Diseñando la Retroalimentación Visual en un Fomalismo de Concepción de Aplicaciones Interactivas

Jaime Muñoz Arteaga
Instituto Nacional de Astrofísica Óptica y Electrónica (INAOE)
Coordinación de ciencias computacionales
Luis Enrique Erro No. 1, Sta. Maria Tonantzintla
Puebla México 72840
jaime@inaoep.mx

Philippe Palanque
Laboratory on Interaction between Human and Systems(LIHS)
Université Toulouse , 1, Place Anatole France,
31042, Toulouse Cedex, France,
palanque@cict.fr

José Martín Abeleira
TA-TA S.A.
José de Béjar 2600, C.P.12.100
Montevideo - Uruguay
mabeleira@tata.com.uy

Abstract

In an interactive application, the term of visual feedback (rendering or application's visual response) is applied to any graphical form of communication directed from the application towards the user. In spite of visual feedback is predominant in current interactive applications and it is a usability factor that guide user task, the designer doesn't have any formal method to specify and to evaluate it. The present paper deals with the formal specification of visual feedback, and its relationship with the formal specification of the dialogue between interactive application and user. We first present a taxonomy of rendering according to its function in the application. We briefly recall the basics of the ICO formalism, which is used for the formal specification of the application. We then present a case study illustrating how various categories of rendering are taken into account in the ICO formalism. Lastly, we show how mathematical analysis can be performed on the ICO models to verify some properties of an interactive application.

Keywords: Formal methods, Petri nets, Visual feedback, Design of Interactive Applications.

Resumen

En una aplicación interactiva la retroalimentación visual es toda forma de comunicación en forma gráfica que va del sistema en dirección al usuario. A pesar que la retroalimentación visual predomina en los actuales aplicaciones interactivas y en particular es considerada como un factor de usabilidad que auxilia al usuario en su tarea, el diseñador no cuenta con un medio para especificarla y evaluarla explícitamente. Este articulo propone la especificación de la retroalimentación visual y su relación con la especificación formal de dialogo entre el sistema y el usuario. El trabajo primero clasifica la retroalimentación visual en diferentes categorías. Enseguida, se presentan los fundamentos del formalismo OCI el cual es propuesto para especificar la retroalimentación visual. Después, presentamos un caso de estudio ilustrando como diversas categorías de retroalimentación son tomadas en cuenta en el diseño de una aplicación interactiva. Finalmente mostramos como un análisis matemático puede ser desarrollado en el formalismo los OCI para verificar la propiedades de una aplicación interactiva.

Palabras claves: métodos formales, redes de Petri, retroalimentación visual, diseño de aplicaciones interactivas

1 Introduction

La retroalimentación visual (feedback o respuesta visual) es toda forma de comunicación en forma gráfica que va de una Aplicación Interactiva (AI) en dirección al usuario. Esto lo podemos ver con la ayuda del modelo arquitectural de Seeheim [4] (ver ***Figura 1***). En este modelo muestra que por muy simple que sea la acción del usuario el sistema responde con una retroalimentación bajo diferentes formas visuales (por ejemplo ventanas, iconos, interactores, mapas sensibles y mensajes de error) con el fin de guiar y hacer fácil la tarea del usuario. Estas diferentes formas visuales pueden provenir de cualquiera de los componentes de software de una aplicación interactiva.

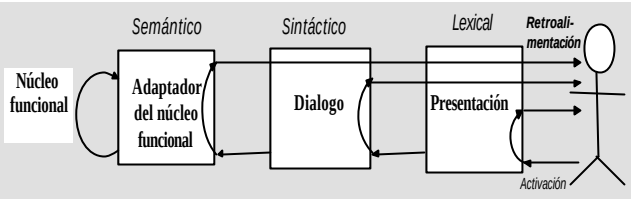


Figura 1. Modelo de Seeheim [4] enfatizando los tres niveles lingüísticos de la retroalimentación visual.

El modelo arquitectural de Seeheim preconiza que el soporte de comunicación entre el sistema y el usuario puede estructurarse como un lenguaje compuesto de tres módulos: el de *Presentación, el de Dialogo y el del Adaptador del núcleo funcional*. El modulo del Adaptador del núcleo funcional gestiona el aspecto semántico a través de una capa de adaptación con la parte non interactivo del sistema (núcleo funcional). El modulo de Presentación gestiona el aspecto lexical de la interacción tanto en el canal de entrada como el de salida de información del sistema. El modulo de Dialogo gestiona el aspecto sintáctico ya que en función del estado del sistema se define la lógica interacción entre los dos módulos anteriores.

Los canales de información de entrada y salida mencionados anteriormente se identifican bajo el nombre de *activación y retroalimentación* respectivamente. Estos dos canales son de naturaleza diferente, puesto que la activación se basa en los eventos, mientras que la retroalimentación se basa en estado de una AI [2].

Es importante mencionar que el paradigma orientado a objetos y los ambientes administradores de interfaces de usuario han permitido resolver en gran parte el problema de la implementación de la retroalimentación. Sin embargo, estas técnicas y herramientas no cubren el diseño explicito y la evaluación de la retroalimentación. Una forma de conseguir dicho objetivo es con el uso de métodos formales en forma de lenguajes, técnicas y herramientas con base matemática que permiten tanto la especificación como la verificación de los diferentes aspectos de una AI.

En la literatura de la Interacción Humano Computadora (IHC) en el área de métodos formales ha habido un gran interés en el estudio de la activación [10], [5] pero se ha descuidado el estudio de la retroalimentación. Según Dix [3], la mayoría de los lenguajes formales para el diseño de sistemas interactivos sufren de los problemas siguientes:

- La especificación de la a retroalimentación visual es ignorada.
- La especificación de la retroalimentación visual es tomada en cuenta de manera incorrecta.
- La especificación de la retroalimentación visual es correcta pero difícil de entender.

