Modelado Formal de los PCOS para Entornos de Pruebas[1]

Gabriel Huecas
gabriel@uax.es

Dept. Informática
Universidad Alfonso X, el Sabio
Av. de la Universidad, Villanueva de la Cañada, E-28691 MADRID

José A. Mañas, Tomás Robles
jmanas@dit.upm.es

Dept. Ingeniería de Sistemas Telemáticos
E.T.S.I. Telecomunicación
Ciudad Universitaria, E-28040 MADRID

Resumen

En este artículo se presenta un estudio del modelado de puntos de control y observación para la integración de técnicas de descripción formal con la ejecución de baterías de pruebas sobre productos reales.

1 - Introducción

En el campo de la Ingeniería de Pruebas, los organismos internacionales de normalización (ISO e ITU) están tomando un papel fundamental mediante el establecimiento de Normas que enmarcan el entorno de trabajo y el universo de discurso desde la concepción y generación de pruebas hasta su ejecución sobre productos finales.

Como resultado de esta actividad ha surgido la norma ISO-9646 ([ISO91]) que define un entorno específico y una metodología para el desarrollo y ejecución de Pruebas de Conformidad. Además, se está trabajando en una norma ([N6294]) que formalice los conceptos expuestos en el 9646. Las Pruebas de Conformidad son una rama de la Ingeniería de Pruebas donde un producto es probado respecto de su especificación.

Por otro lado, la creciente complejidad de los protocolos y sistemas han provocado la necesidad de especificaciones bien definidas, completas, precisas y no ambiguas, por lo que han surgido Técnicas Formales de Descripción (FDTs) de sistemas. Entre ellas, LOTOS [ISO89] destaca por su fuerte base matemática, existiendo entornos de desarrollo y metodologías de diseño, generación automática de pruebas, etc.

[1] Este trabajo ha sido parcialmente financiado por el proyecto E-MEDAS (Programa Nacional de Investigación en Tecnologías de Información y Comunicación) y por el proyecto ATAC (LOTOS Validation of an Ada Runtime Environment, ESA 10797/94/NL/FM(SC)).

1.1 - Arquitecturas de Pruebas

En la ejecución de pruebas sobre la IUT (Implementation Under Test), dos aspectos destacan pos su influencia en los resultados a obtener y en la operatividad de la ejecución: el Método (o Arquitectura) de Prueba y los Puntos de Control y Observación (PCOs).

Método de prueba El Método de Pruebas define un modelo para la accesibilidad de la IUT en términos de PCOs y su lugar dentro del modelo de Referencia OSI [ISO84].

Poco se puede hacer en cuanto a la elección del Método Abstracto de Prueba. Normalmente, nos limitamos a elegir uno de ellos simplemente por la disponibilidad de los PCOs o conveniencias técnicas de la ejecución de pruebas.

Puntos de Control y Observación El otro aspecto que tiene un papel fundamental en la ejecución son los PCOs. Si la ejecución de pruebas tuviese acceso directo a los IAPs, coincidirían plenamente con los PCOs, caso que se considera ideal[2] porque normalmente la Especificación del Protocolo hace referencia a los IAPs (nunca a los PCOs), por lo que las Pruebas derivadas para el sistema se referirá a los IAPs. Por otro lado, veremos que la no coincidencia entre PCOs e IAPs modificará el comportamiento de las interacciones, alejando la semántica de las sincronizaciones esperada por las pruebas de la que realmente nos presenta nuestro SUT (System Under Test).

2 - Objetivo

La existencia de especificaciones formales de protocolos y sistemas de comunicaciones ha provocado el desarrollo de métodos automáticos para generar pruebas. Estas pruebas típicamente se describen en el mismo formalismo que la propia especificación. El inconveniente surge a la hora de ejecutar dichas pruebas sobre un producto real, cuyo interfaz dista mucho del modelo de sincronización de la FDT utilizada, en particular, con el interés centrado en LOTOS.

El objetivo principal de este artículo es establecer una caracterización formal de los PCOs, con una doble meta: por un lado, desarrollar una base para su estudio y comparación; por otro, mejorar la operativa de la ejecución de pruebas a partir del modelado formal de los PCOs. Estos dos objetivos redundan en una mejora de la capacidad de pruebas de los sistemas en los que los PCOs se determinan desde tempranas etapas en la especificación y diseño.

