IMPACTO EN LA CALIDAD DE LA FASE DE PRUEBAS CON EL USO DE TÉCNICAS FORMALES*

Gabriel Huecas

Dpt. Informática
Univ. Alfonso X el Sabio
gabriel@uax.es

José A. Mañas
Tomás Robles

Dpt. Ing. Telemática
E.T.S.I. Telecomunicación
jmanas@dit.upm.es

4 de abril de 1997

Resumen

El surgimiento de normas por parte de organismos normalizadores de Servicios y Sistemas de Comunicaciones, como ISO e ITU supone un cambio en la mentalidad tanto en el desarrollo como en la producción para los fabricantes de sistemas de comunicaciones; normas que los fabricantes deben cumplir y organismos independientes deben certificar u homologar.

En este entorno, las Técnicas de Descripción Formal son un mecanismo clave para el diseño y especificación de dichos protocolos.

En este artículo se presentan las áreas de impacto donde las FDTs suponen una mejora en la calidad y productividad. Para ello, se conceptualiza y formaliza el proceso de ejecución de Pruebas de Conformidad y elementos integrantes en las arquitecturas de pruebas. Además, se definen y formalizan los componentes de las pruebas.

Todo ello se contextualiza en el entorno definido por las normas 9000 de ISO, donde se definen los parámetros que definen la calidad de productos ISO. Para ello, se describe el impacto de esta formalización y el uso de las FDTs en los parámetros que definen la calidad de productos ISO.

*Este trabajo ha sido parcialmente subvencionado por el proyecto E-MEDAS

1 Introducción

En la actualidad, las Técnicas Formales de Descripción (FDTs) han tomado un papel fundamental en la descripción de sistemas de forma precisa, concisa y sin ambigüedades. Como resultado de ello se disponen de Especificaciones Formales de sistemas de telecomunicación que pueden ser tratadas automáticamente por herramientas software. En principio, estas especificaciones tienen como objeto mejorar la fase de diseño y facilitar el entendimiento entre el cliente y el diseñador. Sin embargo, el hecho de disponer de tales especificaciones ha generado multitud de teorías, metodologías y algoritmos que intentan explotarlas en los campos de Generación y Ejecución de Pruebas, Entornos Transformacionales, Simuladores, Generadores de Prototipos. Esto ha provocado un gran impacto en el mercado de servicios y sistemas, que evoluciona hacia un soporte automatizado en todas las fases del ciclo de vida.

Particular interés tiene la fase de ejecución de pruebas: al terminar el producto, éste se ve sometido a una serie de pruebas con diferentes objetivos: por un lado, se desea que el producto cumpla unos mínimos de calidad. Por otro lado, el cliente está interesado en comprobar que dicho producto cumple cierto requisitos, normalmente, que se atienen a una cierta norma. Asimismo, se ejecutan pruebas no funcionales, como pruebas de prestaciones, robustez, etc. En este artículo nos vamos a centrar en las denominadas *pruebas de conformidad*, cuyo objetivo es garantizar que el producto se atiene a una norma. Para lograr este objetivo, la implementación es probada respecto de su especificación de referencia. Está demostrado [Dil91] que los productos homologados tienen una alta probabilidad de *interoperar* con otros, y que esta probabilidad es menor para productos no conformes a la misma norma[1].

ISO ha normalizado una metodología y un entorno específico de ejecución de pruebas de conformidad [ISO91]. La idea básica es disponer de baterías de pruebas para cada norma, disponible públicamente para fabricantes y usuarios. Estas pruebas podrán ser desarrolladas manualmente o bien derivadas de forma automática o semi-automática a partir de la especificación formal de la norma. Puntos claves de esta norma son al *reusabilidad* y *flexibilidad* de las Baterías de Pruebas.

Por otro lado, ISO define la serie de normas 9000, con el objeto de identificar los parámetros que definen la calidad de productos e identificar que aspectos influyen en dichos parámetros.

2 Objetivos

El objetivo de este artículo es presentar un modelo de ejecución de pruebas genérico, en las que pruebas descritas mediante FDTs son ejecutadas contra productos reales. Dado que se tratan de Pruebas de Conformidad, la Implementación Bajo Pruebas (IUT) será tratada como una caja negra que se somete a experimentación. A cada ejecución de una prueba se le asigna un veredicto que determina si se cumplió el propósito para el cual fue diseñada.

