(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 989 018**

(51) Int. CI.:

G05B 17/02 (2006.01)
G06F 30/20 (2010.01)
G05B 19/418 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **18.04.2019 E 19170105 (1)**

(97) Fecha y número de publicación de la concesión europea: **04.09.2024 EP 3726305**

(54) Título: **Control de procedimiento con gemelos digitales**

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
25.11.2024

(73) Titular/es:

ASCON SYSTEMS HOLDING GMBH (100.0%)
Curiestrasse 5
70563 Stuttgart, DE

(72) Inventor/es:

GREFEN, KILIAN

(74) Agente/Representante:

QUIRÓS MARÍN, María

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

DESCRIPCIÓN

Control de procedimiento con gemelos digitales

Campo de la invención

La presente invención se refiere al campo del control de procedimiento en tiempo real de sistemas de fabricación usando gemelos digitales. En más detalle, la presente invención se refiere al campo del modelado de sistemas de fabricación distribuidos de eventos discretos usando gemelos digitales y al uso posterior de los modelos para controlar en tiempo real de sistemas de fabricación distribuidos discretos uniformes.

Antecedentes de la técnica

El control de procedimiento en tiempo real en la fabricación es un desafío constante y está sometido a desarrollo continuo. La figura 1 muestra un ejemplo típico de un entorno de fabricación sometido a control de procedimiento en tiempo real.

Tal como se muestra en la figura 1, el entorno de fabricación considerado puede comprender, por ejemplo, una primera línea 1. La primera línea 1 puede estar configurada por tres transportadores, es decir el transportador 11, el transportador 12, y el transportador 13, cada uno dotado de un sensor 11, 12, y 13, respectivamente.

Tal como se muestra en la figura 1, el entorno de fabricación considerado también puede comprender, por ejemplo, una segunda línea 2. La segunda línea 2 puede estar configurada por tres transportadores, es decir el transportador 21, el transportador 22, y el transportador 23, cada uno dotado de un sensor 21, 22, y 23, respectivamente.

Tal como se muestra en la figura 1, el transportador 11 de la línea 1 porta una caja y el objetivo global del control de procedimiento puede ser un movimiento seguro de la caja a lo largo de la línea 1.

La figura 2 muestra un modelo de árbol jerárquico que es un modelo típico usado para el control de procedimiento del entorno de fabricación mostrado en la figura 1.

Tal como se muestra en la figura 2, en el modelo de árbol jerárquico diferentes subsistemas del entorno de procedimiento se representan como nodos. Normalmente, el modelo de árbol jerárquico se prepara antes del inicio del control de procedimiento. La asignación de un subsistema a un nivel de jerarquía en el modelo de árbol jerárquico puede ser específica de aplicación y habitualmente diferentes capas en el modelo de árbol jerárquico se hacen funcionar de manera independiente.

Además, no hay ninguna opción de modificar el modelo de árbol jerárquico generado una vez que empieza el control de procedimiento ya que los subsistemas del entorno de fabricación están codificados de manera rígida en el modelo de árbol jerárquico sin flexibilidad. Si, por ejemplo, debe configurarse una nueva línea para el entorno de fabricación o si cambia la aplicación del mismo, entonces esto requerirá una completa recodificación del modelo de árbol jerárquico subyacente al control de procedimiento. Además, un cambio de funcionalidad de los diferentes subsistemas necesitará un procedimiento de remodelado para el modelo de árbol jerárquico.

El documento de patente US 2018/0039249 A1 describe un control dinámico automatizado de operaciones y diseño de sistemas físicos a lo largo del tiempo, incluyendo componentes mecánicos y sensores asociados. Un ordenador está programado con un modelo de optimización operativa para el producto instalado, incluyendo modelado de los componentes mecánicos y sensores. El ordenador también está programado con un modelo de estimación de la probabilidad que proporciona una estimación de una medida de diferencia relacionada con la entrada y salida del producto instalado. Se reciben datos de sensor, que se habían generado durante el funcionamiento del producto instalado. Se genera una salida a partir del modelo de optimización operativa basándose en los datos de sensor recibidos. Se genera una salida a partir del modelo de estimación de la probabilidad basándose al menos en parte en la salida a partir del modelo de optimización operativa. Se modifica al menos un componente de producto instalado basándose al menos en parte en la salida a partir del modelo de estimación de la probabilidad. Además, el documento de patente US 2016/0247129 A1 describe un sistema para usar gemelos digitales para mantenimiento predictivo de máquina basado en modelo, ajustable a escala, que comprende una pluralidad de gemelos digitales y una plataforma de simulación. La pluralidad de gemelos digitales corresponde a una pluralidad de máquinas físicas ubicadas de manera remota. Cada gemelo digital respectivo comprende: datos de placa de características de producto correspondientes a una máquina física única, uno o más modelos de simulación, y una base de datos que comprende datos de registro de tiempo de ejecución recopilados a partir de sensores asociados con la máquina física única. La plataforma de simulación está configurada para procesar modelos de simulación correspondientes a la pluralidad de gemelos digitales usando una pluralidad de sistemas informáticos de múltiples procesadores.