El enfoque que se adoptará para formalizar los PCOs es estudiar todas las propiedades relevantes que modifican el comportamiento del PCO, diferenciando entre las propiedades abstractas o de alto nivel de los PCOs (semántica) de las propiedades físicas (naturaleza).

Con la identificación y definición de estas propiedades que modelan un PCO podremos establecer una clasificación de los mismos. Por lo tanto, su comparación se podrá establecer si podemos formalizar y expresar en el mismo formalismo dichas propiedades.

Mediante una combinación adecuada de estas propiedades, es posible establecer modelos para los diferentes PCOs que intervengan un un Sistema de Pruebas, ya sean definidos en un

[2] En la Norma 9646 se considera el Método de Referencia.

lenguaje de programación ya sean interfaces de comunicaciones convencinales, como líneas de comunicaciones, RPCs, etc.

Además, necesitamos un modelo operativo de ejecución de pruebas que aproveche la semántica de la especificación de las pruebas

3 - Modelado y Caracterización de los Puntos de Control y Observación

Según la norma ISO-9646, un PCO es "Un punto dentro del entorno de pruebas donde la ocurrencia de eventos de prueba será controlada y observada, según se define en un método abstracto de prueba". En la misma norma, se dice que un PCO es caracterizado por el conjunto de Primitivas de Servicio Abstractas (ASPs) y/o las Unidades de Datos del Protocolo (PDUs) que pueden intercambiarse en dicho PCO.

De la primera definición es claro que un PCO modela conceptualmente los límites entre un sistema y su entorno. De la segunda, se deduce que un PCO es un punto de interacción externo del sistema [Rei93]. También puede ser entendido como los mecanismos o medios por los cuales el entorno afecta a un sistema y viceversa. En definitiva, un PCO es un concepto arquitectural en el campo de los Sistemas Abiertos.

3.1 - Características de un PCO

3.1.1 - Propiedades de Alto Nivel

Estas propiedades están relacionadas con el comportamiento observable de los PCOs, en cuanto a las características y naturaleza de las interacciones que tiene lugar en ellos. Ya en [Rei93] se ha hecho un amplio estudio de la naturaleza de las interacciones desde el punto de vista de comunicaciones entre entidades participantes. En nuestro trabajo, haremos hincapié en cómo la naturaleza de los PCOs influyen en dichas interacciones, centrando nuestra atención en el concepto arquitectural de Punto de Interacción.

- **Fiabilidad**: Se dice que un PCO es Fiable si no crea, duplica, corrompe o pierde interacciones.

  Aunque parece que este concepto es básico desde el punto de vista de ejecución de pruebas, veremos que existen Implementaciones cuyos IAPs no serán fiables. Esto conducirá a que, independientemente del Método de Prueba, los PCOs resultantes no serán fiables, con el consiguiente efecto negativo sobre los veredictos resultantes. En el caso peor, puede ocurrir que la IUT no sea susceptible de ser conformada respecto de la Norma.

- **Modo de Interacción**:

  Dependiendo de la temporización en que el Probador y el SUT atienden a las interacciones que se intercambian, distinguiremos entre:

  - **Síncrono**: la oferta y aceptación de la interacción son simultáneas, es decir, no transcurre el tiempo entre ellas.

  - **Asíncrono**: la oferta y aceptación no son simultáneas, es decir, tendrán lugar en momentos diferentes y, como es lógico, la aceptación será posterior a la oferta o entrega.
  - **Mixto**: algunas interacciones son síncronas y otras son asíncronas. Se incluye este modo más por completitud en la definición que por encontrarlo en modelos reales de PCOs.

- **Estrategia**: cuando existe una serie de interacciones listas para ser entregadas, existen diferentes estrategias de entrega:
  - **Exclusividad**: cuando una interacción ocurre, las demás son inhibidas, en el sentido de que ya nunca ocurrirán (a menos, claro está, que vuelvan a ser ofertadas).
  - **Equidad**: todas las interacciones tendrán lugar alguna vez.
  - **Temporización**: las interacciones se acompañan de unos plazos de entrega o ejecución, transcurrido el cual, desaparecerán.
  - **Prioridad**: las interacciones pueden tener diferente prioridad de ejecución.