El sistema utilizará la especificación de referencia con el propósito de poder informar al fabricante de los errores hallados, expresados en términos de dicha especificación. Ahora bien, el fabricante es libre de elegir la forma en que se accede a su producto. Por lo tanto, como paso previo, se debe conseguir un modelo operativo de cada ejecución, que abstraiga los elementos comunes de la ejecución de pruebas y que minimice la adaptación del sistema de ejecución a cada nueva IUT. De esta forma, se mejora la *reusabilidad* y la *fiabilidad* de la fase de ejecución de pruebas.

[1] Evidentemente, ni siquiera se intenta que interoperen productos que no han sido fabricados para ello.

3 Entorno y Metodología OSI

El proceso de garantizar la conformidad de una implementación respecto de su especificación propuesto por ISO se muestra en la figura 1.

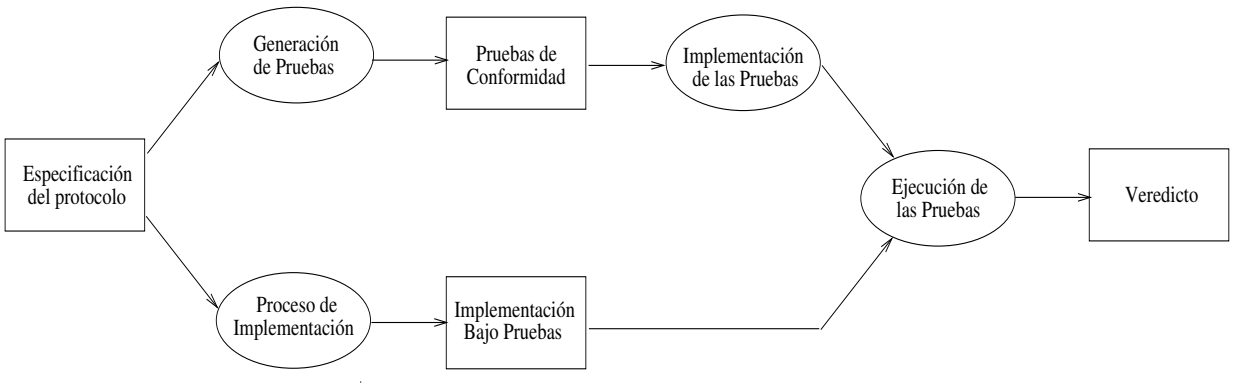

Figura 1: Proceso de Certificación de Conformidad

La primera fase es generar una batería de pruebas (*Test Suite*) para un protocolo (OSI) en particular. Esta fase se la conoce como Generación o Derivación de Pruebas. Existen serios intentos de automatizar esta fase, como por ejemplo [Rob91]. La Batería de Pruebas generada es abstracta, en el sentido de que es completamente independiente de la implementación. En ISO, típicamente esta batería estará expresada en el lenguaje TTCN [ISO91].

Como esta batería es genérica, habrá una subfase de *Selección de Pruebas*, puesto que el fabricante habrá optado por una serie de opciones de la norma, dejando de lado otras. Esta información la proporciona el fabricante en un documento conocido como PICS[2].

La segunda fase consiste en la realización de los medios de ejecución de pruebas, es decir, *Implementar las Pruebas*. Los *Casos de Prueba* son transformados en *Pruebas Ejecutables*, que serán particulares para cada sistema de pruebas. Esta particularización dependerá de los detalles de implementación de los eventos representados de forma abstracta en la especificación de referencia. Existe un documento, llamado PIXIT[3], donde el fabricante incluye toda la información necesaria para poder ejecutar estas pruebas. En definitiva, este documento describe las interfaces físicas del producto, así como detalles de codificación de datos, etc.

La última fase es la *Ejecución de Pruebas*. El producto es ejercitado de acuerdo a los *Casos de Prueba Ejecutables*, con el objetivo de emitir un *veredicto*. Este veredicto determina la consecución del objetivo para el cual se desarrolló la prueba. El veredicto puede ser de **PASA** si el objetivo se ha cumplido, **FALLA** si no se cumple e **INCONCLUSO** si no es ninguno de los anteriores[4]. Este resultado es documentado en un *Informe de Pruebas de Conformidad del Protocolo*.

