Estudio de la Efectividad de Tres Técnicas de Evaluación de Código: Resultados de una Serie de Experimentos

N. Juristo[1], S. Vegas[1], M. Solari[2], S. Abrahao[3], I. Ramos[4]

[1]Universidad Politécnica de Madrid, [2]Universidad ORT Uruguay,
[3]Universidad Politécnica de Valencia, [4]Universidad de Sevilla

[1]{natalia,svegas}@fi.upm.es,[2]martin.solari@ort.edu.uy, [3]sabrahao@dsic.upv.es,[4]iramos@us.es

Abstract. Hasta la fecha se han evaluado distintas técnicas de verificación y validación teórica y empíricamente. La mayoría de las evaluaciones empíricas se han llevado a cabo sin sujetos, abstrayendo el efecto del sujeto sobre la técnica a la hora de aplicarla. Hemos evaluado mediante un experimento con sujetos la efectividad de tres técnicas de verificación y validación de código: partición en clases de equivalencia, cobertura de decisión y lectura de código mediante abstracciones sucesivas, estudiando la capacidad de las técnicas para la detección de fallos en tres programas distintos. Hemos replicado el experimento ocho veces en cuatro entornos distintos. Los resultados arrojan diferencias entre las técnicas y señalan variables contextuales del proyecto software que deberían considerarse cuando se quiera elegir o aplicar una técnica de verificación y validación.

Keywords: Verificación y validación, experimentación, combinación de resultados experimentales

1 Introducción

La Verificación y Validación (V&V) es una parte importante del proceso de desarrollo del software, a la vez que cara. En vista de la variedad de técnicas de V&V disponibles, es interesante explorar las características de cada técnica en término de las variables que afectan a su efectividad [3]. Es decir, qué técnicas son más adecuadas para: encontrar un determinado tipo de defecto, ser usadas por un determinado tipo de desarrollador, un determinado tipo de software, y, en general, para determinadas características del proyecto software.

Hay estudios empíricos que han intentado entender el comportamiento de distintas técnicas de V&V. La mayor parte de estos trabajos se han llevado a cabo sin utilizar sujetos que aplicasen las técnicas, tal y como se muestra en la sección 2. Dicho de otro modo, estos trabajos se centran en examinar el comportamiento de las técnicas en lo que podríamos llamar su *estado puro*. Sin embargo, no sólo es importante investigar el comportamiento *puro* de las técnicas, sino la influencia del factor humano en su aplicación.

La mayor parte de los estudios empíricos que usan sujetos versan sobre técnicas estáticas [1], [6], [10], [31], [38], [46]. Tan sólo hay unos pocos estudios que investigan técnicas de pruebas [2], [7], [26], [32], [40]. Sin embargo, debido a las diferencias contextuales y a la baja tasa de replicación (tanto interna como externa), los resultados son extremadamente inmaduros, no

siendo posible distinguir eventos fortuitos de regularidades.

En este artículo examinamos la efectividad de tres técnicas de V&V: dos técnicas dinámicas (partición en clases de equivalencia y cobertura de decisión) y una técnica estática (lectura de código mediante abstracciones sucesivas), aplicadas a tres programas distintos. Hemos replicado el experimento ocho veces en cuatro universidades distintas. Los resultados arrojan diferencias en la efectividad de las técnicas, siendo posible identificar variables clave que influyen en su efectividad.

El artículo está organizado del siguiente modo. La sección 2 describe trabajos relacionados con la investigación que aquí se presenta. La sección 3 explica el método de investigación utilizado. La sección 4 describe el experimento realizado. Las secciones 5 y 6 presentan los resultados obtenidos. La sección 7 muestra la interpretación de los resultados. La Sección 8 lista las amenazas a la validez del experimento. Finalmente, la sección 9 muestra las conclusiones.

2 Trabajos Relacionados

Existen distintos estudios sobre el comportamiento de técnicas de V&V. Hemos agrupado estos estudios en: estudios teóricos, estudios empíricos con sujetos y estudios empíricos sin sujetos.

Los estudios teóricos examinan las técnicas en estado puro, desde el razonamiento y la lógica deductiva. Analizan el fundamento teórico de las técnicas, estudiando la efectividad de técnicas basadas en el código [8], [9], [11], [17], [21], [22], [34], [35], [39], [49], [52], [56] o técnicas de regresión [41], [43].

Los estudios empíricos sin sujetos investigan las técnicas desde la práctica y el razonamiento inductivo, simulando su aplicación. Tratan distintos aspectos de las técnicas de V&V [25]. Algunos tratan la generación de casos de prueba, y comparan la efectividad y eficiencia de técnicas basadas en la especificación, en el código, o en las faltas [5], [23], [36], [37], [50], [54]. Otros, evalúan conjuntos de casos de prueba atendiendo a distintos criterios (flujo de control y datos, mutación y sus variantes), midiendo el nivel de cobertura (adecuación), y relacionando cobertura y tamaño con la efectividad [13], [14], [15], [16], [28], [51]. Finalmente, algunos estudios evalúan técnicas para seleccionar casos de prueba de un conjunto ya existente (técnicas de regresión [4], [44], [48], filtrado [19], [27], [29], o priorización [12], [42], [45], [53]).

Los estudios empíricos con sujetos tienen en cuenta cómo influye el sujeto en el comportamiento de la técnica. La mayor parte de estos estudios evalúan técnicas estáticas [1], [6], [10], [31], [38], [46]. Aquellos que evalúan técnicas de pruebas, han investigado la generación de casos de prueba, comparando la eficiencia y efectividad de técnicas basadas en la especificación y de flujo de control aplicadas por sujetos [2], [7], [26], [32], [40].

Ambos estudios empíricos, con y sin sujetos, son necesarios. Los estudios con sujetos están más cercanos a la realidad, en la que los humanos aplican las técnicas. Pero no consiguen separar el comportamiento de la técnica de la influencia que el sujeto podrían tener en la efectividad de la técnica.

3 Método de Investigación

Hemos realizado un estudio empírico con sujetos para avanzar en el conocimiento de tres técnicas de evaluación de código. Dicho estudio consta de varias replicaciones de un mismo experimento. Los resultados de un único experimento podrían corresponder a un suceso fortuito obtenido por casualidad. Los experimentos deben ser repetidos para comprobar que los resultados obtenidos inicialmente son reproducibles y siguen un patrón regular [18].