- **Cardinalidad**: dependiendo del número de participantes en una interacción[3], distinguiremos entre:
  - **Binaria**: 2 participantes.
  - **n-aria**: más de dos participantes.

- **Intercambio de Valores**: los valores intercambiados en la interacción pueden ser impuestos por un participante o negociados entre todos. Por lo tanto, diferenciaremos entre:
  - **Paso de Valores**: un participante impone un(os) valor(es) al resto.
  - **Negociación de Valores**: los valores son decididos entre todos los participantes.

- **Tipado**: el conjunto de posibles tipos de interacción puede estar restringido.

- **Rango de valores**: los datos intercambiados en una interacción pueden tener restringido su valor en un cierto rango.

- **Selección**: la elección de la siguiente interacción puede depender de restricciones a su tipo o a sus parámetros.

Existen una serie de relaciones entre estas propiedades, de forma que algunas imponen una modificación o incluso una restricción total a otras. Por ejemplo, es claro que una interacción asíncrona no puede negociar valores. Asimismo, la negociación de valores impone una sincronía en la interacción.

Como vemos, estas propiedades no hacen referencia directa a la naturaleza física del PCO. Sin embargo, existe una estrecha relación, puesto que va a depender de las capacidades elementales del PCO el que estas propiedades existan o no. Veamos primero cuáles son las características elementales del PCO para así establecer la relación entre unas y otras.

[3] En Pruebas de Conformidad, solamente participarán dos entidades, la IUT y el Probador

3.1.2 - Propiedades Elementales: Definición y Formalización

La caracterización puede basarse en la naturaleza más física de los PCOs, es decir, en su naturaleza y construcción, en cuyo caso las propiedades relevantes serán Capacidad de Encolamiento, Simetría, Rechazo, etc. Veremos una definición intuitiva y su expresión formal usando LOTOS como lenguaje descriptivo.

El entorno en el que nos movemos en la formalización se representa en la figura 1:

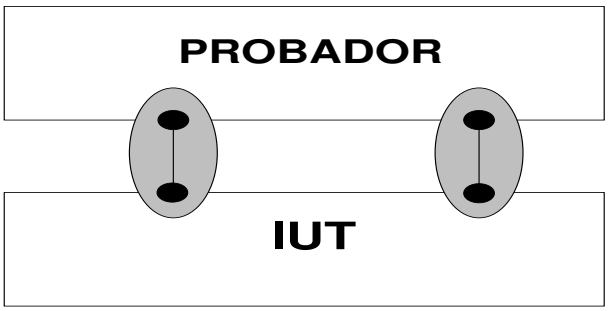

Figura 1: Los PCOs como interfaz entre Probador e IUT

- **Sincronicidad**: se dice que un PCO es Síncrono si todos los participantes se ven envueltos en el mismo momento en las interacciones que tienen lugar en él, entendiendo que ninguno podrá llevar a cabo ninguna actividad hasta que la interacción presente este terminada.

  El comportamiento formal de un PCO síncrono y asíncrono se presenta en la figura 2.

process PCOsynch [interchange]:**noexit** :=	**process** PCOasynch [send, recv]:**noexit** :=
interchange; PCOsynch [interchange]	send; recv; PCOasynch [send, recv]
endproc	**endproc**

Figura 2: PCO síncrono y asíncrono

- **Simetría**: se dice que un PCO es Simétrico si todos los participantes conectados a él lo ven igual, es decir, no existe la posibilidad de hacer una distinción entre ellos. Entre otras implicaciones, esto significa que todos los participantes pueden, en principio, comenzar o aceptar una interacción.