[2]Protocol Implementation Conformance Statement [ISO91].

[3]Protocol Implementation eXtra Information for Testing [ISO91].

[4]Este caso puede darse cuando en un sistema se ejecuta una secuencia *legal* de acciones pero que aparta al sistema de *verificar* la consecución del objetivo de la prueba.

4 Casos de Prueba

La obtención de la batería de pruebas es una fase previa orientada a organizar técnica y administrativamente la realización de las pruebas, con el objeto de facilitar el *mantenimiento* de las mismas.

Los casos de prueba constan de un **preámbulo**, un **cuerpo** y un **postámbulo** [ISO91]. El preámbulo es una fase preliminar en la que la implementación se lleva al estado en el que se quiere realizar la prueba. En pruebas muy sencillas, este preámbulo es nulo. El cuerpo es el objeto en sí de la prueba. Los preámbulos de pruebas complejas son el cuerpo de pruebas preliminares. Por último, el postámbulo es el conjunto de acciones que nos permiten llevar la implementación a un estado reconocible tras una prueba. A veces es tan simple como un botón de *reset*.

De acuerdo con la definición de conformidad, solamente trazas contempladas en la especificación serán probadas sobre el producto. La selección de casos de prueba se reduce a extraer trazas posibles de la especificación de referencia.

4.1 Propósitos de Prueba

Las pruebas se organizan por propósitos. Estos consisten en identificar un estado inicial, al que llega gracias al preámbulo, y un objetivo que es el cuerpo de la prueba.

> Un propósito de prueba es *"una descripción en prosa de un objetivo de prueba definido con precisión, centrado en un único requisito de conformidad, de acuerdo con la especificación de la norma correspondiente"* [ISO91]

5 Ejecución de pruebas

La ejecución de una prueba consiste en evolucionar en paralelo la IUT y la prueba. Hay que tener en cuenta, además de los aspectos operacionales, que mientras que la IUT sólo ejecuta una posible secuencia de acciones, sobre la prueba se pueden seguir diferentes trazas en un momento dado. Evidentemente, si la prueba está bien diseñada, será posible decidir, tarde o temprano, cuál de las trazas está siendo realmente observada.

Por lo tanto, el Ejecutor de Pruebas debe encargarse de decidir si estimula o espera la reacción de la IUT. Si la prueba ha terminado, emitirá el veredicto adecuado. En caso contrario, deberá determinar cual es el nuevo evento a ofertar/observar. Por ello tendrá que seguir los diferentes estados de la prueba. Además, tendrá en cuenta aspectos organizativos como la división entre preámbulo y cuerpo de la prueba, y el posible bloqueo de la IUT.

5.1 Temporizadores

Hay que prever que la implementación falle y que durante las pruebas algún evento previsto en las trazas no se lleve a cabo. La decisión de que un evento no ocurre debe tomarse en base al vencimiento de un temporizador: "si no ocurre en T segundos, considérese que no va a ocurrir nunca". Esta información sobre tiempos de espera va estrechamente asociada a los propósitos de prueba, pudiéndose expresar bien en términos globales (idéntico criterio para todos los eventos) o particulares (usando temporizadores específicos para cada caso de prueba o, incluso, para algunos eventos).

5.2 Clases de Pruebas

En la fase de generación de la batería de pruebas se hace un análisis exhaustivo de la especificación buscando trazas que recorran los objetivos de pruebas. De este análisis resultan los casos de prueba clasificados en tres clases:

obligatorios.- Una cierta traza de prueba es de obligado cumplimiento. Es el grupo ideal y más simple de tratar. U ocurre, o se rechaza el producto.

opcionales.- Una cierta traza de prueba es opcional, siendo decisión del fabricante implementarla o no. Resultan casos de prueba parametrizados por valores que sólo se conocerán al ir a pasar las pruebas. Estos valores los proporcionará el fabricante en un documento conocido como PICS[5]. El operador introducirá los valores concretos del PICS para un cierto producto. Un cierto número de pruebas puede quedar anulado en base a estos valores. Típicamente, las pruebas asociadas a cierto parámetro del PICS suelen estar agrupadas.

indeterministas.- Es muy frecuente en protocolos de comunicaciones que un cierto objetivo no se pueda alcanzar, sin que ello implique un error en la implementación. Por ejemplo, una desconexión en un protocolo orientado a conexión, puede impedir que se observe la entrega de un dato; pero la desconexión iniciada por el medio es un comportamiento correcto contemplado en la norma[6].