Estas tres situaciones suelen ocurrir a pesar que la retroalimentación es considerada como un factor clave para garantizar el fácil uso (o usabilidad) y la utilidad en una aplicación interactiva [8] [11], y muy en particular en la especificación de la retroalimentación visual, la cual predomina en los sistemas actuales. El presente trabajo propone como el formalismo OCI a base de redes de Petri Orientadas a Objetos puede ampliarse para dar soporte a la especificación de la retroalimentación visual, de tal manera que permite razonar sobre el aspecto externo de un sistema eliminado las ambigüedades o inconsistencias en la especificación.

El presenta trabajo primero propone una clasificación de la retroalimentación visual en función de sus tres niveles lingüísticos. Enseguida, una vez presentado los fundamentos del formalismo OCI, el trabajo explica como dicho formalismo especifica los diferentes tipos de retroalimentación visual. Al final presentamos un caso de estudio ilustrando como diversas categorías de retroalimentación son tomadas en cuenta por el formalismo OCI.

2 Taxonomía de retroalimentación visual

El usuario cuando interactúa con una AI espera que la retroalimentación visual le auxilie en tres aspectos fundamentales de la interacción, a saber: *(1) guiarle en sus acciones, (2) informarle sobre la disponibilidad de los servicios de la AI y (3) notificarle el estado actual de la AI.* En otros términos, a cualquier instante de la interaccion, la retroalimentación visual debe tener respuesta a preguntas del usuario tales como: ¿Dónde estoy?, ¿Qué puedo hacer?, ¿Qué esta pasando?, ¿Qué sigue? y ,¿Qué es este objeto? etc.

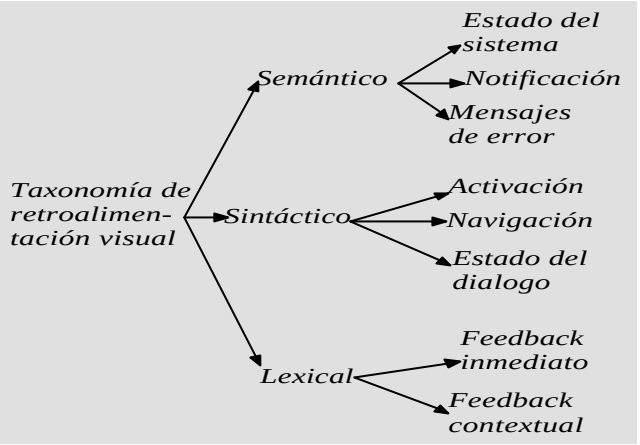


Figura 2.Taxonomía de retroalimentación visual de Aplicaciones Inteeractivas [7].

Del conjunto de respuestas que responden a las preguntas anteriores se deriva una gran diversidad de tipos de retroalimentación visual, esta diversidad ha sido clasificada en la taxonomía propuesta por [7]. La ***Figura 2*** muestra dicha taxonomía la cual permite identificar diferentes niveles de abstracción de la retroalimentación visual conforme a la naturaleza lingüística del dialogo humano-computadora. Los tres niveles lingüísticos de la retroalimentación visual se definen en función del triple rol que esta tiene en la interacción humano computadora, es decir: guiar al usuario en sus acciones e informarle sobre el comportamiento y el estado actual de la AI. Según el tipo de retroalimentación determina el tipo de modulo de software del modelo arquitectural de Seeheim (ver figura 1) del cual proviene la información a visualizar. Así el modulo de Dialogo es la fuente de información para el tipo de feedback de navegación. La taxonomia es independientemente de cualquier ambiente gráfico, cubriendo una gran cantidad de estilos de interacciones desde el línea de comando hasta la manipulación directa.

En los párrafos siguientes se presenta una breve descripción de los elementos de la taxonomía de la Figura 2. Para mayor información sobre esta taxonomía consultar el articulo de [7].

2.1 La retroalimentación visual a un nivel semántico

El nivel semántico reagrupa los tipos de retroalimentación concerniente al acceso y la manipulación de los objetos que pertenecen al núcleo funcional del sistema. Tres tipos de retroalimentación son posibles, a saber:

- *El estado del sistema* en un diseño orientado a objetos, consiste en desplegar ya sea el valor de los atributos públicos de los objetos del núcleo funcional, ya sea el resultado de las llamadas a los métodos de estos objetos o la combinación funcional de estos dos tipos de información.
- *La notificación* responde en general a la pregunta ¿Que esta pasando?. Este tipo de retroalimentación informa sobre el avance de un proceso del núcleo funcional (e.gr. notificar el avance de copiar un documento voluminoso) que interviene en la tarea actual del usuario.
- *Los mensajes de error* tienen como objetivo asistir al usuario en su tarea en caso de dificultades o advertirle de un error ocurrido en el sistema. En todo mensaje de error debe informar sobre la naturaleza del error, la causa probable y las medidas necesarias para corregir el error.

2.2 La retroalimentación visual de nivel sintáctico

Los tipos de retroalimentación visual a nivel sintáctico son aquellos que mantienen informado al usuario sobre la evolución del comportamiento del sistema. En este nivel se sitúan los tipos de retroalimentación siguientes:

- *La activación* define el espacio de interacción, es decir informa a todo instante de la interacción sobre el conjunto de servicios disponibles e indisponibles al usuario.
- *La navegación* de una AI facilita el acceso de la información en unidades lógicas (ventanas, cuadros de dialogo, ventanas secundarias etc..) con el fin de facilitarle la exploración de la información y responder a las preguntas del usuario tales como ¿donde estoy? y a ¿donde puedo ir?
- *El estado de dialogo* ofrece facilidades y servicios para que la tarea del usuario sea mas precisa y confortable. Este tipo de retroalimentación es administrada por los objetos de interacción, pero no está relacionada directamente con la activación (o desactivación) dichos objetos.

2.3 La retroalimentación visual de nivel lexical

La retroalimentación a nivel lexical muestra como el sistema interpreta las acciones del usuario, el usuario puede así

ver si esta haciendo bien lo que el cree hacer.

- *El feedback inmediato* permite visualizar las acciones de bajo nivel del usuario por ejemplo presionar una cierta tecla o desplazar el cursor del mouse. Este tipo de feedback es necesario notificarlo de inmediato en la interfaz gráfica de la AI.
- *El Feedback contextual* responde a la pregunta ¿Qué es este objeto?, ya que tiene el rol de asistir al usuario en la especificación de su acción actual. Por ejemplo el cursor de doble flecha es utilizado para indicar cambios en las propiedades del objeto seleccionado.