El experimento se ha replicado ocho veces. Cinco replicaciones se llevan a cabo en la Universidad Politécnica de Madrid (a las que denominaremos UPM01, UPM02, UPM03, UPM04 and UPM05, respectivamente). Adicionalmente, se realiza una replicación del experimento en la Universidad de Sevilla (a la que denominaremos UdS05), otra en la Universidad Politécnica de Valencia (a la que denominaremos UPV05) y otra en la Universidad ORT en Uruguay (a la que denominaremos ORT05). El número de sujetos participantes es: 42, 39, 29, 35 y 31 en la UPM; 31 en la UPV; 172 en UdS; y 76 en ORT.

Para entender mejor los resultados del estudio, se llevan a cabo reuniones entre los investigadores participantes en cada replicación. En estas reuniones se examinan posibles desviaciones de la operación experimental con respecto a lo planificado inicialmente, así como incidentes que hayan podido ocurrir. Estas reuniones ayudan a entender mejor las condiciones bajo las cuales los sujetos aplican las técnicas.

Para sintetizar los resultados de las ocho replicaciones, seguimos una aproximación en dos etapas. En primer lugar, examinamos de forma conjunta los resultados de todas las replicaciones. Para esto, identificamos la tendencia hacia la (no) significación de los efectos estudiados (incluyendo sus interacciones). Examinamos el comportamiento en cada replicación de aquellos efectos con tendencia a la significación, basándonos en las medias obtenidas para la variable respuesta. Esto implica reanalizar algunos de los resultados ya analizados para cada experimento por separado o realizar nuevos análisis. En segundo lugar, estudiamos las diferencias contextuales entre los entornos en los que se llevaron a cabo las replicaciones. Finalmente, integramos ambos tipos de resultados.

4 Descripción del Experimento

El experimento usa los paquetes de replicación elaborados por Kamsties y Lott [26] y Roper *et al.* [40] para sus experimentos, aunque el material ha sido adaptado para nuestro objetivo y contextos. La hipótesis nula del experimento es:

H_0: *No hay diferencia en la efectividad de partición de clases de equivalencia, cobertura de decisión y lectura de código.*

Para medir la efectividad de las técnicas, distinguimos entre *defectos observables* y *observados*. Si un sujeto aplica una técnica dinámica para generar un caso de prueba que ejercita una falta y genera un fallo, este fallo es *observable* por la técnica. Sin embargo, el sujeto podría no ver (*observar)* el fallo, y por tanto no reportarlo. En las técnicas estáticas, los *defectos observables* coinciden con los *observados*. Aquí, cuando hablemos de efectividad nos referiremos a *defectos observables*. Para las técnicas dinámicas, medimos los *defectos observables* como el porcentaje de sujetos que generan un caso de prueba que revela el fallo asociado a una determinada falta. Para la técnica estática, es como el porcentaje de sujetos que reportan una determinada falta.

El modo en que los sujetos aplican las técnicas es:

- En la *lectura por abstracciones sucesivas (LC)* [30], los sujetos reciben el código fuente del programa. Identifican funciones y escriben sus respectivas especificaciones. Agrupan las funciones y sus especificaciones y repiten el proceso hasta que hayan abstraído el código fuente entero para conformar la especificación del programa. Después, reciben la especificación original, e identifican las faltas en el programa a partir de las inconsistencias entre las especificaciones originales y abstraídas.
- En *partición en clases de equivalencia (PCE)* [33], los sujetos reciben la especificación del programa y diseñan casos de prueba. Después, reciben la versión ejecutable del programa y ejecutan los casos de prueba. Finalmente, identifican los fallos encontrados.
- Se requiere a los sujetos que consigan, en la medida de lo posible, 100% *cobertura de*

decisión (CD) [33]. Reciben el código fuente, junto con una descripción a alto nivel de lo que hace el programa (no su especificación) y diseñan casos de prueba. Después, reciben la versión ejecutable del programa y ejecutan los casos de prueba. Finalmente, reciben la especificación del programa e identifican los fallos encontrados.

El experimento usa tres programas distintos, escritos en C:

- *cmdline*. Parser que lee la línea de entrada y muestra un resumen de los contenidos de la misma. Tiene 209 líneas de código (LOC).
- *nametbl*. Implementa la estructura de datos y operaciones de una tabla de símbolos. Tiene 172 LOC.
- *ntree*. Implemente la estructura de datos y operaciones de un árbol n-ario. Tiene 146 LOC.

Cada programa contiene siete faltas distintas. Están definidas según [2], que identifica seis tipos de faltas. Cada tipo de falta puede ser de *omisión* (algo que falta) o *comisión* (algo que está mal). Los tipos de faltas usados en nuestro experimento son:

- *Inicialización*. Una falta de comisión es, por ejemplo, asignar un valor incorrecto a una variable en una función, mientras la falta de inicialización de una variable es una falta de omisión. El experimento tiene faltas de inicialización de comisión (F4) y de omisión (F3).
- *Control*. Una falta de comisión es, por ejemplo, un predicado incorrecto en la condición de una decisión, mientras que si falta el predicado es una falta de omisión. El experimento tiene faltas de control de comisión (F5) y de omisión (F6).
- *Computación*. Una falta de comisión puede ser un operador aritmético incorrecto en la parte derecha de una asignación. El experimento tiene faltas de computación de comisión (F7).
- *Cosmética*: Una falta de comisión es, por ejemplo, una palabra mal escrita en un mensaje de error. En las faltas de omisión falta un mensaje de error. El experimento cubre faltas de comisión (F2) y de omisión (F1).

Se crean *dos versiones* de cada programa, para tener dos instancias de cada falta en cada uno. Los programas son pequeños, y no podemos insertar tantas faltas como querríamos sin que unas faltas enmascaren a otras. Dos versiones de un mismo programa difieren únicamente en sus faltas concretas, pero siempre contienen el mismo número y tipo de faltas. Las faltas tienen una probabilidad del 100% de ser detectados por todas las técnicas.