Debido a comportamientos no deterministas de la especificación, puede ocurrir que, al intentar ejecutar la prueba, el producto actúe de forma legal (según lo especificado) pero apartándose del propósito de la prueba. La prueba contempla estos indeterminismos dirigiendo la implementación a un estado estable pero, evidentemente, no se puede asegurar que el propósito ha sido cumplido. Por ello, se clasifican las pruebas dependiendo de si se puede asegurar el cumplimiento de su objetivo. Esta clasificación, ampliada de [dNH84] es:

- **MUST ACCEPT** La especificación es determinista en el comportamiento referente al objetivo de la prueba, por lo que la IUT debe aceptar, en todo momento, los eventos de la misma.
- **MAY ACCEPT** La especificación admite comportamientos que se apartan del objetivo de la prueba, por lo que su terminación puede no asegurar el cumplimiento de su propósito.
- **MUST REJECT** Igual que MUST, pero la intención es que se rechace.
- **MAY REJECT** Igual que MAY, pero con la intención de que se rechace.

5.3 Relación entre la formalización y su concreción

Todo el mecanismo de descripción formal se basa en abstraer detalles concretos de la realidad, darles un nombre simbólico, y manipular este mundo de símbolos. Todo esto es muy cómodo para

[5]Protocol Implementation Conformance Statement [ISO91].

[6]Otro tipo de pruebas, por ejemplo las de prestaciones, pueden fallar si, por ejemplo, el número de desconexiones iniciadas por el medio es escandalosamente elevado, impidiendo la transferencia efectiva de información. Pero aquí nos estamos centrando en pruebas de conformidad, que no entran en aspectos de prestaciones.

realizar manipulaciones simbólicas sobre la especificación; pero tiene un límite que se alcanza cuando hay que interactuar con un producto.

La correlación entre el mundo abstracto y el concreto se realiza, en nuestro sistema, en base a un mecanismo de anotaciones[7] [MdM89] que permite asociar aspectos externos a eventos simbólicos. Hay básicamente dos casos a contemplar: eventos que ocurren en el mundo porque la especificación así lo desea (salientes), y elementos que ocurren en el mundo y la especificación constata que han acaecido (entrantes). ISO ha propuesto un lenguaje de especificación de pruebas (TTCN [ISO91]) que contempla esta clasificación explícitamente. Los eventos salientes se denotan por `!`, y los entrantes por `?`. Asimismo, SDL y ESTELLE definen los eventos que son salientes y entrantes. Desgraciadamente, LOTOS abstrae este "detalle", lo que requiere que sea introducido posteriormente.

5.4 Imposición versus Observación

¿Quién decide qué evento ocurre? ¿El probador o el producto sujeto a pruebas? La respuesta depende de la estructura de la especificación. Esta ya fue analizada durante la fase de generación de pruebas, y la información pertinente consta en el caso de prueba. Si en un estado dado el caso sólo prevé un posible evento, la respuesta es simple: no hay nada que elegir, u ocurre ese evento o algo ha fallado (se suele decir que hay bloqueo, aspecto que trataremos más adelante). Si el caso de prueba permite una decisión indeterminista, el probador debe estar preparado para observar cualquiera de las posibilidades y emitir el veredicto correspondiente.

5.5 Bloqueo

Los casos de prueba sólo contemplan terminaciones exitosas o inconclusas. Pero en cualquier momento puede ocurrir cualquier evento no previsto o, más precisamente, que ninguno de los eventos previstos ocurran dentro del plazo marcado en el propósito de prueba. Cuando un temporizador salta decimos que se ha producido un **bloqueo**.

Un bloqueo puede ocurrir durante la ejecución del preámbulo o durante la ejecución del cuerpo de la prueba. En el primer caso se asigna un veredicto de **INCONCLUSO**, siguiendo la costumbre de los centros de homologación y las recomendaciones de la ISO [ISO91]. Esta asignación es arbitraria y, usualmente, conlleva la repetición de la prueba un cierto número de veces. En cuanto pasa una vez, se considera pasada y no se repite más; pero si al cabo de un número predeterminado de intentos sigue fallando el preámbulo, se le asigna un veredicto final de **INCONCLUSO**. Esto también es arbitrario; pero sigue siendo práctica habitual.