3 El formalismo de los Objetos Cooperativos Interactivos (OCI)

El formalismo de Objetos Cooperativos Interactivos (OCI) [1] permite especificar cada uno de los componentes arquitecturales de una AI en términos de un conjunto de objetos OCI comunicantes. Cada objeto OCI se compone de cuatro elementos, a saber:

$$OCI = Comportamiento + Servicios + Estado + Presentación$$

El formalismo de OCI es un formalismo mixto pues se basa tanto en el paradigma orientado a objetos para especificar el aspecto estático (Servicios y Presentación) como en las redes de Petri para definir el aspecto dinámico (estado y comportamiento) de una AI. Esta sección tiene por objetivo presentar solo un resumen de este formalismo. El lector interesado en obtener mayor información sobre el formalismo de OCI, consultar los trabajos de [1].

3.1 Comportamiento

El comportamiento de un OCI se define como un objeto que reacciona a los estímulos exteriores en función de su estado interno. Este comportamiento es descrito por una red de Petri a Objetos llamada ObCS ("Object Control System") . El ObCS es un tipo de red de Petri orientada a objetos la cual especifica el comportamiento de AI términos de un conjunto de variables de estado (llamadas plazas representadas gráficamente por circulos) mas un conjunto de operadores de cambio de estado (llamados transiciones representadas gráficamente por rectángulos). Las plazas y las transiciones son unidas por arcos etiquetados, indicando el sentido de evolución del estado sistema.

El estado del sistema es representado por la distribución de tokens en las plazas de la red de Petri. Esta red de Petri es calificada de orientada a objetos, ya que los tokens pueden representar valores simples (enteros, boléanos, cadenas de caracteres, etc..) o valores de referencias sobre los objetos de la AI a desarrollar. En cuanto a las transiciones, estas pueden tener asociada una precondición, la cual es una función lógica compuesta de variables provenientes de los arcos de entrada a la misma transición. Una transición esta activada cuando:

- Cada plaza de entrada tiene al menos un token.
- Si hay precondición, los valores de los tokens de las plazas de entrada validan tal precondición.

Cuando una transición se dispara, la acción de la transición se ejecuta la cual puede crear o destruir objetos y puede invocar métodos sobre los objetos especificados en los parámetros de entrada de la acción. El disparo de una transición retira también un token de cada una de las plazas de entrada y deposita un token en cada una de las plazas de salida.

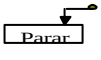
3.2 Servicios

La interfaz (en el sentido informático del termino) de un objeto OCI se define por el conjunto de los servicios para interactuar con su medio. En el caso de una AI, este medio es el usuario mismo u otro sistema. Cada servicio de la interfaz del objeto OCI esta en relación con al menos una transición del ObCS. La relación entre transiciones y servicios esta expresado por *la función de disponibilidad* siguiente:

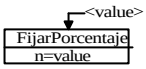
$$Avail: Services \rightarrow \mathrm{P}(T)$$ donde $\mathrm{P}(T)$ es el conjunto de partes de T:

$$\forall s \in Services, Avail(s) \neq \varnothing \text{ y } \forall s, s' \in Services / s \neq s', Avail(s) \cap Avail(s') = \varnothing$$

En un OCI se distinguen dos tipos de servicios:

- *Los servicios del usuario* son las transiciones del usuario y están en relación directa en la parte presentación de un OCI. La representación grafica es una transición con un flecha quebrantada con un circulo gris: 

- *Los servicios internos* son ofrecidos a los objetos de a la aplicación. La representación gráfica es una transición

solo con una flecha quebrantada, por ejemplo el servicio interno *FijarPorcentaje* es como sigue: 

3.3 Estado

El estado de un OCI esta determinado por la distribución de tokens (o marcaje) en cada una de las plazas del ObCS. El echo de que todo servicio este asociado a una de las transiciones de la red de Petri del ObCS es posible determinar a todo momento la manera de la interacción en que el estado influye sobre la disponibilidad de los servicios. De una manera recíproca, es posible determinar la influencia de la ejecución de un servicio sobre el estado de un objeto OCI.

3.4 Presentación

La parte Presentación define la interfaz de usuario, la cual se compone por un conjunto de ventanas conteniendo interactores tanto de salida como de entrada de información del usuario. El formalismo OCI propone solo la gestión de la información de entrada en los objetos de interacción en términos de la *función de activación* siguiente:

Activation : (IxEvent)→ Services

La función de activación asocia la acción del usuario (representada por un Evento) sobre un interactor (I) que activa la ejecución de uno de los servicios (Services) del ObCS.
El presente trabajo integra la *función de retroalimentación* con el fin de habilitar el elemento de Presentación para el despliegue de información en la parte externa de una AI.

Feedback : PÈT® P(I)

La función de retroalimentación anterior pone en relacion el conjunto de plazas (P) y/o transiciones (T) del ObCS con el conjunto de interactores (I) de la interfaz de usuario.

El formalismo de ICO se basa en las redes de Petri de alto nivel para especificar los diferentes aspectos de una aplicación interactiva interacción. Uno puede imaginarse asociar la especificación a cualquier de los elemento de las red de Petri, es decir plazas, transiciones o arcos. En una red de Petri, las plazas son las variables del estado de un sistema (el estado que es una distribución de tokens en las plazas) y las transiciones son los operadores de cambia de estado. Los arcos modelan la estructura conectiva que determina las precondiciones previas de los cambios de estado, y su efecto sobre el estado del sistema.
Nosotros consideramos que la retroalimentación gestiona el estado del sistema: el principal objetivo de la retroalimentación es hacer que la información asociada al estado de una AI este disponible al usuario. Es un tanto lógico asociar la especificación de retroalimentación en una primera instancia con las plazas de una red de Petri.
Sin embargo una análisis mas profundo de los tipos de retroalimentación revela que es necesario también asociar la especificación la retroalimentación a las transiciones de la red: en el formalismo de ICO, las transiciones modelan las acciones atómicas (no interrupible) del sistema. Las transiciones se caracterizaran frecuentemente por hacer llamadas al núcleo función (la parte no interactivo de la aplicación) y estas llamadas de la función pueden tomar un tiempo arbitrario para completarse. Conforme a la taxonomía de retroalimentación resulta necesario notificar al usuario sobre el tiempo de ejecución de una operación (por ejemplo cambiando la forma del cursor del ratón o notificando el avance de la operación a través de un mensaje de progreso). Es posible imaginar un modelo de dialogo con una granularidad fina y asociado a varios subestados (modelados en términos de plazas) para el desarrollo de la operación, pero esta haría perder la naturaleza no interrupible de una operación. La no interruptibilidad de una operación puede recuperarse solamente con una estructura mucho más compleja de la red de Petri del ObCS. *La razón anterior nos permite asociar la especificacion de la retroalimentación visual en las transiciones así como en las plazas del ObCS.*