El experimento tiene un diseño factorial [24] con cuatro factores: programa, técnica, versión y falta, y se lleva a cabo en varias sesiones. Cada sujeto aplica cada técnica una vez, sobre un programa distinto. Cada entorno tiene condiciones contextuales que afectan al diseño del experimento, lo que provoca que las replicaciones en cada localización no sean exactamente iguales. La Tabla 1 muestra estas condiciones, junto con las decisiones de diseño tomadas. La Tabla 2 resume la descripción de cada replicación.

- **UPM:** Se ejercitan las tres técnicas y se usan los tres programas, pues no hay restricciones de tiempo. Se realizan tres sesiones, ya que no hay límite en la duración de la sesión. En cada sesión, cada sujeto aplica una técnica a un programa individualmente, y distintos sujetos aplican distintas técnicas, para cubrir las seis posibilidades de orden de aplicación de las tres técnicas (PCE-CD-LC, PCE-LC-CD, CD-PCE-LC, CD-LC-PCE, LC-PCE- CD y LC-CD-PCE). De esta forma, las tres técnicas se ejercitan en cada sesión, y se trabaja sobre el mismo programa. El día y el programa se confunden, por este motivo, los programas se usan en distinto orden en distintas replicaciones (cmdline-ntree-nametbl en UPM01, UPM02 y UPM04, y ntree-cmdline-nametbl en UPM03 y UPM05). Cada sesión tiene una duración de cuatro horas, lo que supone tiempo ilimitado, ya que la tarea puede completarse en menos tiempo. En cada sesión, lo sujetos aplican la técnica correspondiente y, para el caso de las técnicas dinámicas, ejecutan los casos de prueba generados. Como los sujetos no conocen las técnicas, reciben tres sesiones de aprendizaje de cuatro horas cada una para aprenderlas.

Tabla 1. Adaptación del experimento al entorno.

VARIABLE	CONDICIÓN	ENTORNO	DECISIÓN DE DISEÑO
Tiempo	No restringido	UPM, UdS	3 técnicas y 3 programas
	Restringido	UPV, ORT	2 técnicas y 2 o 3 programas
Disponibilidad de ordenadores	Sí	UPM, UPV	Trabajo individual
	Limitada	UdS	Trabajo en parejas
	No	ORT	No se ejecutan casos de prueba
Conocimiento de las técnicas	Sí	UPV, UdS	Tutorial
	No	UPM, ORT	Curso completo
Secuencia de formación	Secuencial	UPM, ORT	Formación, luego experimento
	Intercalado	UPV, UdS	Formación intercalada con experimento
Duración de la sesión	Ilimitada	UPM	Ejecución de casos de prueba al aplicar la técnica.
		ORT	Aplicación de todas las técnicas en 1 sesión
	Limitada	UPV, UdS	Ejecución de casos de prueba en una sesión separada para un solo programa

Tabla 2. Description de cada replicación.

Localización	UPM	UdS	UPV	ORT
Técnicas	Part. clas. eq. Cob. decisión Lectura código	Part. clas. eq. Cob. decisión Lectura código	Part. clas. eq. Cob. decisión	Part. clas. eq. Cob. decisión
Programas	cmdline nametbl ntree	cmdline nametbl ntree	cmdline nametbl ntree	cmdline nametbl
Sujetos	Alumnos 5º curso informática	Alumnos 5º curso informática	Alumnos 5º curso informática	Alumnos 2º curso informática
Aplicación de la técnica	Individual	Parejas	Individual	Individual
Formación	Curso	Tutorial	Tutorial	Curso
Sesiones	3	4	3	1
Técnicas y programas por sesión	1 programa 3 técnicas	1 técnica 3 programas	1 técnica 3 programas	2 técnicas 2 programas
Trabajo por sesión	1 técnica	1 técnica	1 técnica	2 técnicas
Ejecución casos prueba	Misma sesión	Sesión separada	Sesión separada	No

- **UdS:** Se ejercitan las tres técnicas y se usan los tres programas, ya que no hay restricciones de tiempo. Se realizan cuatro sesiones de dos horas, ya que la duración de la sesión está limitada. En cada una de las tres primeras sesiones, cada sujeto aplica una técnica a un programa, trabajando en parejas ya que no hay suficientes ordenadores; distintas parejas trabajan con la misma técnica, pero sobre programas distintos, ya que la formación tiene que intercalarse con la operación experimental. En la cuarta sesión, los sujetos ejecutan casos de prueba para uno de los programas para los que han aplicado una técnica dinámica. La formación consiste en tres tutoriales de dos horas, porque los sujetos ya conocen las técnicas. Cada tutorial se realiza antes de la aplicación de la técnica.
- **UPV:** La técnica de lectura de código no se ejercita por restricciones de tiempo, pero se utilizan los tres programas. Se realizan tres sesiones de dos horas, ya que la duración de la sesión está limitada. En cada una de las dos primeras sesiones, cada sujeto aplica una técnica a un programa individualmente, ya que hay suficientes ordenadores; distintos sujetos aplican la misma técnica, pero a distintos programas, ya que la formación debe intercalarse con la operación experimental. En la tercera sesión los sujetos ejecutan casos de prueba para uno de los programas para los que aplicaron una técnica dinámica. La formación consiste en dos tutoriales de dos horas, ya que los sujetos conocen las técnicas. Cada tutorial se realiza antes de la aplicación de la técnica.

- **ORT:** La técnica de lectura de código no se ejercita, ni se usa el programa ntree por restricciones de tiempo. El experimento se realiza en una sesión, en la que cada sujeto aplica las dos técnicas a los dos programas de forma individual, ya que no hay límite en la duración de la sesión. Distintos sujetos aplican las técnicas a distintos programas y en distintos órdenes para cubrir las seis posibilidades de orden de aplicación de dos programas y dos técnicas (PCE/cmdline-CD/nametbl, PCE/nametbl-CD/cmdline, CD/cmdline-PCE/nametbl y CD/nametbl-PCE/cmdline). Lo sujetos no ejecutan casos de prueba, ya que no hay ordenadores disponibles. Como los sujetos no conocen las técnicas, reciben dos sesiones de aprendizaje de cuatro horas cada una para aprenderlas. Los sujetos no tienen apenas experiencia en programación.

Para analizar los resultados individuales de cada replicación [24], utilizamos el análisis de varianza (ANOVA), al 95% de confianza y pruebas post-hoc. Es posible utilizar el ANOVA porque todas las muestras cumplen los requisitos de normalidad y homogeneidad de varianzas.

