Introducción

La prueba es una importante actividad de ingeniería, y es responsable de una porción significativa de los costos del desarrollo y el mantenimiento del software (Leung& White, 1989, Beizer, 1990), por lo que es necesario que investigadores y profesionales comprendan sus ventajas y desventajas, lo mismo que los factores que influyen en las técnicas que se utilizan para llevarla a cabo. Se pueden obtener algunos conocimientos mediante el uso de marcos analíticos, relaciones de subsunción o por axiomas (Rapps & Weyuker, 1985, Weyuker, 1988, Rothermel & Harrold, 1996); sin embargo, en general, las técnicas de prueba son heurísticas y su desempeño varía de acuerdo con los diferentes escenarios, por lo que deben estudiarse empíricamente.

Los estudios empíricos de las técnicas de prueba, como los estudios de los ingenieros que realizan las pruebas, implican muchos desafíos, y ventajas y desventajas de costo-beneficio, lo que limita el progreso de los estudios en esta área. En general, se pueden considerar dos clases de estudios empíricos de pruebas del software: 1) los experimentos controlados, y 2) los estudios de caso. Los experimentos controlados se centran en un riguroso control de las variables en un intento por preservar la validez interna y apoyar conclusiones acerca de la causalidad, pero las limitaciones que resultan de ejercer este control pueden restringir la capacidad de generalizar los resultados (Trochim, 2000). Los estudios de caso sacrifican el control, por lo tanto la validez interna, pero pueden incluir un contexto más amplio (Yin, 1994). Cada una de estas clases de estudios puede ofrecer una perspectiva sobre las técnicas de prueba del software, y juntos son complementarios.

Sin importar el tipo de estudio aplicado, en este artículo de reflexión se discute una serie de cuestiones que se plantean típicamente cuando se realizan estudios empíricos acerca de las técnicas de prueba del software. Aunque algunos problemas son generales a todas las disciplinas empíricas, los estudios de prueba del software enfrentan una serie de desafíos específicos, y algunos de los más importantes se discuten en este trabajo.

2. Siembra de fallas

Uno de los principales temas que se debe enfrentar cuando se evalúan y comparan técnicas de prueba es la cuantificación de su eficacia en la detección de fallas. Idealmente, las fallas en el software deberían representar las fallas de la vida real (DeMillo, 1978), de modo que los resultados de su eficacia fueran en sí mismos representativos. Pero existe una serie de problemas con esta estrategia:En primer lugar, muchos componentes del mundo real, los subsistemas o incluso los sistemas, no tienen un conjunto lo suficientemente grande de defectos que sea adecuado para un análisis cuantitativo. Es necesario recordar que, aunque se espera que las técnicas de prueba estén asociadas con tasas de detección de fallas, esta relación es de naturaleza estadística, por lo que la detección de dichas fallas será, hasta cierto punto, un proceso estocástico. Por lo que, muy a menudo, cuando se comparan las técnicas de prueba se están comparando las distribuciones de la detección de fallas, y se recurre a las pruebas de inferencia estadística para determinar si los resultados podrían haberse obtenidos al azar. Para ello es necesario trabajar con muestras de fallas suficientemente grandes, que raramente están disponibles en los sistemas del mundo real ¡afortunadamente!.

En segundo lugar, cuando el referente son las fallas del mundo real, muy probablemente se refieren a una población estadística que no existe, o que cada empresa y sistema estén asociados con diferentes poblaciones de fallas. En otras palabras, si el objetivo es la "falla real", esa meta probablemente sea difícil de alcanzar. Aunque esto no quiere decir que los estudios industriales no sean útiles.

Debido a todo lo anterior, muchos investigadores tienen que recurrir a la siembra de fallas para llevar a cabo estudios empíricos (Weyuker, 1993, Hutchins et al., 1994, Graves et al., 2001, Briand et al., 2003, Briand et al., 2004), aunque son bien conocidos los problemas asociados a dicha siembra. ¿Cómo ser imparcial y objetivo cuando se siembran fallas con el fin de evaluar una técnica? ¿Cómo garantizar que los resultados obtenidos se pueden generalizar a la población de fallas real? No existe una respuesta adecuada. Sin

embargo, los operadores de mutación utilizan una técnica para sembrar fallas (Offutt, 1992, Briand et al., 2003). No es posible garantizar que las fallas sembradas sean representativas de una población en particular, pero sí es posible asegurar que una amplia variedad de fallas sean insertadas sistemáticamente, de manera un tanto imparcial, al azar (King &Offutt, 1991, Kim et al., 2000, Delamaro et al., 2001), y probablemente esto es lo mejor que se puede hacer en un entorno artificial.