3.4.1 La retroalimentación en las plazas

Las plazas de un red de Petri representan las variables de estado de un sistema, lo que permite a la retroalimentación hacer visible al usuario el estado interno del sistema. La asociación entre las plazas del ObCS y los interactores de la presentación es establecida por uno de los tres métodos de retroalimentación siguientes:

- Un método **`token_Entrante()`**, el cual es ejecutado cada vez que un token es agregado a la plaza en cuestión.
- El método **`token_Saliente()`**, es ejecutado cada vez que un token es removido de la plaza en cuestión
- El método **`token_Modificado()`** es ejecutado cada vez que un token es previamente almacenado en una plaza con una transición.conectada con un arco bidireccional.

Estos tres métodos de retroalimentación son asociados a las plazas de la red de Petri, y pueden tener acceso a la informacion de todos los tokens contenidos en las plazas. Los métodos de retroalimentación reciben como parámetros la plaza misma, el token y el conjunto de objetos de interacción implicados en la retroalimentación. La

especificación de la signatura de un método de retroalimentación es de la manera siguiente:

```
token_Saliente(Plaza aPlaza, Token aToken, ConjuntoDeInteractores aInteractores);
```

3.4.2 La retroalimentación en las transiciones

Cualquier transición de un ObCS puede tener asociada una acción a notificar al usuario. Esta acción se define como una función que toma como parámetros de entrada (respectivamente de salida) las variables de los arcos de entrada (respectivamente salida) de la transición en cuestión. En el cuerpo de una acción puede utilizarse cualquier tipo de retroalimentación visual enviándolo a los interactores asociados a la transición especificados por la función de retroalimentación. De esta manera, es posible realizar la retroalimentación justo cuando la acción de la transición comienza a ejecutarse (e.g. cambiando la forma del cursor del ratón), durante varias etapas de la ejecución (e.g. actualizando continuamente con una barra de progreso), y justo después de haberse terminado la acción (e.g. regresando el cursor del ratón a su forma original).

3.5 Consideraciones arquitecturales

De entre los módulos de la arquitectura de Seheeim (ver Figura 1), el módulo de Dialogo es el más difícil de diseñar dado que en el se determina el comportamiento de una AI. Utilizando el formalismo de OCI, el ObCS especifica el comportamiento de una AI. En el ObCs el adaptador del núcleo funcional y el núcleo funcional mismo son modelisados por el conjunto de tokenes circulando en la red de Petri. El elemento Presentación del formalismo de OCI define el modulo del mismo nombre de la arquitectura de Seheeim. Gracias a la integración de la función de retroalimentación aquí propuesta, el componente Presentación no solo cubre la gestión de la entrada de información, sino también de la salida de información.

Ahora bien, en cuanto a la comunicación entre el modulo de Dialogo y el Adaptador del núcleo funcional esta determinado por el flujo de tokens en la red de Petri del ObCS y por las llamadas de métodos en los tokens al interior de las acciones de las transiciones.

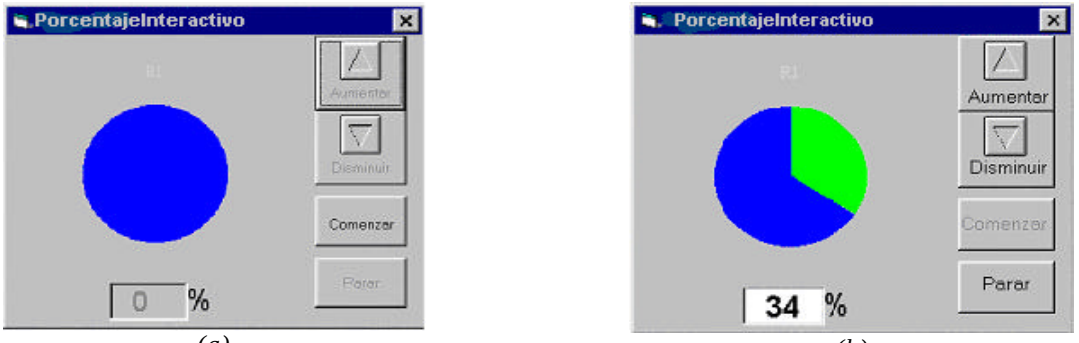
Este trabajo considera que la comunicación entre los módulos de Dialogo y de Presentación es mucho más compleja, ya que es necesario especificar el dialogo entre el usuario con el AI en términos de la información de la activación y de la retroalimentación.

- En cuanto a la activación, el módulo de Presentación influencia al Dialogo a través la ocurrencia de eventos lo que provoca que la Presentación deposita un token en cada una de las plazas de eventos. Una transición de puede tener a lo máximo una plaza de eventos en entrada. La plaza de eventos es necesaria por el hecho que un evento dado puede activar diferentes transición de un sistema, en función del estado interno del sistema.

- En lo que concierne a la retroalimentación, el estado de Dialogo (i.e. el marcaje de la red ObCS) puede influenciar el modulo Presentación. En función del estado del Dialogo, diversas transiciones eventos pueden ser activadas lo que implica un estado de disponibilidad en los interactores asociados. En el caso donde una transición no sea activada, los interactores asociados son desactivados en la interfaz de usuario.