5 Resultados Obtenidos

La Tabla 3 muestra los p-valores del ANOVA para cada replicación. Las dos primeras filas de la Tabla 3 indican, respectivamente, que el modelo del ANOVA es adecuado (el p-valor es menor de 0,05 en todos los casos), y que la potencia estadística es alta (el p-valor es mayor de 0,8 en todos los casos). Las celdas sombreadas indican los efectos o combinación de efectos significativos. Es decir, efectos cuyo p-valor es menor de 0,05.

Para poder interpretar de forma conjunta los resultados de las ocho replicaciones, se ha etiquetado cada efecto o combinación de efectos con un valor que resume su tendencia hacia la significación. Un efecto o combinación de efectos tiene una tendencia significativa si ha resultado significativo entre el 67%-100% de las replicaciones (6-8), tiene tendencia ambigua si es significativo entre el 34%-66% de las replicaciones (3-5) y tiene una tendencia no significativa si es significativo entre el 0%-33% de las replicaciones (0-2).

La Tabla 4 muestra los resultados del etiquetado. Discutiremos los efectos con tendencia significativa: técnica, y las combinaciones técnica*programa y programa*falta.

La Fig. 1. Muestra el valor estimado para la media de efectividad para cada **técnica** en cada replicación. El análisis estadístico muestra que:

- En ningún caso las técnicas llegan al 100% de efectividad. La efectividad media entre replicaciones es del 79,26% para partición en clases de equivalencia, 78,43% para cobertura de decisión y 47,23% para la lectura de código.
- La lectura de código es menos efectiva que partición en clases de equivalencia y cobertura de decisión en todas las replicaciones que ejercitan la técnica estática.
- Partición en clases de equivalencia y cobertura de decisión tienen una efectividad similar, excepto en el caso de ORT'05.
- Hay valores de efectividad bajos[1], señalados con un círculo en la Fig. 1. : partición en clases de equivalencia en UdS05 y UPV05, y cobertura de decisión en UPV05 y ORT05. Esto podría indicar algún tipo de influencia de la localización en la efectividad de la técnica.

[1] Para detectar valores anormalmente altos/bajos en una replicación, establecemos valores de referencia para la media y desviación típica basada en los resultados de las 5 replicaciones UPM (no ha habido variación contextual entre estas replicaciones). Siguiendo el procedimiento para identificar valores atípicos [55], aquellos valores que no caen en el rango establecido por la media ±3 desviaciones típicas, se consideran distintos (altos/bajos) a los de referencia.

Tabla 3. P-valores del ANOVA para cada replicación.

	UPM01	UPM02	UPM03	UPM04	UPM05	UdS05	UPV05	ORT05
Modelo	0,000	0,005	0,004	0,014	0,002	0,008	0,036	0,004
Potencia del modelo	1,000	0,993	0,995	0,979	0,998	0,982	0,862	0,992
Programa	0,041	0,390	0,163	0,907	0,002	0,009	0,026	0,003
Tecnica	0,000	0,000	0,000	0,000	0,000	0,000	0,765	0,003
Version	0,153	0,095	0,123	0,323	0,361	0,070	0,387	0,635
Falta	0,071	0,316	0,882	0,009	0,215	0,046	0,013	0,000
Programa*Falta	0,000	0,041	0,057	0,002	0,000	0,023	0,021	0,001
Tecnica*Falta	0,001	0,269	0,032	0,472	0,028	0,037	0,255	0,264
Version*Falta	0,491	0,152	0,044	0,361	0,670	0,078	0,014	0,092
Programa*Tecnica	0,013	0,017	0,005	0,048	0,040	0,141	0,022	0,014
Programa*Version	0,295	0,590	0,811	0,156	0,061	0,760	0,054	0,710
Tecnica*Version	0,074	0,539	0,459	0,904	0,236	0,034	0,039	0,221
Programa*Tecnica*Falta	0,094	0,564	0,102	0,113	0,060	0,172	0,066	0,028
Programa*Version*Falta	0,125	0,708	0,242	0,075	0,245	0,212	0,412	0,019
Tecnica*Version*Falta	0,348	0,888	0,847	0,779	0,674	0,210	0,516	0,309
Programa*Tecnica*Version	0,410	0,646	0,752	0,800	0,275	0,230	0,999	0,896

Tabla 4. Tendencia a la significación.

Significativo (67%-100%)	Ambiguo (34%-66%)	No significativo (0%-33%)
Tecnica Tecnica*Programa Programa*Falta	Programa Falta Tecnica*Falta	Version Programa*Version Tecnica*Version Falta*Version Interacciones de tercer orden

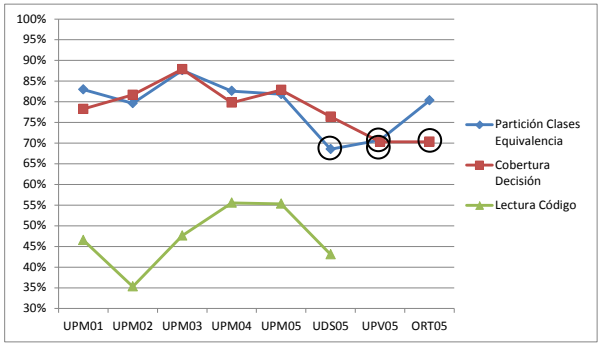

Fig. 1. Valores de la efectividad para la técnica.