Muy diferente resulta la situación en la que el probador se bloquea durante la ejecución del cuerpo de la prueba. En estos casos, el veredicto es inequívocamente FALLO. Se emite un informe del estado de la especificación de referencia en el momento del bloqueo, y se rechaza el producto[8].

5.6 Prueba Correcta

Definimos como *prueba correcta* aquella que cumple:

[7] Básicamente, una anotación es un comentario especial con significado para ciertas herramientas.

[8] A efectos prácticos, puede que se intenten algunas pruebas más para aprovechar la sesión de homologación e intentar descubrir más errores antes de devolver el producto al fabricante; pero pase lo que pase con cualquier otra prueba, el producto será rechazado.

- Cada caso de prueba debe incluir información acerca de la consecución de su propósito. Éste debe ser único.
- La terminación de una prueba irá marcada según el comportamiento esperado: **PASA** o **FALLA**. Estos casos corresponden a pruebas de aceptación o de rechazo. Se incluye la terminación de **INCONCLUSO** para tratar el posible no determinismo de la especificación.
- En cada estado de la prueba o bien existe un único evento *imponible* o uno o varios *observables*.
  - **imponibles**: un evento imponible es aquel que la prueba trata de forzar su ejecución por la implementación. Si ésta lo rechaza se considera que le ejecución ha sido bloqueada.
  - **observables**: un evento observables es aquel que la prueba se limita a observar en su ejecución por parte de la IUT. Cuando ésta ejecuta uno de tales eventos la prueba se limita a evolucionar acordemente, sincronizando adecuadamente e intercambiando datos si fuera menester.
- Cada traza de la prueba que termine deberá identificar un único estado sobre la especificación, con el propósito de asegurar que la asignación de cobertura se hace sobre partes ejercitadas de la especificación.

5.7 Aspectos Operacionales y FDTs

En una prueba se ha de marcar el punto donde termina el preámbulo y comienza el cuerpo. Además, hay que tener en cuenta que la prueba puede no tener preámbulo. Por ello la anotación `END_OF_PREAMBLE` indicará que el preámbulo ha terminado.

El último evento de la prueba debe adjuntar la etiqueta de terminación. Se proponen las anotaciones `PASA`, `FALLA` e `INCONCLUSO` para los tres veredictos asignables, con las restricciones detalladas en la sección 5.6.

Hemos visto que los eventos serán imponibles u observables. Los lenguajes TTCN, SDL y ESTELLE indican la direccionalidad de los eventos externos. No así LOTOS, que requiere haber caracterizado el interfaz con anotaciones, mediante las anotaciones sobre la interfaz `IN` (imponible) y `OUT` (observable).

Por último, se consideran los bloqueos. Evidentemente, los debidos a reacciones erróneas de la IUT se detectan por la propia semántica de pruebas: al no ser una acción incluida en la prueba se determinará que es un bloqueo. Los debidos a falta de reacción por la la IUT se resuelven con un temporizador. En SDL, ESTELLE y TTCN se dispone de primitivas de temporización. En LOTOS será introducido en el interfaz mediante la anotación `timeout #`. Tras `#` unidades de tiempo desde la última acción sin actividad, se considerará que se ha producido un bloqueo.

5.8 Asignación de veredictos

La asignación de veredictos [Rob91] descrita en la norma IS-9646 se resumen en la tabla 1, relacionando la terminación de la ejecución de la prueba (filas) con la clasificación de la misma (columnas).

CLASIFICACIÓN

EJECUCION

	MUST ACCEPT	MAY ACCEPT	MUST REJECT	MAY REJECT
Bloqueo Preámbulo	I	I	I	I
Bloqueo Cuerpo	F	F	P	P
Termina	Etiqueta terminal			

I: Inconcluso P: Pasa F: Falla

Tabla 1: Asignación de veredictos

En la primera columna se presentan las terminaciones de la ejecución de una prueba. Si existe un bloqueo en el preámbulo, el veredicto es **INCONCLUSO**. Si ocurre ya en el cuerpo principal de la prueba obtenemos normalmente un **FALLO** para pruebas de aceptación, excepto en el caso de prueba **MUST REJECT**, que es precisamente la intención de la prueba.