4 Caso de estudio

Con el fin de ilustrar la eficiencia del formalismo OCI para especificar la retroalimentación visual, se presenta en esta sección el caso de estudio titulado “PorcentajeInteractivo”. La aplicación *PorcentajeInteractivo* es útil par al usuario para modificar y visualizar fácilmente el valor numérico de una variable porcentaje que varia entre el rango de valores de 0 a 100.



(a) *(b)*

Figura 3. Visualización del estado inicial (a) y de un estado valido (b) de la aplicación PorcentajeInteractivo

La aplicación cuenta con dos ínteractores de entrada para modificar la variable porcentaje (ver Figura 3):

- El botón *Aumentar* permite incrementar el valor del porcentaje en una unidad. El botón esta disponible cuando

el valor del porcentaje llega a ser igual a 100.

- El botón *Disminuir* permite disminuir el valor del porcentaje en una unidad. El botón esta disponible cuando el valor del porcentaje llega a ser igual a 0.

La aplicación *PorcentajeInteractivo* ofrece dos interactores de entrada y salida de la variable porcentaje, a saber:

- Un interactor *zona de texto* es una vista textual don el usuario puede teclear un nuevo valor de porcentaje.
- Un interactor *circular* donde el usuario tiene representación gráfica y puede modificar directamente el valor del porcentaje. Esto ultimo se logra cuando el usuario sitúa el cursor del ratón sobre la forma circular, para después presionar el botón izquierdo del ratón y mover el cursor del ratón hasta el valor deseado.

La Figura 3a muestra el estado inicial de cuando se lanza la aplicación *PorcentajeInteractivo* donde el usuario inicialmente solo puede presiona el botón *Comenzar.* Una vez presionado dicho botón todos los interactores que permiten modificar el porcentaje estarán disponibles hasta que el usuario presiona el botón *Parar.* Este ultimo botón inhibirá la mayoría de los objetos interacción de la aplicación y regresa la aplicación al estado inicial.

La Figura 3b muestra un estado de interacción donde la variable porcentaje tiene un valor de 34%. Es importante observar las coherencia entre la representaciones textual y gráfica de dicha variable, con esto se logra que el usuario puede modificar e interpretar sin confusión el valor del porcentaje en cada una de vistas.

4.1 Servicios

Dentro del paradigma orientada objetos, la aplicación *PorcentajeInteractivo* puede ser diseñada por una sola clase. Los servicios de esta clase conforme al formalismo OCI se define de la manera siguiente:

```
Class PorcentajeInteractivo
Services //Definición de la interfaz de la clase
     Comenzar,Parar,Aumentar, Disminuir;
     TomarCursor,LiberarCursor;
     TeclearValor <x:Str>;
     MoverCusor<x,y:Int>;
```

Figura 4. Servicios de la clase PorcentajeInteractivo.

4.2 Comportamiento

El comportamiento de la clase *PorcentajeInteractivo* es expresado en términos del conjunto de plazas, arcos y transiciones que conforman la red de Petri del ObCS siguiente:

```
ObCS
Places // Definición de tipos plazas y del marcaje inicial
  //Las plazas Inactivo y NoTrazar tienen al inicio un token simple
  Inactivo : <> = {<>};
  NoTrazar : < > = {<>};
  //Las plazas Activo y Trazando aceptan tokens simples y vacías al inicio
  Activo: <> = {};    Trazando : <> = {};
  // La plaza Porcentaje contiene un token entero, cuyo valor inicial es 0
  Porcentaje: <Int> = {<0>};
Transitions// Definición de las transiciones y sus componentes
  T1,T2: {}
  T3: Precondition { n > 0; }
       Action { n = n – 1; }
  T4: Precondition { n < 100; }
       Action { n = n + 1; }
  T5:  Action { P_CalcTxt(Str); }
  T6:   Action { n=Valor; }
  T7,T8:{}
```

4.3 Estado

Gracias a la representación grafica de la red de Petri a objetos del ObCS anterior es posible especificar explícitamente el estado de la aplicación *PorcentajeInteractivo* en cualquier momento de la interacción. Además, en este trabajo se propone la forma de plazas de circulo estrelladas para especificar las plazas de retroalimentación, como es el caso de las plazas *Trazando* y *Porcentaje (ver **Figura 5**)*.

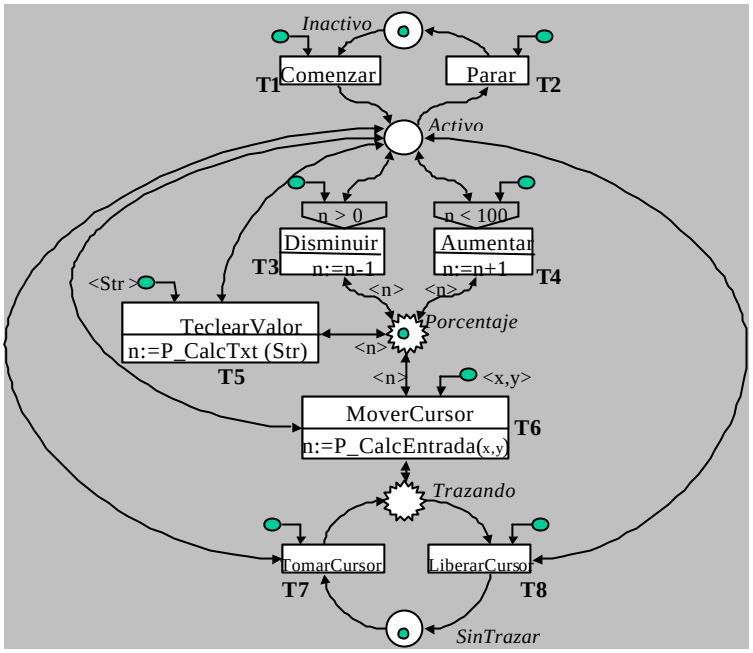


Figura 5.Especificación del dialogo de la aplicación PorcentajeInteractivo.

La ***Figura 5*** muestra la distribución de los token en la red de petri del ObCS en la que la transición *T1* este activa y en la interfaz corresponde a la disponibilidad del servicio proporcionado por el botón *Comenzar (*Ver Figura 3).

```
Inactivo ={};
Activo= {< >};
Trazando = {};
NoTrazar  = {< >};
Porcentaje = {<34>};
```

Figura 6. Un marcaje posible de la aplicación de la Figura 3b.