La Fig. 2. muestra el valor estimado para la media de efectividad al 95% de nivel de confianza para cada **técnica por programa** en cada replicación. La interacción tecnica*programa ha resultado ser significativa en todas las replicaciones excepto en UdS05. La interacción es ordinal (los efectos de un tratamiento no son iguales para todos los niveles del otro tratamiento, pero siempre van en la misma dirección [20]), lo que significa que puede ser interpretada. El análisis estadístico muestra que:

Cmdline	Nametbl

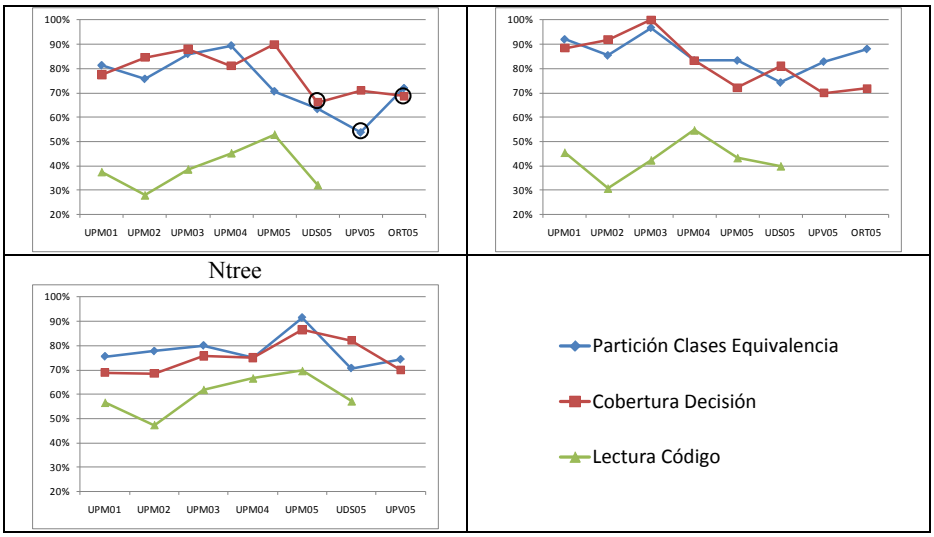

Fig. 2. Valores de efectividad para la técnica por programa.

- La lectura de código se comporta peor que partición en clases de equivalencia y cobertura de decisión para todos los programas. Más concretamente, la lectura de código se comporta mucho peor para cmdline y nametbl. Para ntree, sin embargo, la lectura de código se comporta solamente algo peor.
- Partición en clases de equivalencia y cobertura de decisión se comportan igual para todos los programas, excepto en tres casos (cmdline en UPM05 and UPV05, y nametbl en ORT05).
- Hay valores bajos de efectividad en cmdline, señalados con un círculo en la Fig. 2. : cobertura de decisión en UdS05 y ORT05, y partición en clases de equivalencia en UPV05.

La interacción programa*falta ha resultado ser no ordinal (las diferencias entre los niveles de un tratamiento cambian, dependiendo de cómo se combinan con los niveles de otro tratamiento [20]), lo que hace imposible su interpretación. La interacción no ordinal deja patente que la misma falta en distintos programas no se comporta igual. A pesar de ello, hemos intentado establecer un patrón de comportamiento para la interacción en las distintas replicaciones, pero no ha sido posible. La única explicación para esta interacción es que la clasificación de faltas usada en el experimento no es discriminante. A pesar de escoger cuidadosamente faltas del mismo tipo (por ejemplo: omisión, control), no se comportan igual entre programas para las técnicas. Las técnicas son capaces de detectar el mismo tipo de defectos en algunos casos y no en otros, sin explicación aparente relacionada con el tipo de falta.

La Tabla 5 resume los resultados obtenidos.

6 Diferencias en los Contextos

Del análisis de las variaciones ocurridas en las distintas replicaciones, hemos encontrado las siguientes variables que podrían estar influyendo en los resultados obtenidos:

- *Copia.* En cada sesión experimental en UdS y UPV, los sujetos aplican la misma técnica, pero distintos sujetos trabajan con distintos programas. Los sujetos podrían intercambiar información acerca de los programas y sus faltas al final de la primera sesión. Esto podría

producir una mayor efectividad en las técnicas aplicadas en la segunda y tercera sesión en UdS y UPV que en otras replicaciones.

Tabla 5. Resumen de resultados.

	UPM	UdS	UPV	ORT
Tech.	(PCE=CD) >LC	(PCE=CD) > LC	PCE=CD	PCE>CD
		PCE↓	PCE↓, CD↓	CD↓
Tech. x Prog.	(PCE=CD) > LC ntree: LC↑ UPM05: CD>PCE	(PCE=CD) > LC ntree: LC↑	cmdline: PCE<CD nametbl, ntree: PCE=CD	cmdline: PCE=CD nametbl: PCE>CD
		cmdline: CD↓	cmdline: PCE↓	cmdline: CD↓

Leyenda: ↓ disminuye, ↑ incrementa, = igual, > mejor, < peor

- *Experiencia en programación.* Cobertura de decisión genera casos de prueba a partir del código fuente, y partición en clases de equivalencia los genera a partir de la especificación. En ORT, los sujetos no tienen apenas experiencia en programación. Esto podría producir una menor efectividad de cobertura de decisión en ORT que en otras replicaciones.
- *Cansancio*. En ORT, el experimento transcurre en una única sesión, en la que los sujetos aplican las dos técnicas. Los sujetos podrían sufrir cansancio al aplicar la segunda técnica. Aunque este efecto se cancela considerando todos los posibles órdenes de aplicación, podría causar una menor efectividad de las técnicas en ORT que en otras replicaciones.
- *Trabajo en parejas.* En UdS, las técnicas se aplican en parejas. Esto podría causar una mayor efectividad de las técnicas en UdS que en otras replicaciones.

De la reunión posterior a la realización de las replicaciones que tuvieron lugar entre los investigadores encargados de la replicación, se identificaron las siguientes variables que podrían estar influyendo en los resultados obtenidos. Para una descripción detallada de cómo se han obtenido todas estas variables, ver [47]:

- *Formación.* En UdS y UPV, los sujetos están familiarizados con las técnicas y por tanto reciben únicamente un tutorial. En UPM y ORT, los sujetos reciben un curso completo. Sin embargo, nos dimos cuenta de que no se consiguió que los sujetos tuviesen el mismo nivel de conocimiento de las técnicas. Esto podría causar una menor efectividad de las técnicas en UdS y UPV que en otras replicaciones.
- *Motivación.* Los sujetos en UdS y UPV no están tan motivados, ya que el experimento no tiene apenas impacto en la nota de la asignatura. En UPM y ORT, el experimento constituye la nota de la asignatura. Esto podría causar una menor efectividad de las técnicas en UdS y UPV que en otras replicaciones.
- *Presión.* Los sujetos de UPV no tuvieron suficiente tiempo para aplicar cobertura de decisión. Fue la primera técnica que aplicaron, y se utilizó parte del tiempo en explicar la dinámica del experimento a los sujetos. Por tanto, los sujetos trabajaron bajo presión para poder finalizar la tarea a tiempo. Esto podría causar una menor efectividad de cobertura de decisión en UPV que en otras replicaciones.