El documento de patente US 2013/0204602 A1 divulga un aparato de simulación para verificar un sistema híbrido. El aparato de simulación incluye una unidad de entrada de modelo de sistema, una unidad de almacenamiento de información de modelo, una unidad de simulación, y una unidad de visualización de resultado.

Finalmente, el documento de patente US 2015/0006122 A1 describe un método y un aparato para simulación probabilística entre organizaciones en tiempo real.

Sin embargo, a la vista de la creciente complejidad de entornos de procedimiento, los enfoques existentes para control de procedimiento ya no cumplen los requisitos para situaciones de aplicación futuras.

Sumario de la invención

A la vista de lo anterior, el objetivo de la presente invención es aumentar la flexibilidad en el modelado de sistemas de fabricación distribuidos de eventos discretos para un control de procedimiento en tiempo real eficiente posterior de los mismos.

La invención se define mediante las reivindicaciones independientes adjuntas. En las reivindicaciones dependientes adjuntas se presentan aspectos opcionales de la invención.

Breve descripción de los dibujos

A continuación se explicará la presente invención y ejemplos relacionados con referencia a los dibujos, en los que:

la figura 1 muestra un ejemplo típico de un entorno de fabricación sometido a control de procedimiento en tiempo real;

la figura 2 muestra un contorno de un modelo típico subyacente al control de procedimiento del entorno de fabricación mostrado en la figura 1;

la figura 3 muestra ejemplos de modelado en una capa de metamodelo usando metaobjetos como tipos de metamodelos de subsistemas;

la figura 4 muestra ejemplos de modelado en una capa de modelo de uso o, de manera equivalente, capa de plantilla para definir agrupaciones de metamodelos de subsistemas;

la figura 5 muestra ejemplos de modelado en una capa de gemelo para describir subsistemas como instancias de agrupaciones de metamodelos de subsistemas;

la figura 6 muestra un resultado de modelado a un metanivel tal como se muestra en la figura 3 en combinación con el modelado a un nivel de uso tal como se muestra en la figura 4 y en combinación con el modelado en una capa de gemelo tal como se muestra en la figura 5;

la figura 7 muestra un contorno general de un modelo de espacio de estados usado según la presente invención para el modelado de sistemas distribuidos de eventos discretos;

la figura 8 muestra una representación detallada adicional del modelo de espacio de estados mostrado en la figura 7;

la figura 9 muestra una representación detallada adicional del modelo de espacio de estados mostrado en la figura 7;

la figura 10 muestra una visión general para un sistema adaptado para modelar un sistema distribuido de eventos discretos y para ejecutar control de procedimiento en tiempo real de los sistemas distribuidos discretos uniformes según la presente invención;

la figura 11 muestra un diagrama de flujo básico de funcionamiento para el sistema adaptado para modelar un sistema distribuido de eventos discretos y para ejecutar control de procedimiento en tiempo real de los sistemas distribuidos discretos uniformes según la presente invención;

la figura 12 muestra un diagrama esquemático de un metamodelo implementado por ordenador según la presente invención que se usa por el motor de modelado mostrado en la figura 10;

la figura 13 muestra un diagrama de flujo de funcionamiento para el motor de modelado según la presente invención tal como se muestra en la figura 10;

la figura 14 muestra una división de un sistema distribuido de eventos discretos según la presente invención, en la que la división se define como un conjunto de agrupaciones de subsistemas de tal manera que todos los subsistemas asignados a una agrupación de subsistemas específica se modelan con un mismo tipo de metaobjeto;

la figura 15 muestra un diagrama esquemático de un motor de servicio según la presente invención, en la que el motor de servicio genera un modelo de uso de al menos un subsistema de un sistema distribuido de eventos

discretos correspondiente a un tipo de metaobjeto usado para modelar el al menos un subsistema;

la figura 16 muestra un diagrama de flujo de funcionamiento para el modelado a nivel de uso según la presente invención tal como se muestra en la figura 14 y la figura 15, respectivamente.