Una vez que el usuario presiona sobre el botón *Comenzar* la transición *T1* pasa el token de la plaza *Inactivo* a la plaza *Activo*, generando un marcaje donde se activan las transiciones de servicios (*T3, T4, T5 y T6*) las cuales permiten modificar el valor porcentaje. La activación de estas transiciones implica que los interactores de la interfaz de usuario de la aplicación *PorcentajeInteractivo* estén disponibles como lo muestra la Figura 3b. Note que el marcaje mostrado por esta última figura se muestra en la ***Figura 6***.

4.4 Presentación

La especificación de la Presentación basado en el formalismo ICO esta definida en términos de la función de activación y de retroalimentación.
la funcion de activacion (***Tabla 1***) especifica los eventos a los que reacciona cada uno de los interactores de la interfaz de usuario de la aplicación *PorcentajeInteractivo* y asocia a la vez a cada interactor el conjunto de servicio especificados en la sección 4.1. Por ejemplo, el interactor *Zona de Texto* reacciona a las entradas de datos provenientes del teclado ejecutando así el servicio *TeclearValor* de la clase de la Figura 4.
Ahora bien, la función de retroalimentación propuesta en este trabajo en la sección 3.4 permite asociar los nodos de la red Petri del ObCS de la ***Figura 5*** con los interactores del elemento Presentación, tal como lo muestra la ***Tabla 1***.
Gracias a la integración de la función de retroalimentación (ver ***Tabla 2***) es posible identificar las plazas de retroalimentación siguientes:

- La plaza *Porcentaje* ofrece una retroalimentación de tipo *estado del sistema* sobre el valor del porcentaje visualizado tanto en su versión grafica como textual.
- La plaza *Trazando* ofrece una retroalimentación de tipo *feedback inmediato* asociando un curso en forma de cruz cuando el usuario modifica el valor del porcentaje a través del interactor *Circulo* y un cursor de flecha cuando se deja de modificar dicho valor.

En este caso de estudio, ningún otra especificación de retroalimentación visual debe ser agregada a las transiciones del ObCS.

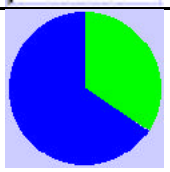
Ineractor	Rep. Gráfica	Eventos	Servicios
Botón Comenzar		Clic	Comenzar
Botón Parar		Clic	Parar
Botón Aumentar		Clic	Aumentar
Botón - Disminuir		Clic	Disminuir
ZonaDe Texto		KeyPress	Teclear-Valor
Circulo		Mouse Move	Mover-Cursor
		Mouse Down	Tomar-Cursor
		MouseUp	Liberar-Cursor

Tabla 1. Función de activación

Estado	Tipo de método	Rep. gráfica
Inactivo		Ningún feedback
Activo		Ningún feedback
Porcentaje	Token_Entrante	No hay arco de entrada
	Token_Modificado	**Estado del sistema** valor del porcentaje en la ZonaTexto y circulo. 
	Token_Saliente	No hay arco de entrada
Trazando	Token_Entrante	**Feedback Inmediato** Cursor del ratón en forma de cruz +
	Token_Modificado	No hay arco de entrada
	Token_Saliente	**Feedback Inmediato** Restablecer cursor normal del ratón
NoTrazar		Ningún feedback

Tabla 2. Función de retroalimentación.

5 La taxonomía de feedback visual en el formalismo de OCI

En esta sección se argumenta a detalle cómo la retroalimentación es especificada en el formalismo OCI conforme a la taxonomía descrita en la sección 2.

5.1 La activación

La retroalimentación de activación tiene por objetivo visualizar el espacio de interacción al usuario. En una especificación a base de OCI, el espacio de interacción puede ser calculado a partir del marcaje actual del ObCS, el cual determina el conjunto de transiciones a disparar. Para esto, la especificación utiliza la función de disponibilidad (relación entre servicios y transiciones) y la función de activación (relación entre objetos de interacción y servicios).

Sea T_e Ì T el conjunto de transiciones activadas

La función de disponibilidad relaciona un servicio con un conjunto de transiciones. Así en S_a Ì *Services* el conjunto de e servicios disponible es:

$$S_a = Avail^{-1}(T_e)$$

El conjunto de servicios del usuario disponibles es:

$$S_{ua} = S_a \text{ Ç } S_u$$

La función de activación *Activation* relaciona un par (Interactor, Acción) con un servicio del usuario. En todo momento, el conjunto de interactores activados por lo tanto se define como:

$$Proy_W(Activation^{-1}(S_{ua}))$$

En función del marcaje actual del ObCS, es posible determinar el conjunto de interactores activos (o inactivos) en cualquier instante de la interacción. Es importante menciona que la estructura de la red de Petri y la función de la activación son suficiente para efectuar la retroalimentación de activación de una manera automática.

5.2 La navegación

La red de Petri del ObCS puede utilizarse para especificar el diálogo a diferentes niveles de granularidad. A un nivel

fino el ObCS puede describir la estructura de diálogo al interior de una ventana. A un nivel de granularidad mas grueso el ObCS puede describir la navegación entre ventanas de una aplicación. En este ultimo caso, los tokens transitando en las plazas de la red de Petri del ObCS son referencias a otros objetos OCI representando diferentes ventanas de la aplicación. La retroalimentación de navegación se lleva a cabo a través de los métodos de retroalimentación asociados a las plazas. La entrada de un token a una plaza implica la apertura de una ventana en la interfaz de usuario, mientras que el retiro de un token acciona el cierre de la ventana. Esta forma de especificar la navegación es similar a la propuesta por [9].

5.3 Estado de diálogo

Dado que el diálogo es modelado por una red de Petri orientada a objetos, el estado del diálogo se define totalmente por el marcaje del ObCS. El marcaje corresponde a la distribución de tokens en las plazas del ObCS, y los tokens pueden representar cualquier clase de datos que se apliquen al diálogo. Como consecuencia, la retroalimentación del estado de dialogo es afectada por los métodos de retroalimentación a las plazas del ObCS.