La Tabla 6 muestra, para cada localización, los valores de las posibles variables influyentes.

Dependiendo del tipo de influencia que estas variables tienen en los resultados, pueden agruparse en tres categorías: variables que influyen en todas las técnicas, variables que influyen a una determinada técnica, y variables que no influyen. Las variables que influyen en todas las técnicas o que no tienen influencia permiten dar recomendaciones generales acerca de cómo aplicar las técnicas de V&V para obtener un mayor rendimiento. Las variables que influyen una determinada técnica, permite dar recomendaciones de cara a la selección de la(s) técnica(s) más apropiada(s) para un determinado contexto de proyecto.

Tabla 6. Variables potencialmente influyentes.

	UPM	UdS	UPV	ORT
Formación	Curso	Tutorial	Tutorial	Curso
Motivación	Alta	Baja	Baja	Alta
Presión	No	No	Sí (CD)	No
Copia	No	Sí	Sí	No
Experiencia en programación	Sí	Sí	Sí	No
Cansancio	No	No	No	Sí
Trabajo	Individual	Parejas	Individual	Individual

7 Interpretación de Resultados

A continuación agrupamos la interpretación de resultados atendiendo a su origen: resultados obtenidos y combinación de resultados obtenidos con variables contextuales.

7.1 Resultados Obtenidos

Hemos encontrado evidencias estadísticas acerca de:

- **No hay diferencia en la efectividad de partición en clases de equivalencia y cobertura de decisión**. La efectividad es igual en siete replicaciones (siendo la excepción ORT'05).
- **La efectividad de partición en clases de equivalencia y cobertura de decisión aumenta si distintos sujetos aplican la misma técnica**. Ninguna técnica consigue alcanzar el 100% de efectividad (probabilidad teórica). Adicionalmente, la efectividad de las técnicas es independiente del tipo de falta en todas las replicaciones. Sin embargo, los sujetos no detectan las mismas faltas, ya que todas las faltas son detectadas por algunos sujetos.
- **Es más efectivo utilizar técnicas dinámicas que lectura de código**. La técnica de lectura de código se comporta peor que partición en clases de equivalencia y cobertura de decisión en todas las replicaciones (seis) en las que ha sido evaluada.
- **La efectividad de las faltas depende del programa**. El programa es significativo en siete de las ocho replicaciones, aunque no hemos podido encontrar un patrón de comportamiento. Esto sugiere que la clasificación de faltas utilizada no es discriminante, ya que faltas supuestamente idénticas (todas las F1, F2, etc.) no se han comportado igual para todos los programas. Es necesario otro esquema de clasificación de faltas para estudiar la efectividad.
- **El programa influye en el comportamiento de la técnica de lectura de código**. La técnica de lectura de código, es más efectiva con ntree que con los otros dos programas, en las seis replicaciones que se examina. Un análisis detallado de ntree no revela ninguna característica especial. De hecho, es cmdline quien podría comportarse de manera distinta, ya que nametbl y ntree son similares en muchos aspectos. La diferencia podría estar relacionada con las faltas de ntree, que, por algún motivo desconocido, son más fáciles de detectar que otras. Es sorprendente que no se hayan obtenido valores similares de efectividad para la técnica de lectura de código, incluso para programas extremadamente similares. Esto sugeriría que la técnica es muy sensible a características propias de los programas o las faltas.

7.2 Resultados Obtenidos y Contexto

Hay signos (no obtenidos estadísticamente) de que:

- **La experiencia en programación disminuye la efectividad de cobertura de decisión.** En

ORT05, cobertura de decisión es menos efectiva que partición en clases de equivalencia. Los sujetos de ORT05 no tiene experiencia en programación. Esta diferencia es mayor para el programa cmdline, que es el programa con mayor complejidad ciclomática.

- **El conocimiento de la técnica podría afectar su efectividad.** La formación de los sujetos en UPV05 y UdS05 no es tan completa como en UPM y ORT05. Esto podría explicar por qué ambas técnicas son menos efectivas en UPV05, y partición en clases de equivalencia es menos efectiva en UdS05.
- **Las técnicas podrían ser menos efectivas si los sujetos no están motivados.** Los sujetos de UPV05 y UdS05 están menos motivados que los de UPM y ORT05. Esto podría explicar por qué partición en clases de equivalencia y cobertura de decisión son menos efectivas en UPV05, y partición en clases de equivalencia es menos efectivas en UdS05.
- **Las técnicas son más efectivas si se trabaja en parejas.** A pesar de una posible influencia de la motivación y la formación, partición en clases de equivalencia se comporta peor únicamente en UdS05. Esto podría indicar que el efecto del trabajo en pareja cancela los posibles efectos de la motivación y la formación para cobertura de decisión y lectura de código (cosa que no ocurre con partición en clases de equivalencia).
- **El trabajo bajo presión no parece influir en la efectividad de cobertura de decisión.** Partición en clases de equivalencia y cobertura de decisión tienen una efectividad similar en UPV05. Si el trabajo bajo presión hubiese influido, cobertura de decisión debería haber sido menos efectiva que partición en clases de equivalencia, ya que fue aplicada bajo presión.
- **El cansancio no parece influir en la efectividad de la técnica.** Si hubiese existido esta influencia, las técnicas deberían haber sido menos efectivas en ORT05.
- **Tener información previa del programa no parece influir en la efectividad de la técnica.** En UPV05 y UdS05 los sujetos pudieron intercambiar información sobre los programas después de la primera sesión. Si tener esa información hubiese influido, partición en clases de equivalencia debería haberse comportado mejor que cobertura de decisión en UPV05 y UdS05, y la lectura de código mejor que las técnicas dinámicas en UdS05.

8 Amenazas a la Validez

Nuestro estudio tiene amenazas a la validez que impiden la generalización de sus resultados

- Los sujetos son alumnos. No podemos generalizar los resultados a profesionales.
- Están limitados a las características de los programas. Hemos usado programas pequeños en C. El comportamiento de las técnicas podría variar con el tamaño, lenguaje de programación, etc.
- Están limitados a la aplicación concreta de las técnicas. Cobertura de decisión se aplica sin utilizar un analizador dinámico de código.
- Las variables identificadas que podrían influir en los resultados (formación, motivación, trabajo en parejas y experiencia en programación) deberían ser exploradas en experimentos futuros para comprobar si dicha influencia existe o no.