Las tres últimas filas se refieren a la terminación de la prueba. Si se llega al final, se asigna el veredicto esperado (anotación de la última acción).

6 Formalización de la Ejecución de Pruebas

Todos los aspectos comentados han de ser combinados en la formalización de la ejecución de pruebas. Se resalta que existe un tercer elemento en juego que no suele tomar parte en técnicas más convencionales: una especificación de referencia formalizada y manejable por herramientas.

Adoptaremos las siguientes hipótesis de partida:

- La IUT sólo ofrece/ejecuta un evento cada vez.
- La prueba es correcta (según se definió en 5.6).
- Supondremos la existencia de la función io, que determina si una acción de una prueba es imponible o no.

Definiciones Sea T una prueba correcta, I una implementación, t el tiempo transcurrido desde la ejecución de una acción. Sean t_0 e i_0 los estados iniciales de T e I, respectivamente. Denotaremos con TS, TS', TS_i a conjuntos de estados sobre la prueba, y con O_NIL el evento nulo.

Definición 1 *Definimos como resultado de la ejecución de una prueba sobre una IUT como un valor en el siguiente dominio:*
$\mathcal{R}$ = (*BloqueoEnPreambulo, BloqueoEnCuerpo, PASA, FALLA, INCONCLUSO*).

Definición 2 *Definimos el estado de una ejecución de una prueba sobre una IUT como la tupla* $\Upsilon =< TS, i >$

donde TS es un conjunto de estados pertenecientes a la prueba e i es un estado de la IUT. Denotaremos como $\varphi, \varphi_1, \varphi', \ldots$ a elementos de Υ.

Definición 3 *Definimos la variable* $preamble \in \{TRUE, FALSE\}$.

Esta variable tomará el valor $TRUE$ si la prueba se encuentra en el preámbulo y $FALSE$ en el caso contrario. Operacionalmente, comienza siempre con el valor $TRUE$ y lo cambia cuando se ejecute la acción anotada como `END_OF_PREAMBLE`.

Definición 4 *Definimos la función de terminación* $T : offert \rightarrow \{$*PASA, FALLA, INCONCLUSO*$\}$.

que devuelve la anotación de un un evento terminante de una prueba, según se definió en 5.7.

Definición 5 *Definimos la función* $\mathcal{L} : offert \rightarrow \{$*IN, OUT*$\}$.

que devuelve la direccionalidad de un evento.

Definición 6 *Definimos la función* Γ *que dado un evento devuelve el valor asignado en la anotación* `timeout`*:* $\Gamma : offert \rightarrow time$.

Definición 7 *Definimos la función* Ofs *de extracción de eventos dado un conjunto de estados como:* $Ofs : state^* \rightarrow off_set$ *con* $OFs(TS) = \{o | o = Of(t_i)\ \forall t_i \in TS\}$.

Definición 8 *Definimos la función* Ofc *de elección de evento entre una prueba y una IUT como:* $Ofc : \Upsilon \rightarrow offert$. *siendo:*

$$Ofc(< TS_n, i_n >) = \begin{cases} O_NIL & \text{si } t > t_i, \forall t_i = \Gamma(o_i)\ \forall o_i \in Ofs(TS_n) \\ Ofs(TS_n) \otimes Of(i_n) & \text{si } \mathcal{L}(o_i) = OUT \wedge t\ leq\ t_i, \\ & \quad \forall o_i \in Ofs(TS_n)\ \forall t_i = \Gamma(o_i) \\ Ofs(TS_n) \otimes Of(i_n) & \text{si } \mathcal{L}(o_i) = IN \wedge t\ leq\ t_i, \\ & \quad \forall o_i \in Ofs(TS_n)\ \forall t_i = \Gamma(o_i) \end{cases}$$

En el primer caso, tenemos que si el tiempo transcurrido tras la la última acción ejecutada sobrepasa todos los `timeouts` especificados se considera que existe un bloqueo. El segundo caso considera el caso de, sin existir vencimientos de plazo (al menos en alguna acción observable por la prueba), la IUT impone alguna acción. En definitiva se espera la reacción de la IUT, ahora bien, esa reacción debe ser acorde al propósito de la prueba.