5.4 El estado del sistema

En un objeto OCI, es posible que los tokens del ObCS contengan referencias de objetos que pertenezcan al núcleo funcional. La retroalimentación del estado del sistema especificada de manera similar que la retroalimentación del estado del diálogo (§ 0) por medio de los métodos de retroalimentación. Los métodos de retroalimentación pueden ejecutar algoritmos arbitrariamente complejos, permitiendo el acceso a los atributos o métodos de los objetos perteneciente a los tokens, con el fin de calcular (o obtener) la información a visualizar.

En el caso de estudio, el estado del sistema es especificado por un único token, el cual contiene el valor del porcentaje. El feedback del valor porcentaje se obtiene utilizando solo un método de retroalimentación asociada a la plaza Porcentaje, este método es ejecutado a cada acceso del token de esta plaza:

```
Token_Modificado(Plaza aPlaza, Token aToken, ConjuntoDeInteractores aInteractores){
//La función de retroalimentación asocia los interactores
// Zonatexto y Circulo a la plaza Porcentaje
     Por todo W dentro aInteractores
       w. VisualizarValor(aToken.Valor);
}
```

El código anterior supone que todo interactor tiene un método *VisualizarValor()*, lo que ciertamente no es el caso de toda caja de herramientas (toolkit) de software. Esto muestra la necesidad de crear una caja de herramientas abstracta (como la toolkit AWT de java) que permita aislar la idiosincrasia de un ambiente grafico especifico, y así asociar a cada interactor de la interfaz de usuario el conjunto de métodos requeridos en el modulo de diálogo.

5.5 La notificación

El hecho que el formalismo de OCI sea orientado a objetos permite que una aplicación interactiva se especifique en términos de una clase. Como se estableció en la sección 3.2,los servicios en una clases pueden ser solicitados por eventos activados por el usuario, y estos servicios forman parte de la función de activación. Otros servicios son simples métodos en el sentido usual del enfoque orientado a objeto, y son susceptibles de ser llamados por otros objetos de la aplicación. Bajo esta óptica, es posible que los objetos del núcleo funcional notifiquen la ocurrencia de un evento de interés al usuario. Eso se logra llamando a un servicio definido en la clase del OCI. La función de disponibilidad relaciona los servicios con las transiciones y la invocación de un servicio dará lugar al disparo de una transición en el ObCS del OCI.

La retroalimentación de notificación puede ser especificada usando la función de retroalimentación asociada ya sea a las plazas de la red de Petri o a la acción de las transiciones de servicio de la misma red.

5.5 Los mensajes de error

Las funciones del núcleo funcional de la aplicación son llamadas por las acciones de las transiciones de la red de Petri. Estas acciones tienen la posibilidad de visualizar los errores que se pueden generar en las llamadas de función usando cualquier interactor asociado a la transición.

5.6 El feedback inmediato

El feedback inmediato es llevado a cabo por los métodos de retroalimentación asociados a las plazas del ObCS. En el caso de estudio existen dos situaciones de retroalimentación inmediata. La primera situación es asignar dar puntero del mouse un curso de cruz justo cuando el usuario inicia el cambio del valor del porcentaje a través del interactor circulo, el método de retroalimentación asociado es el siguiente:

```
token_Entrante(Plaza aPlaza, Token aToken, ConjuntoDeInteractores aInteractores){
//La función de retroalimentación asocia los interactores
```

```
// el interactor Circulo a la plaza Trazando
//Los otros parámetros no son utilizados
     Por todo W dentro aInteractores
       w.VisualizarCursor(Cursor_DeCruz);
}
```

La segunda forma de feedback inmediato es asignar el cursor de flecha justo cuando el usuario termina de modificar dicho valor del porcentaje a través del interactor Circulo, el método de retroalimentación asociado es:

```
token_Saliente(Plaza aPlaza, Token aToken, ConjuntoDeInteractores aInteractores){
     Por todo W dentro aInteractores
       w.VisualizarCursor (Cursor_Normal);
}
```

5.7 El feedback contextual

De una manera similar al feedback inmediato, el feedback contextual además de utilizar los métodos de feedback asociados a las plazas del ObCS, toma información sobre el estado y el tipo token para determinar el tipo de puntero de ratón a mostrar en la acción del usuario.

6 Verificación formal

La verificación formal tiene como objetivo probar las propiedades basándose en una la especificación formal de una AI. Al tomar en cuenta el aspecto externo en la concepción de AI, se tiene generalmente como consecuencia no solo evaluar aspecto que están relacionadas a facilitar la tarea del usuario y visualizar la información sino también a evaluar aspectos ligadas a utilidad misma del sistema. Por tal razón este trabajo propone analizar propiedades relacionadas tanto al aspecto interno y externo de una AI.

6.1 Propiedades internas

Las propiedades internas son para evaluar aspectos relacionados a la funcionalidad de una AI. Algunas propiedades son de gran interés para la especificación de un sistema donde la teoría de redes de Petri ofrece técnicas predefinidas para comprobarlas. Por ejemplo un sistema es *reinicialisable,* solo si existe una secuencia de acciones que permitirán al sistema regresar a su estado inicial a partir del estado actual de la AI. Un sistema se califica de *acotado* si no hay producción o extinción de recursos durante la actividad de la AI. Esto es una característica importante ya que garantiza que el sistema no derrocha recursos. Un AI se califica de *dinámica* si es posible activar al menos una transición en cualquier estado del sistema.

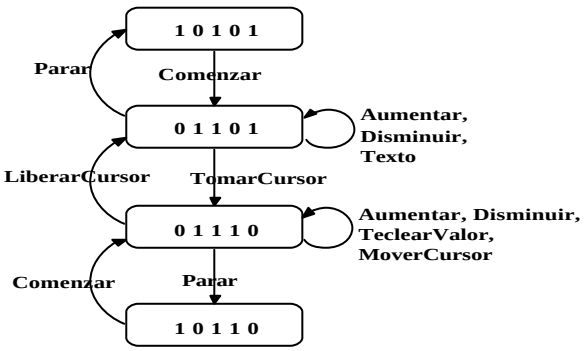


Figura 7.Grafo de marcaje correspondiente a la red de Petri de la figura 6.