  El concepto de simetría se expresa en diversas facetas de un comportamiento LOTOS. Por un lado, existe cierta asimetría cuando uno de los participantes ofrece un valor y otro lo acepta. Por otro lado, también existe asimetría cuando los participantes ven diferente al PCO. Por lo tanto, tendremos en la figura 3 dos formas de ver la simetría (diferentes aspectos):

- **Capacidad de Almacenamiento**: se dice que un PCO tiene Capacidad de Almacenamiento si puede mantener varias interacciones sin perderlas. De esta forma, en general

```
process OneWay [recv. send] : noexit :=
    recv ? Inter : ANY; send ! Inter; OneWay [recv, send]
endproc

process PcoAsimetrico [UpperIn, LowerOut] : noexit :=
    OneWay [UpperIn, LowerOut]
endproc

process PcoSimetrico [UpperIn, LowerOut, Lowerin, UpperOut] : noexit :=
        OneWay [UpperIn, LowerOut]
    |||
        OneWay [LowerIn, UpperOut]
endproc
```

Figura 3: PCO simétrico y asimétrico

```
process PCO [recv, send] (InterPendientes : SET_of_Inter) : noexit :=
    recv ? Inter: ANY;
        PCO [recv, send] (Add (Inter, InterPendientes))
        []
        [IsNotEmpty (InterPendientes)] ->
            send ! Saca (InterPendientes);
            PCO [send, recv] (Saca (InterPendientes))
endproc
```

Figura 4: PCO con Capacidad de Almacenamiento

un participante puede continuar con otras acciones. No hay espera ni bloque de los participantes atendiendo a una interacción. En la figura 4 se especifica este comportamiento.

- **Rechazo**: se dice que un PCO tiene Capacidad de Rechazo si puede impedir a un participante ofertar una interacción.

- **Pérdidas**: Se dice que un PCO puede causar la Pérdida de una interacción si ésta es aceptada en su oferta pero no hay reacción subsiguiente, es decir, no se entrega a ninguno de los destinatarios. En general, se entiende que el ofertante no recibe información referente a esta pérdida.

  El comportamiento de este PCO se especifica en la figura 5.

La principal diferencia entre Ignorar y Perder una interacción es que la primera está relacionada con los participantes que reciben la interacción, mientras que la segunda es responsabilidad del PCO.

```
process PCO [recv, send] : noexit :=
        recv ? Inter: ANY;
        send ! Inter;
        PCO [recv, send]
    []
        recv ? Inter: ANY;
        PCO [recv, send]
endproc
```

Figura 5: PCO con Pérdidas

3.2 - Clasificación de Modelos de PCOs Reales

En esta sección vamos a estudiar, mediante la formalización propuesta, diferentes modelos de PCOs que se dan en la realidad. La idea básica es poder comparar dichos modelos y establecer cuáles son las Propiedades Elementales de los mismos, de forma que se puedan comparar y por lo tanto, prever cuáles serán los principales puntos que hagan disminuir la capacidad del SUT de ser probado.

A la hora de escoger los PCOs a estudiar, nos centraremos en aquellos que entendemos de más uso en el campo de la Ingeniería Software, específicamente en Pruebas de Conformidad y Validación de Protocolos, puesto que este es el objetivo principal de la Norma ISO9646.

Estos modelos serán los representados por las sincronizaciones en lenguajes de Especificación y lenguajes de programación, ambos usados para describir y realizar Protocolos. Dentro de los lenguajes de Especificación tendremos las FDTs LOTOS, Estelle y SDL. En los lenguajes de programación hemos elegido el Ada como más representativo, de amplio uso en Ingeniería Software Aeroespacial y en Ingeniería de Protocolos y porque incluye la comunicación entre tareas como elemento propio del lenguaje.

Como punto de referencia, también se han incluido dos modelos de componentes físicos que nos pueden aparecer en Sistemas Bajo Pruebas: una linea de comunicaciones con Protocol Ethernet y una linea de conexión Hardware entre circuitos integrados.

La información se ha organizado en forma de tabla (figura 6 donde se explicitan las Propiedades Elementales que nos presentan estos PCOs[4].