Por último, la prueba intenta imponer una acción a la IUT. Recordemos que en este caso TS_n debe ofrecer una sola acción. Si esta acción no es ejecutada por la IUT, el resultado de $Ofs(TS_n) \otimes Of(i_n)$ será, evidentemente, O_NIL.

Definición 9 *Definimos la función* $A : state^* \times offert \rightarrow state^*$
$A(TS, a) = \{s_{i+1} | s_i = a \Rightarrow s_{i+1}\ \forall s_i \in TS\}$

como la función que, dado un conjunto de estados de la prueba y un evento ejecutado, determina el conjunto de estados resultantes en la prueba. Esta función, en definitiva, hace un seguimiento sobre la especificación de los estados de la prueba por los que está pasando la ejecución de la misma.

Definición 10 *Definimos la función $\mathcal{P}$ de paso de ejecución de una prueba sobre una IUT como* $\mathcal{P} : \Upsilon \times offert \rightarrow \Upsilon$.

$$\mathcal{P}(< T, i >, o) = \{(< T', i' >) |\ T' = A(T, o),\ i' = gl(i, o)\}$$

Definición 11 *Definimos la función de ejecución de una prueba sobre una IUT como* $\xi : test, IUT \rightarrow \mathcal{R}$*, donde el comienzo de la ejecución se denota como* $\xi(\{t_0\}, i_0)$.

$$\xi(< TS_n, i_n >) = \begin{cases} T(o) & \text{si } Ofs(TS_n, o) = \emptyset \wedge o \neq O_NIL \\ \textit{BloqueoEnPreambulo} & \text{si } Ofc(< TS_n, i_n >) = O_NIL \\ & \wedge preamble = TRUE \\ \textit{BloqueoEnCuerpo} & \text{si } Ofc(< TS_n, i_n >) = O_NIL \\ & \wedge preamble = FALSE \\ \xi(P(< TS_n, i_n >, Ofc(TS_n, i_n))) & \text{en otro caso} \end{cases}$$

Esta función va evolucionando la prueba acorde con la IUT, decidiendo entre imponer o esperar un evento. Si se detecta un bloqueo, la función determina si ha sido en el preámbulo o en el cuerpo. Una vez terminadas las acciones de la prueba, el resultado es la anotación del evento terminante de la misma.

7 Impacto en parámetros de calidad

En esta sección describen los principales parámetros de calidad y se analiza el impacto de este proceso de formalización:

- Corrección: *capacidad de un producto de satisfacer todos los parámetros de usuario*. Todo el proceso de conformidad persigue, precisamente, la *corrección* como objetivo primordial.
- Fiabilidad: *capacidad de un producto de llevar a cabo las funciones para las cuales fue diseñado*. Este proceso queda soportado por herramientas automáticas que llevan a cabo la ejecución de pruebas. La reducción del factor humano disminuye la posibilidad de error en la interpretación de resultados de la ejecución de pruebas.
- Mantenimiento: *capacidad de un producto para detectar, localizar y corregir errores residuales*. El mantenimiento es facilitado por una Generación Automática de Pruebas y su posterior clasificación según objetivos.
- Testabilidad: *capacidad de un producto para ser probado y verificado*. La formalización de la ejecución de pruebas, junto con la norma ISO 9646 suponen un gran incremento en las posibilidades de pruebas de un sistema, tanto por la definición exacta de interfaces como de métodos.
- Eficiencia: *capacidad de un producto para usar la cantidad mínima de recursos*. La generalidad de métodos y herramientas puede suponer una disminución en la eficiencia del proceso de pruebas, tanto en los aspectos operativos como de preparación de expertos.
- Flexibilidad: *capacidad de un producto para evolucionar*. La existencia de métodos automáticos supone un incremento en la flexibilidad, por la posibilidad de regeneración automática de pruebas y su ejecución.

- Reusabilidad: *capacidad de un producto para ser usado en varoas aplicaciones con los mismos requisitos funcionales*. Puesto que estos han sido capturados formalmente, sólo la fase de implementación necesitará ser revisada, evitando un rediseño del sistema. Asimismo, la disponibilidad de *Pruebas Abstractas* implica una aplicación directa del sistema ante nuevas implantaciones.