Las propiedades anteriores puede obtenerse a partir de un grafo de marcaje. El grafo de marcaje es un autómata con un número finito de estados y puede ser generado automáticamente a partir de una red de Petri. Varias herramientas son actualmente disponibles con este fin, un ejemplo es el sitio web http://www.daimi.au.dk/PetriNets/..

El grafo de marcaje (ver ***Figura 7***) de la aplicación *PorcentajeInteractivo* muestra un numero de tokens constante y que en cualquier estado hay siempre una acción que puede ejecturar indicando respectivamente que la aplicación es acotada y no pasiva. La aplicación *PorcentajeInteractivo* es reinicialisable ya que a partir de cualquier estado existe una secuencia de acciones finitas que permite al grafo de marcaje regresar al estado inicial (ver ***Figura 7***)

6.2 Propiedades externas

Los trabajos de Gram y Cockton (6) han clasificado un gran número de propiedades sobre el aspecto externo de una AI, donde la mayoría han sido especificadas formalmente utilizando lenguajes formales declarativos tales como las lógicas temporales. Entre las propiedades más sobresalientes encontramos, la observabilidad, la insistencia y la honestidad. *La observabilidad* es cuando una AI asegura que toda la información relevante este disponible al

usuario. *La insistencia* es cuando una AI garantiza al usuario visualizar la información necesaria y suficiente para realizar su tarea. *La honestidad* implica que el sistema asegura que el usuario interpreta correctamente la información percibida. *La predictibilidad* significa que el usuario puede anticipar el estado y tiempo de respuesta del sistema a partir de las interpretaciones del estado actual de dicho sistema.
Realizar pruebas formales sobre estas propiedades resulta difícil con técnicas de análisis matemático provenientes de las redes de Petri, ya que además es necesario modelar el comportamiento cognoscitivo del usuario y tener medios para realizar pruebas junto con el usuario. A pesar de estas dificultades, es importante indicar que el formalismo OCI puede ayudar a demostrar cuando una especificación de una AI no satisface una de estas propiedades. Por ejemplo, si el conjunto de acciones no esta disponible al usuario el sistema no puede ser calificado de observable. Este mismo principio es valido para la propiedad de predictibilidad. Las técnicas de especificaciones de la retroalimentación visual en términos de OCI junto con las técnicas de análisis que ofrecen las redes de Petri parecen ser una dirección muy prometedora para verificar las propiedades sobre el aspecto externo de una AI.

7 Conclusiones

La aportación fundamental del presente trabajo es la integración en el formalismo de Objetos Cooperativos Interactivos (OCI) de la especificación formal de la retroalimentación visual en base a las plazas y transiciones de las redes de Petri Orientadas a Objetos. La presente aportación caracteriza al formalismo de OCI en los aspectos siguientes:

- El formalismo de OCI soporta especificar sin ambigüedad y a diferentes niveles de abstracción de información la respuesta visual de una AI.
- El formalismo OCI cubre por completo el triple rol de la retroalimentación visual es decir mantener informado al usuario: sobre sus acciones, sobre el comportamiento de la AI y sobre los cambios de estado de la AI a diseñar.
- El uso de las técnicas de especificacion de la retroalimentación visual de OCI junto con las técnicas de análisis de la redes de Petri parece ser una dirección prometedora para verificar las propiedades externas de una AI.

Las expectativas de la presente contribución son diversas. Una de ellas es de tomar en cuenta la tarea del usuario dentro de la especificación formal de la retroalimentación con el fin de realizar pruebas sobre las propiedades externas en términos del formalismo OCI. Otras de las expectativas es extender la especificación formal de la retroalimentación a diversos tipos tales como el feedback sonoro, animado, y haptico. El objetivo es la investigación en el diseño y desarrollo de sistemas altamente interactivos, en particular en los sistemas multimedia y multimodal.

8 Referencias

[1] Bastide, R. and Palanque, P., "Conformance and Compatibility between Models as Conceptual Tools for a Consistent Design of Interactive Systems," *CHI 99 workshop on Tool Support for Task-Based User Interface Design*, 1999

[2] Campos, J. C. and Harrison, M., "*Formally verifying interactive systems: A review*," *4th Eurographics workshop on Design, Specification and Verification of Interactive System (DSV-IS'97)*, Springer, 1997, pp. 109-124.

[3] Dix, A. J., "Findig Out: event discovery using status-event analysis," *Formal aspects of the human computer interaction , BCS-FACS Workshop*, UK, 1998, pp. 214-220.

[4] Dix, A. J., Russell, B., and Wood, A., "Architectures to make Simple Visualisations using Simple Systems," *Advanced Visual Interfaces - AVI2000*, ACM Press, Italy, 2000, pp. 51-60.

[5] Dragicevic, P. and Jean-Daniel , F., "Input Device Selection and Interaction Configuration with ICON (Conférence)," *actes de la Conférence Internationale IHM-HCI 2001*, Springer Verlag, Lille, France, 2001

[6] Gram, C. and Cockton, G., *Design principles for interactive software*, Chapman & Hall 1996.

[7] Munoz, A. J., "Una Taxonomía de Retroalimentación Visual para las Aplicaciones Interactivas," *Conferencia Latinoamericana de Estudios en Informática (CLEI 2001)*, 24-28 Sept del 2001 en la Universidad de los Andes.

[8] Preece, J., Rogers, Y., and Sharp , H., *Interaction Design*. John Wiley & Sons. ., New York, NY: 2001.

[9] Schlungbaum, E., "Support of Task-based User Interface Design in TADEUS," *CHI'98 Workshop* , 1998.

[10] Tovi, G., Ravin, B., Gordon, K., George, F., Azam, K., and Bill, B., "Creating principal 3D curves with digital tape drawing," *ACM CHI Letters*, Vol. 4, No. 1, 2002, pp. 121-128.

[11] Vanderdonckt, J. and Puerta, A., *Computer-Aided Design of User Interfaces II*, II ed., Kluwer academic publisher, Dordrecht, Netherland, 1999.