la figura 17 muestra un diagrama esquemático de un motor de gemelos virtuales según la presente invención, en la que el motor de gemelos virtuales tiene preinstalado al menos un núcleo de software de modelado ejecutable preparado por el motor de servicio tal como se muestra en la figura 15;

la figura 18 muestra un diagrama de flujo de funcionamiento para el motor de gemelos virtuales según la presente invención tal como se muestra en la figura 17;

la figura 19 muestra detalles de funcionamiento adicionales para un módulo de procesamiento de módulo tal como se hace funcionar en el motor de gemelos virtuales;

la figura 20 muestra un diagrama esquemático adicional que ilustra el funcionamiento del motor de gemelos virtuales según la presente invención,

la figura 21 muestra un diagrama de flujo de funcionamiento de detalle adicional para el módulo de procesamiento de flujo de datos mostrado en las figuras 17 y 20, respectivamente;

la figura 22 muestra un ejemplo de un enfoque de caja de semillas para una evaluación de vectores de estado y vectores de condición en el módulo de procesamiento de flujo de datos según la presente invención tal como se muestra en la figura 17;

la figura 23 muestra un ejemplo adicional del enfoque de caja de semillas para la evaluación de vectores de estado y vectores de condición en el módulo de procesamiento de flujo de datos según la presente invención tal como se muestra en la figura 17; y

la figura 24 muestra un ejemplo de actualización de una caja de semillas tal como se muestra en las figuras 22 y 23 tras la ejecución de una transición de estado con respecto a un objeto gemelo.

Descripción detallada de la invención

A continuación, se explicará en detalle la presente invención con referencia a los dibujos. En este caso, debe entenderse que tal explicación se refiere únicamente a ejemplos de la presente invención y no es vinculante en cuanto al alcance de la presente invención, que se define por las reivindicaciones Los ejemplos/aspectos/realizaciones que no son según la invención tal como se define por las reivindicaciones se presentan únicamente con propósitos de ilustración.

En la medida en que se hace referencia a componentes estructurales o de procedimiento específicos, esto debe considerarse como un ejemplo de la funcionalidad subyacente, en la que los componentes estructurales o de procedimiento pueden intercambiarse claramente siempre que se logre la misma funcionalidad. Por tanto, la presente invención se explicará usando terminología genérica para evitar la restricción del alcance de protección.

Además, la funcionalidad del control de procedimiento que usa gemelos digitales según la presente invención es múltiple. No sólo soporta un modelado flexible de sistemas distribuidos de eventos discretos sino también un control de procedimiento en tiempo real de los sistemas distribuidos de eventos discretos.

Generalmente, un sistema distribuido de eventos discretos que va a considerarse dentro del contexto global de la presente invención se divide en una pluralidad de subsistemas en los que cada subsistema individual funciona de manera autónoma y no siempre tiene una imagen completa de los estados de funcionamiento globales en el sistema distribuido de eventos discretos.

Además, un sistema distribuido de eventos discretos puede clasificarse, por ejemplo, como un sistema continuo, un sistema determinista de sistema discreto, sistema estocástico, sistema dinámico, sistema de valor real, sistema causal, sistema lineal, sistema no lineal, sistema invariable en el tiempo, sistema variable en el tiempo, sistema estable, sistema con estabilidad BIBO, sistema libre de pérdidas, etc., o cualquier combinación adecuada de los mismos.

Además, el término procedimiento en un subsistema del sistema distribuido de eventos discretos describe una secuencia de eventos con una relación de tiempo causal mutuamente fijada. Esto implica la capacidad de distinción causal de eventos dentro de subsistemas del sistema distribuido de eventos discretos: dos eventos locales diferentes en un subsistema no deben producirse nunca en el mismo punto de tiempo.

A la vista de lo anterior, los eventos se clasifican en tres clases:

Una primera clase se refiere a eventos internos que se producen dentro de un subsistema del sistema distribuido de eventos discretos sin transmisión externa de mensajes.