Se han hecho estudios con la idea de obtener datos únicos de la efectividad en la detección de fallas y en el costo (Briandet al., 2004), buscando además una mejor comprensión de qué tipo de fallas son más difíciles de detectar para una determinada técnica (Antoniolet al., 2002). Tales estudios deben complementarse con trabajos de campo, como se discute a continuación. Otra cuestión relacionada con la siembra de fallas es que si muchas de las fallas sembradas son demasiado fáciles de detectar, entonces no es posible diferenciar la eficacia de las técnicas alternativas en la detección de fallas; pero sí resultan útiles para comprobar el porcentaje de casos de prueba que encuentran fallas en un programa defectuoso. Las fallas sencillas, así como las que posiblemente no tienen un impacto sobre el comportamiento del sistema, probablemente se deban ignorar a la hora de evaluar la eficacia.

3. Contexto académico vs industrial

Se han dado muchas discusiones acerca del valor de los experimentos en el ambiente académico (Juristo & Moreno, 2001), y especialmente la preocupación por el hecho de que las herramientas utilizadas pueden no ser representativas, en términos de escala y complejidad, de sus homólogas industriales; y que cuando participan estudiantes, sería natural preguntarse si los resultados obtenidos pueden ser comparados con los de los profesionales, quienes muchas veces tienen más experiencia, y por supuesto, como se mencionó anteriormente, que existe un conjunto de fallas con las que se tiene que trabajar y que puede no ser representativo de las fallas detectadas en el campo.

Todas son preocupaciones válidas. Pero lo que la experiencia ha demostrado, en el ámbito más general de la Ingeniería de Software empírica, es que no existen estudios empíricos perfectos. Los estudios de campo, en entornos industriales, también implican una serie de problemas.

• El número de fallas detectadas pueden no ser lo suficientemente amplio como para permitir un análisis estadístico cuantitativo.

• Cuando se trabaja con profesionales activos, a veces no se preocupan por actualizar su campo de formación. En situaciones en las que se requiere evaluar nuevas o avanzadas técnicas, pueden no estar actualizados, especialmente al evaluar el límite de los beneficios potenciales de una técnica cuando se aplica correctamente.

• Los estudios en el ámbito académico suelen ser más fáciles de controlar. Uno de los retos importantes de la experimentación es asegurarse de que no existan efectos de confusión entre los factores que se están estudiando técnicas de la prueba y otros factores extraños, humanos (Wohlin et al., 2000). Esto suele ser muy difícil de asegurar en un contexto industrial.

Así, a partir de la discusión anterior, es posible ver que tanto el contexto académico como el industrial tienen puntos fuertes y débiles. Los estudios en el mundo académico a menudo son fuertes en términos de validez interna por ejemplo, la capacidad para sacar conclusiones adecuadas de los datos; y puntos débiles en cuanto a la validez externa se refiere por ejemplo, es difícil conocer hasta qué punto se pueden generalizar los resultados en los contextos industriales. Los estudios de campo tienen exactamente los puntos fuertes y débiles opuestos. Aunque también se podría argumentar que los resultados en un proyecto determinado en una organización podrían no ser generalizables a otras organizaciones y proyectos.

De acuerdo con las referencias mencionadas, se concluye que tanto los estudios académicos como los de campo, son necesarios. Los primeros son útiles para obtener datos únicos de la eficacia de las técnicas y para comprender mejor sus fortalezas y debilidades. Los estudios de campo son más adecuados para evaluar las dificultades al aplicar las técnicas en la práctica y para confirmar

los resultados obtenidos en un conjunto real de fallas. También son muy útiles para intentar análisis de costo-eficacia como el costo de los datos que se recoge. Ningún estudio llegará a ser perfecto, o a estar cerca de serlo, pero con el tiempo la acumulación de conocimientos obtenidos en estudios posteriores puede permitir la construcción de un cuerpo de conocimientos. Por último, los estudios en un entorno académico a menudo son el primer paso antes de los estudios en entornos industriales (Juristo & Moreno, 2001).