4 - Especificación de PCOs en LOTOS

4.1 - Anotaciones

Hemos visto que LOTOS puede ser usado para caracterizar formalmente las distintas propiedades que presentan los PCOs. Sin embargo, es una aproximación nada práctica el usar dichas especificaciones en sistemas que requieran PCOs con memoria, por ejemplo. En definitiva, la semántica de LOTOS impone un modelo de sincronización que no es, con mucho, el más general presentado por los sistemas de comunicaciones reales. En este sentido es

[4]Las RPCs corresponden al modelo UNIX, ver [Blo92]

	Sincronía	Simetría	Cola	Rechazo	Ignora	Pérdidas	Temporización	Prioridad
LOTOS	Si	Si	No	Si	No	No	No	No
SDL	No	No	Si	No	Si	No	Si	Si
Estelle	No	No	Si	No	Si	No	Si	No
TTCN	No	No	Si	No	No	No	Si	No
Hardware	Si	No	No	No	Si	No	Si	Si
Ada	Si	No	Si	No	No	No	Si	Si
Ethernet	No	No	Si	No	Si	Si	No	No
RPC UNIX	Si	No	x	Si	No	Si	Si	No

x: puede implementarse con o sin cola

Figura 6: Taxonomía de algunos modelos de PCOs

deseable enriquecer el lenguaje con construcciones que determinen el tipo de sincronización adecuada a nuestro sistema.

En definitiva se busca un lenguaje que pueda modelar cualquier tipo de sincronización de forma directa y sin construcciones indirectas. No obstante, no se busca una redefinición del lenguaje, por varios motivos: Primero, la definición del lenguaje debe ser normalizada, sin derivar dialectos distintos. Segundo, el lenguaje se está redefiniendo actualmente por ISO. Tercero, existe ya un mecanismo utilizado para ampliaciones no normalizadas del lenguaje llamado anotaciones [MdMSA93]. Este mecanismo introduce primitivas en el lenguaje en forma de comentarios especiales, de forma que son transparentes a las herramientas LOTOS, excepto aquellas que se ven afectadas por la modificación introducida (usualmente, generadores de código).

Actualmente las anotaciones se están utilizando para:

- entrada/salida de datos del exterior, introducción de temporización y prioridades [Sal93].
- modificación y adaptación del código generado para los tipos de datos cuando la eficiencia (tiempo y/o memoria) de los compiladores no es adecuada [Vei93].
- introducción de cálculos estadísticos y tratamiento probabilístico [MN91].

La sintaxis de las anotaciones es muy sencilla: se introducen como comentarios a la especificación, de forma que las herramientas normalmente no las consideran. Sólo aquellas que se ven afectadas las reconocen y las interpretan. Para diferenciarlas de los comentarios normales, se añade el carácter "|" al empezar y acabar el comentario. Además, se adjunta una palabra clave que las identifica, mas el texto necesario si la anotación lo requiere. Como ejemplos:
`(*| C printf ("%s\n", kd_draw ($1) |*),(*| delay 10 |*),...`

4.2 - Anotaciones para Modelización de PCOs

Para adecuar la semántica de las sincronizaciones de LOTOS a los distintos modelos de PCOs se proponen las anotaciones descritas más abajo, teniendo siempre como modelo por defecto la sincronización LOTOS. Por lo tanto, las anotaciones están orientadas a aquellas características que el lenguaje LOTOS no contempla. Estas anotaciones solo deberán afectar

a las puertas externas de la especificación, puesto que son éstas los puntos de sincronización con el entorno.

Se proponen las siguientes anotaciones:

- `priority`

  En LOTOS, cuando se ofrecen varias ofertas, se elige una de forma equitativa (es decir, no hay distinción por prioridades). Esta anotación caracteriza una puerta externa con un valor entero de prioridad. Las acciones que no lleven esta anotación se les supone una prioridad 0. A la hora de elegir que acción se ofertará, se elige aquella de mayor prioridad. Entre ofertas de la misma prioridad, se elige con la semántica habitual de LOTOS.

- `asynchronous`

  Este anotación indica que la oferta es asíncrona, es decir, no se espera que el entorno reaccione, por lo que el sistema puede seguir evolucionando. Otra implicación es que no puede existir negociación de valores.