9 Conclusiones

Hemos realizado ocho replicaciones del mismo experimento (en distintos entornos) para intentar entender la efectividad de tres técnicas de V&V: dos técnicas dinámicas (partición en clases de equivalencia y cobertura de decisión) y una técnica estática (lectura de código mediante abstracciones sucesivas). Hemos interpretado de forma conjunta los resultados obtenidos, y realizado un estudio de diferencias contextuales entre las distintas localizaciones en las que se realizaron las replicaciones.

La interpretación de los resultados ha mostrado que partición en clases de equivalencia y cobertura de decisión son igualmente efectivas y más efectivas que la lectura de código. Igualmente, hemos descubierto que la efectividad de partición en clases de equivalencia y cobertura de decisión es independiente del tipo de falta. Esto sugiere la posibilidad de combinar sujetos aplicando la misma técnica para incrementar la efectividad de las técnicas.

Por otra parte, hemos encontrado variables contextuales que podrían influir en los resultados. Formación, motivación y trabajo en parejas podrían influir en la técnica, y la experiencia en programación podría influir en la efectividad de cobertura de decisión. Trasladando estos resultados a la industria, la formación, motivación y trabajo en parejas podría mejorar la efectividad de las pruebas, mientras que la experiencia en programación podría influir a la hora de elegir entre partición en clases de equivalencia y cobertura de decisión.

Más experimentos con otras técnicas podrían mostrar si el comportamiento de partición en clases de equivalencia y cobertura de decisión pueden ser generalizados a pruebas de caja negra y de caja blanca.

10 Agradecimientos

Trabajo financiado por el Ministerio de Ciencia e Innovación, proyecto TIN2008-00555.

11 Referencias

[1] V. R. Basili, S. Green, O. Laitenberger, F. Lanubile, F. Shull, S. Sorumgard, M. V. Zelkowitz, The Empirical Investigation of Perspective Based Reading. *Journal of Empirical Software Engineering*. 1(2):133–164, 1996.

[2] V.R. Basili, R.W. Selby. Comparing the Effectiveness of Software Testing Strategies. *IEEE Transactions on Software Engineering,* 13(2):1278-1296. 1987.

[3] A. Bertolino. Guide to the Knowledge area of Software Testing. *Software Engineering Body of Knowledge*. IEEE Computer Society. 2004.

[4] J. Bible, G. Rothermel, D. Rosenblum. A Comparative Study of Coarse- and Fine-Grained Safe Regression Test Selection. *ACM Transactions on Software Engineering and Methodology* 10(2):149-183. 2001.

[5] J.M. Bieman, J.L. Schultz. An Empirical Evaluation (and Specification) of the All-du-paths Testing Criterion. *Software Engineering Journal*, January 1992, pp. 43–51.

[6] S. Biffl, Analysis of the Impact of Reading Technique and Inspector Capability on Individual Inspection Performance. 7th *Asia-Pacific Software Engineering Conference*, pp. 136–145. 2000.

[7] L.C. Briand, M. Penta, Y. Labiche. Assessing and Improving State-Based Class Testing: A Series of Experiments. *IEEE Transactions on Software Engineering*. 30(11):770-793. 2004.

[8] T.Y. Chen, Y.T. Yu. On the Relationships between Partition and Random Testing. *IEEE Transactions on Software Engineering*. 20(12): 977-980, 1994.

[9] L.A. Clarke, A. Podgurski, D.J. Richardson, S.J. Zeil. A Formal Evaluation of Data Flow Path Selection Criteria. *IEEE Transactions on Software Engineering*. 15(11):1318-1332, 1989.

[10] A. Dunsmore, M. Roper, M. Wood. Further Investigations into the Development and Evaluation of Reading Techniques for Object-oriented Code Inspection. *24th International Conference on Software Engineering*, pp. 47–57.2002.

[11] J.W. Duran, S.C. Ntafos. An Evaluation of Random Testing. *IEEE Transactions on Software Engineering*. SE-10(4): 438—444, 1984.

[12] S. Elbaum, A.G. Mailshevsky, G. Rothermel. Prioritizing Test Cases for Regression Testing. *International Symposium on Software Testing and Analysis*. 2000, pp. 102–112.

[13] P.G. Frankl, Y. Deng. Comparison of Delivered Reliability of Branch, Data Flow and Operational Testing, a case study. *International Symposium on Software Testing and Analysis*. 2000.

[14] P. Frankl, O. Iakounenko. Further Empirical Studies of Test Effectiveness. *International Symposium on Foundations on Software Engineering*. Pp 153-162. 1998

[15] P.G.Frankl, S.N. Weiss. An Experimental Comparison of the Effectiveness of Branch Testing and Data Flow Testing. *IEEE Transactions on Software Engineering*, 19(8):774-787. 1993.

[16] P.G. Frankl, S.N. Weiss, C. Hu. All-Uses vs Mutation Testing: An Experimental Comparison of Effectiveness. *Journal of Systems and Software*, 38:235-253. 1997.

[17] P.G. Frankl, E.J. Weyuker. Assessing the Fault--Detecting Ability of Testing Methods. *International Conference on Software for Critical Systems*, pp. 77—91. 1991.

[18] O.S. Gómez, N. Juristo, S. Vegas. Replication, Reproduction and Re-analysis: Three ways for verifying experimental Findings. *International Symposium on Workshop on Replication in Empirical Software Engineering Research*. 2010.

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

[20]J.F. Hair, W.C Black, B.J. Babin, R.E. Anderson, R.L. Tatham. *Multivariate Data Analysis 6th edition*. Prentice Hall. 2006.

[21] R. Hamlet. Probable correctness theory. *Information Processing Letters*. 25:17-25, 1987.

[22] D. Hamlet, R. Taylor. Partition Testing does not Inspire Confidence. *IEEE Transactions on Software Engineering*. 16(12): 1402—1411, 1990.

[23] M. Hutchins, H. Foster, T. Goradia, T. Ostrand. Experiments on the Effectiveness of Dataflow- and Controlflow-Based Test Adequacy Criteria. *16th International Conference on Software Engineering*. Pp. 191-200. 1994.