4. Replicación

La anterior discusión implica que los estudios se replicarán en diferentes entornos, pero un sólo estudio es probable que no tenga un impacto significativo, por lo que es necesario que se replique para que los resultados sean creíbles. La replicación se ha aplicado en otras disciplinas a lo largo del tiempo y existen técnicas, llamadas meta-análisis, para obtener conclusiones de un conjunto de experimentos (Wohlin*et al*., 2000); sin embargo, replicación necesariamente no significa repetición exacta de un estudio. Esto se debe a que: los artefactos pueden ser diferentes, la formación y antecedentes de los participantes si los hay puede variar considerablemente, incluso el diseño del estudio pueden cambiar para hacer frente a una amenaza en particular por validar, o simplemente el número de observaciones puede ser mayor o menor. Todo esto afecta la capacidad de obtener efectos visibles. Esta información se debe reportar al escribir acerca de una replicación, de modo que a largo plazo las diferencias entre los estudios se puedan explicar, posiblemente a través de meta-análisis. Probablemente sería útil para la comunidad de investigación en pruebas de software definir una plantilla de información para reportar cuándo se realizan, y para documentar los experimentos realizados.

5. Participación de seres humanos

Los estudios en pruebas del software se pueden clasificar en dos categorías: 1) estudios que involucran sujetos humanos aplicando las técnicas de prueba (Antoniol*et al*., 2002), y 2) las simulaciones, donde las técnicas se aplican mediante simulación de la construcción del plan de pruebas y su ejecución en la versión defectuosa del programa (Briand*et al*., 2004). Ambos tipos de estudios tienen ventajas e inconvenientes.

El primero permite evaluar no sólo la rentabilidad sino también la aplicabilidad de las técnicas por los ingenieros especializados, lo que explica los factores humanos en las conclusiones; después de todo, los seres humanos son un factor que todavía no puede estar por fuera de las pruebas de software, y que es una cuestión importante relacionada con la naturaleza estadística de las tasas de detección de fallas. Para comparar realmente las técnicas de prueba, de forma rigurosa y cuantitativa, se necesita generar un gran conjunto de pruebas para cada una de las técnicas. Esto suele ser inviable cuando los seres humanos están involucrados, y es aquí donde la simulación entra en acción para permitir la generación de un conjunto amplio de pruebas, ejecutarlo en un gran número de programas defectuosos y, por lo tanto, que sean susceptibles de un riguroso análisis estadístico. Sin embargo, el principal problema es garantizar que el proceso de simulación no introduzca algún sesgo en los resultados, que sea de alguna manera representativo del conjunto de pruebas que pueden generar los seres humanos. Aunque ésta no es la forma como los humanos cometen errores al usar técnicas de prueba, algunas técnicas pueden ser más complejas y dar lugar a más errores, especialmente si no están bien soportadas por herramientas. Así, de nuevo, se concluye que ambos tipos de estudios son necesarios debido a que son complementarios.

6. Conclusiones

Existe una serie de prácticas que pueden ayudar a aliviar los problemas discutidos en este trabajo. Debido al amplio uso de la experimentación se requiere que esos experimentos se preparen mejor y que exijan más esfuerzo; además, es necesario que los investigadores compartan el material experimental que obtienen de sus aplicaciones. Por otra parte, con la experimentación en un conjunto de sistemas comunes de referencia es posible hacer comparaciones más factibles de las técnicas, y ese conjunto de sistemas se pueden utilizar como punto de referencia para la validación inicial de las mismas técnicas. Una de

las dificultades es que, por lo general, diferentes técnicas de prueba requieren diferentes piezas de información del sistemapor ejemplo, especificaciones en diferentes formas, o tener como objetivo diferentes tipos de sistemas, por lo que elegir apropiados puntos de referencia puede no ser tan fácil.

Los experimentos deben producir una documentación que les permita a otros investigadores replicarlos fácilmente. Idealmente, deberían contener todo el material necesario para realizar el experimento y ser accesible públicamente, bajo ciertas condiciones. Esto permitiría a la comunidad de investigadores converger mucho más rápido hacia resultados creíbles.

Cada vez que en un experimento se realiza un análisis de datos, es necesario que dichos datos queden disponibles para la comunidad investigadora, incluso si ello implica, de alguna manera, sanearlos cuando la confidencialidad es un problema. Esto permitiría que otro investigador los analizara y, posiblemente, llegara a conclusiones diferentes.