- `in, out`

  Esta anotación afecta al modo en que se tratan los valores ofertados en una oferta. Así, `in` indica que la interacción es de entrada y `out` de salida. LOTOS, por defecto, no impone ninguna restricción al sentido de los datos, tomándose entonces como de entrada o salida indistintamente. Esta anotación es obligatoria cuando la oferta en que aparecen se ve afectada por una anotación `asyn\-ch\-ro\-nous`.

- `timeout`

  Siendo value un valor de tipo real, que dependerá de la plataforma de ejecución. Esta anotación indica que pasados value unidades de tiempo, las acciones sobre la puerta dejarán de ser ofertadas.

- `queue`

  Esta anotación indica que las sincronizaciones sobre la puerta afectada serán encoladas y quedarán a la espera de ser consumidas. Esta anotación podrá estar acompañada de una serie de atributos que caracterizarán el tratamiento ante casos de colas llenas o vacías, según la siguiente lista:

  - `queue`: cola ilimitada en tamaño.
  - `queue #`: donde # es una valor entero. En este caso la cola tiene una capacidad máxima de # posiciones.
  - `queue full => lost`: Este atributo indica que si la cola está llena, las siguientes interacciones se perderán.
  - `queue full => wait`: Este atributo indica que si la cola está llena, el enviante queda a la espera, sin posibilidad de evolucionar.
  - `empty => default value`: Si la cola está vacía, la interacción por defecto es `value`.

- `empty => wait`: Si la cola está vacía, el receptor queda en espera de la interacción.
- `empty => nosynch`: Si la cola está vacía, la interacción no tiene lugar.

Los atributos caracterizados como `full` aplican sobre interacciones de salida (`out`), mientras que los caracterizados como `empty` aplican a interacciones de entrada (`in`).

5 - Conclusiones

En este artículo se modela y formaliza unos de los puntos claves en los Sistemas de Pruebas: los Puntos de Control y Observación. Para ello se enumeran las propiedades los definen, y se expresan en un formalismo que permita su comparación y taxonomía.

Además se presenta un mecanismo para incluir los aspectos no funcionales de los PCOs en un lenguaje de especificación formal como LOTOS.

Con ello se consigue completar un escenario de ingeniería donde se aúnan sistemas y baterías de pruebas descritas formalmente, ejecutables sobre productos reales. Todo ello sobre un ejecutor de pruebas que será aplicable a cualquier sistema. Las diferencias entre los interfaces serán absorbidas mediante un colchón software guiado por la taxonomía propuesta.

Referencias

[Blo92] John Bloomer. *Power programming with RPC*. O'Reilly & Associates, Inc., 1992.

[ISO84] ISO. Information technology - open systems interconnection - basic reference model. IS 7498, ISO, 1984.

[ISO89] ISO. Information processing systems - Open systems Interconnections - LOTOS a Formal Description Technique based on the Temporal Ordering of Observational Behaviour. IS 8807, ISO, 1989.

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

[MdMSA93] José A. Mañas, Tomás de Miguel, Joaquín Salvachua, and Arturo Azcorra. Tool Support to Implement LOTOS Formal Specifications. *Computer Networks and ISDN Systems*, 25(7):815–839, February 1993.

[MN91] Carlos Miguel Nieto. *Técnicas de Descripción Formal Aplicadas a la Evaluación de Prestaciones de Sistemas de Telecomunicación*. PhD thesis, Universidad Politécnica de Madrid, Madrid, 1991.

[N6294] ISO/IEC JTC1/SC21 N6201. Information retrieval, transfer and management for osi, formal methods in conformance testing, working draft. Technical report, Southampton Output, ISO, February 1994.

[Rei93] Gotzhein Reinhard. *Open Distributed Systems: On Concepts, Methods, and Design from a Logical Point of View*. PhD thesis, University of Hamburg, Hamburg, April 1993.

[Sal93] Joaquín Salvachúa. *Arquitectura de Programas a Partir de Especificaciones Formales*. PhD thesis, Universidad Politécnica de Madrid, Madrid, 1993.

[Vei93] Marcelino Veiga. GLAD: General Language to Annotate Data using TOPO. Technical report, Dpt. Telematics Engineering Technical Univ. Madrid, Ciudad Universitaria, E-28040 Madrid, Spain, July 1993. Version 3R2.