[24] Juristo N, Moreno A.M. *Basics of Software Engineering Experimentation*. Kluwer. 2001.

[25] N. Juristo, A.M. Moreno, S. Vegas, M. Solari. In Search of What We Experimentally Know about Unit Testing. *IEEE Software*. 23(6):72-80. 2006.

[26] E. Kamsties, C.M. Lott. An Empirical Evaluation of Three Defect-Detection Techniques. *Fifth European Software Engineering Conference*. 1995.

[27] J.M. Kim, A. Porter, G. Rothermel. An Empirical Study of Regression Test Application Frequency. *22nd International Conference* on *Software Engineering*. Pp.126-135. 2000.

[28] H. Lee, Y. Ma, Y. Kwon. Empirical Evaluation of Orthogonality of Class Mutation Operators. *11th Asia-Pacific Software Engineering Conference*. 2004, pp. 512–518.

[29] D. Leon, W. Masri, A. Podgurski. An Empirical Evaluation of Test Case Filtering Techniques Based on Exercising Complex Information Flows. *27th International Conference on Software Engineering*. 2005.

[30] R.C. Linger, H.D. Mills, B.I. Witt. *Structured Programming: Theory and Practice*. Addison-Wesley, 1979.

[31] J. Maldonado, J. Carver, F. Shull, S. Fabbri, E. Dória, L. Martimiano, M. Mendonça, V. Basili. Perspective-based Reading: A Replicated Experiment Focused on Individual Reviewer Effectiveness. *Empirical Software Engineering*. 11(1):119–142, 2006.

[32] G.J. Myers. A Controlled Experiment in Program Testing and Code Walkthroughs/Inspections. *Communications of the ACM*. 21(9): 760-768, 1978.

[33] Myers G.J. T. Badgett, C. Sandler. *The Art of Software Testing*. Wiley-Interscience. 2nd edition. 2004.

[34] S. Ntafos. A comparison of some structural testing strategies. *IEEE Transactions on Software Engineering*. 14(6):368-874, 1988.

[35] S.C. Ntafos. On Random and Partition Testing. *International Symposium on Software Testing and Analysis*, pp. 42-48. 1998.

[36] A.J. Offut, S.D. Lee. An Empirical Evaluation of Weak Mutation. *IEEE Transactions on*

Software Engineering, 20(5):337-344. 1994.

[37] A.J. Offut, A. Lee, G. Rothermel, RH. Untch, C. Zapf. An Experimental Determination of Sufficient Mutant Operators. *ACM Transactions on Software Engineering and Methodology,* 5(2):99-118. 1996.

[38] A. Porter, L. Votta, V.R. Basili. Comparing Detection Methods for Software Requirements Inspection: A Replicated Experiment. *IEEE Transactions on Software Engineering*, 21(6): 563–575, 1995.

[39] S. Rapps, E. Weyuker. Selecting Software Test Data Using Data Flow Information. *IEEE Transactions on Software Engineering*. SE-11(4):367-375, 1985.

[40]M. Roper, M. Wood, J. Miller. An empirical evaluation of defect detection techniques. *Information and Software Technology*. 39:763-775. 1997.

[41] D.S. Rosenblum, E.J. Weyuker. Using Coverage Information to Predict the Cost-Effectiveness of Regression Testing Strategies. *IEEE Transactions on Software Engineering*. 23(3):146—156, 1997.

[42] G. Rothermel, S. Elbaum, A.B. Malishevsky, P. Kallakuri, X. Qiu. On Test Suite Composition and Cost-Effective Regression Testing. *ACM Transactions on Software Engineering and Methodology*. 13(3):277-331. 2004.

[43] G. Rothermel, M.J. Harrold. Analyzing Regression Test Selection Techniques. *IEEE Transactions on Software Engineering*. 22(8):529-551, 1996.

[44] G. Rothermel, M.J. Harrold. Empirical Studies of a Safe Regression Test Selection Technique. *IEEE Transactions on Software Engineering*. 24(6):401-419. 1998.

[45] G. Rothermel, R.H. Untch, C. Chu, M.J. Harrold. Test Case Prioritization: An Empirical Study. *International Conference on Software Maintenance*. Pp. 179-188. 1999.

[46] T. Thelin, P. Runeson, C. Wohlin, T. Olsson, C. Andersson. Evaluation of Usage-Based Reading—Conclusions after Three Experiments. *Empirical Software Engineering*. 9:77–110, 2004.

[47] S. Vegas, N. Juristo, A.M. Moreno, M. Solari, P. Letelier. Analysis of the Influence of Communication between Researchers on Experiment Replication. *International Symposium on Empirical Software Engineering*. Pp. 28-37. 2006.

[48] F. Vokolos, P.G. Frankl. Empirical Evaluation of the Textual differencing Regression Testing Technique. *International Conference on Software Maintenance*. 1998.

[49] M.D. Weiser, J.D. Gannon, P.R. McMullin. Comparison of Structural Test Coverage Metrics. *IEEE Software*. 2(3):80-85, 1985.

[50] E.J. Weyuker. The Complexity of Data Flow Criteria for Test Data Selection. *Information Processing Letters*. 19(2):103-109. 1984.

[51] E.J. Weyuker. The Cost of Data Flow Testing: An Empirical Study. *IEEE Transactions on Software Engineering*. 16(2): 121–128, 1990.

[52] E.J. Weyuker, B. Jeng. Analyzing Partition Testing Strategies. *IEEE Transactions on Software Engineering*. 17(7): 703—711, 1991.

[53] W.E. Wong, J.R. Horgan, S. London, H. Agrawal. A Study of Effective Regression Testing in Practice. *8th International Symposium on Software Reliability Engineering*. Pp.264-274. 1998

[54] E. Wong, A.P. Mathur. Fault Detection Effectiveness of Mutation and Data-flow Testing. *Software Quality Journal*, 4:69-83. 1995.

[55] T.W Wright. A Treatise on the Adjustment of Observations by the method of Least Squares. Van Nostrand. 1884.

[56] S.J. Zeil. Selectivity of Data--Flow and Control--Flow Path Criteria. *2nd Workshop on Software Testing, Verification and Analysis*, pp. 215—222. 1988.