Por último, ya que no es posible esperar que los experimentos sean perfectos en lo que respecta a la validez de todas las fallas, es necesario establecer estándares acerca de cómo llevarlos a cabo y cómo reportarlos. Algunos podrían basarse en otros campos experimentales, pero definitivamente habría que hacer una adecuada adaptación. Esto facilitará la revisión de los artículos que detallan resultados de pruebas experimentales y ayudará a la efectiva publicación de más resultados.

7. Agradecimientos

Los autores queremos agradecer a la Fundación Universitaria Luis Amigó la financiación de la investigación "Estructuración de una metodología genérica para la realización de pruebas de caja negra en los sistemas de información", de cuyo proceso se obtuvo el material para realizar este artículo.

8. Referencias

Antoniol, G., Briand, L. C., Di Penta, M. &Labiche, Y. (2002). *A Case Study Using the Round-Trip Strategy for State-Based Class Testing*, In *13th IEEE International Symposium on Software Reliability Engineering*, Annapolis, MD, USA, 269-279.

Beizer, B. (1990). *Software Testing Techniques*. New York: Van Nostrand Reinhold.

Briand, L. C., Labiche, Y. & Sun, H. (2003). Investigating the Use of Analysis Contracts to Improve the Testability of Object-Oriented Code, *Software - Practice and Experience*, 33 (7), 637-672.

Briand, L. C., Labiche, Y. & Wang, Y. (2004). *Using Simulation to Empirically Investigate Test Coverage Criteria*. In IEEE/ACM International Conference on Software Engineering, Edinburgh, UK, 86-95.

Delamaro, M. E., Maldonado, J. C. &Mathur, A. P. (2001). Interface Mutation: An Approach for Integration Testing, *IEEE Transactions of Software Engineering*, 27 (3), 228-247.

DeMillo, R. A. (1978). Hints on Test Data Selection: Help for the Practicing Programmer. *IEEE Computer* 11 (4), 34-41.

Graves, T. L., Harrold, M. J., Kim, J. M., Porter, A. &Rothermel, G. (2001). An Empirical Study of Regression Test Selection Techniques, ACM *Transactions on Software Engineering and Methodology,* 10 (2), 184-208.

Hutchins, M., Froster, H., Goradia, T. &Ostrand, T. (1994). *Experiments on the Effectiveness of Dataflo and Controlflow-Based Test Adequacy Criteria,* In16th International Conference on Software Engineering, Sorrento, Italy, 191-200.

Juristo, N. & Moreno, A. M. (2001). *Basics of Software Engineering Experimentation*, Kluwer: Springer.

Kim, S., Clark, J. A. &McDermid, J. A. (2000). *Class Mutation: Mutation Testing for Object-Oriented Programs*, In28th International Conference on Software Engineering, Erfurt, Germany, 869-872.

King, K. N. & Offutt, A. J. (1991). A Fortran Language System for Mutation-Based Software Testing, *Software-Practice and Experience*, 21 (7), 686-718.

Leung, H. K. N. & White, L. (1989). *Insights into regression testing*. In Conference on Software Maintenance, Miami, FL, USA, 60-69.

Offutt, A. J. (1992). Investigations of the Software Testing Coupling Effect, *ACM Transactions on Software Engineering and Methodology*, 1 (1), 3-18.

Rapps, S. &Weyuker, E. J. (1985). Selecting software test data using data flow information. *IEEE Transactionson Software Engineering* 11 (4), 367375.

Rothermel, G. &Harrold, M. J. (1996). Analyzing regression test selection techniques. *IEEE Transactionson Software Engineering*, 22 (8), 529-551.

Trochim, W. (2000). *The Research Methods Knowledge Base*. Cincinnati: AtomicDog.

Weyuker, E. (1988). The evaluation of program-based software test data adequacy criteria. *Communications of ACM*, 31 (6), 668-675.

Weyuker, E. (1993). More Experience with Data Flow Testing, *IEEE Transactions on Software Engineering,* 19 (9), 912-919.

Wohlin, C., Runeson, P., Host, M., Ohlsson, M. C., Regnell, B. &Wesslen, A. (2000). *Experimentation in Software Engineering* - An Introduction, Kluwer: Springer.

Yin, R. K. (1994). *Case Study Research: Design and Methods*. (Applied Social Research Methods). London: Sage Publications.