- Interoperabilidad: *capacidad de un producto para ser usado conjuntamente con otro(s)*. Puesto que es posible demostrar la conformidad del producto respecto de su norma, se garantiza la satsfacción de definiciones de interfaces [Dil91].

8 Conclusiones

En este artículo se aborda la formalización de la fase de ejecución de pruebas de conformidad en un entorno en el que se dispone de una especificación formal del sistema, que consideramos como *referencia*. Se abordan desde aspectos metodológicos (*prueba correcta*) hasta operacionales (ejecución de pruebas), embebido en un entorno normalizado (ISO-9646).

Se define la *prueba correcta*, de crucial importancia a la hora de ejecutar, interpretar resultados y asignar veredictos. En la definición se tienen en cuenta aspectos como el indeterminismo en la especificación de referencia, imponibilidad y observabilidad, y otros que afecta al proceso de ejecución.

Se elabora un modelo *operacional* de ejecución de prueba genérico, no dedicado a ningún producto en particular y orientado a las pruebas de Conformidad.

ISO ha desarrollado una serie de normas (9XXX) con el objetivo de cuantificar la calidad de un producto. Para ello define una serie de parámetros que tienen influencia en la calidad, para posteriormente cuantificarlos en base a ciertas métricas. En esta sección se relata la influencia que tiene la introducción de FDTs en el ciclo de vida, particularizando para la fase de pruebas.

En primer lugar, el simple hecho de utilizar FDTs elimina los problemas de ambigüedades, faltas de precisión que puedan existir en los requisitos de usuario tal como el cliente los describe.

El hecho de que existan Baterías de Pruebas para homologación de productos supone una mejora inmediata en cuanto a *reusabilidad* y *flexibilidad* para determinar la conformidad de un producto. Por otro lado, como se demuestra en el presente artículo, es posible formalizar el proceso de ejecución de pruebas lo cual, junto al hecho de la existencia de los PICS y PIXIT, arroja un resultado positivo en cuanto al incremento de la *testabilidad*[9] de un producto.

La propia automatización del proceso, junto a la existencia de métodos automáticos de Generación de Baterías de Pruebas supone una mejora importante en dos aspectos básicos de la calidad del producto: su *corrección*, que queda asegurada, y su *fiabilidad*. Es importante resaltar de nuevo, para no llamarse a engaño, que las FDTs se han pensado con el objetivo de capturar requisitos funcionales. Esto deja de lado aspectos de *robustez*.

Por último, el *mantenimiento* de grandes Baterías de Pruebas es mejorado por el hecho de disponer de herramientas software que las generan automáticamente.

Como aspecto negativo, cabe destacar que las herramientas de manejo de FDTs hoy en día consumen bastantes recursos de memoria y tiempo, con la consiguiente pérdida de *prestaciones*. Afortunadamente, el rápido avance en ordenadores hace que esta faceta se diluya en importancia con la aparición de sistemas más capaces.

[9] Testability

Referencias

[Dil91] F. Dilonardo. A Comparison Between Conformance, Interoperability Performance, Development Testing. In CEN/CENELEC & ETSI, editor, *Conformance Testing and Certification in Information Technology and Telecommunications*, pages 377–384. IOS Press, 1991. Proc. of the European Conf., Brussels, 13–15 June, 1990.

[dNH84] Rocco de Nicola and M. Hennessy. Testing Equivalences for Processes. *Theoretical Computer Science*, 34:83–133, 1984.

[ISO91] ISO. Information Processing Systems - Open Systems Interconnection - Conformance Testing Methodology and Framework. IS 9646, ISO, 1991.

[MdM89] José A. Mañas and Tomás de Miguel. From LOTOS to C. In Ken J. Turner, editor, *Formal Description Techniques, I*, pages 79–84, Stirling, Scotland, UK, 1989. IFIP, North-Holland. Proceedings FORTE'88, 6–9 September, 1988.

[Rob91] Tomás Robles. *Contribución al Tratamiento Formal de la Fase de Pruebas del Ciclo Software en Ingeniería de Protocolos*. PhD thesis, Dpt. Ingeniería Telemática, ETSITM, Tech. Univ., Madrid (ES), 1991. In Spanish.