Una segunda clase se refiere a eventos de envío que se producen dentro de un subsistema que desencadenan el envío de mensajes desde el subsistema hasta subsistemas diferentes del sistema distribuido de eventos discretos. Desde un punto de vista global, preceden a eventos de recepción correspondientes en diferentes subsistemas a los que se dirigen los mensajes.

Una tercera clase se refiere a eventos de recepción que se producen dentro de subsistemas tras la recepción de mensajes a partir de diferentes subsistemas del sistema distribuido de eventos discretos. Desde un punto de vista global, siguen a eventos de envío correspondientes en diferentes subsistemas desde los que se envían los mensajes.

Generalmente, el modelado de sistemas distribuidos de eventos discretos se basa en un compromiso entre la gestión de una tarea predeterminada y esfuerzos relacionados frente a la complejidad implicada de todo el modelo. En particular, la manera de configurar sistemas parciales particulares para el modelo y la conexión relacionada contribuye a la complejidad de modelo.

En este caso, en el estado de la técnica, un enfoque típico es una estructuración jerárquica y descomposición para una tarea de modelado específica. Generalmente, en el estado de la técnica, los subsistemas se modelan de una manera de composición, es decir con un enfoque de cualquiera a cualquiera. Aunque la estructuración de composición del modelo aumenta la adaptabilidad del modelo según nuevos objetivos de modelado, aumenta la complejidad del modelo, en particular debido al alto número de conexiones de señalización entre los subsistemas.

A diferencia de la anterior, el enfoque de modelado según la presente invención soporta modelado y control de procedimiento de un sistema distribuido de eventos discretos alineado para una modificación dinámica del modelo de sistema global en tiempo real.

Tal como se mostrará a continuación, hasta ahora la presente invención usa diferentes niveles de abstracción y modelado del sistema distribuido de eventos discretos, es decir un metanivel, un nivel de uso, y un nivel de gemelo que permiten una división del sistema distribuido de eventos discretos global en sistemas parciales que entonces se especifican adicionalmente desde arriba hacia abajo según una jerarquía virtual o especificaciones.

La figura 3 muestra ejemplos de modelado en una capa de metamodelo usando metaobjetos como tipos de metamodelos de subsistemas.

Tal como se muestra en la figura 3, los metamodelos usan funciones de transformación o, de manera equivalente, una variedad de expresiones, semántica que describe metaestados, y modelos de espacio de estados que describen un comportamiento lógico.

Generalmente, el metamodelo destaca propiedades de un modelo real. Por tanto, el metamodelado es una construcción de conceptos dentro de un determinado dominio de aplicación. El metamodelo implica normalmente estudiar relaciones de entradas y salidas y el ajuste de metamodelos para representar un comportamiento dado.

Además, la relación entre el nivel de gemelo digital y el nivel de metamodelo es de tipo instancia de relación. Además, la relación entre el nivel de uso y el metamodelo es de tipo de instancia de relación.

Tal como se muestra en la figura 3, el metamodelado implica la abstracción a partir de instanciaciones del mundo real específicas basándose en metaobjetos. Para el ejemplo particular mostrado en la figura 3, los metaobjetos son un metaobjeto transportador y un metaobjeto sensor. Entonces, para el metaobjeto transportador, la funcionalidad relacionada puede describirse usando un diagrama de estados, por ejemplo que representa el estado activado/desactivado del motor de transportador.

Tal como se muestra en la figura 3, también puede abstraerse un sensor a partir del mundo real y describirse de nuevo mediante un diagrama de estados relacionado que caracteriza estados típicos del sensor en general, es decir un estado en el que se completa una medición y un estado en el que se analiza una señal y una transición de estado relacionada para analizar la señal en una medición doble, respectivamente.

Tal como se muestra en la figura 3, aún otro aspecto importante de la presente invención es una metarrepresentación de estados en diferentes niveles de abstracción. Tal como se muestra en la figura 3, una representación de estados abstraídos de este tipo para el metaobjeto transportador será, por ejemplo, una señal de metaestado analizada, una especificación de una característica objetivo con respecto a una velocidad objetivo, y una representación específica de una característica real que muestra la velocidad real del transportador durante el tiempo de ejecución.

Tal como se muestra en la figura 3, de manera similar, la metarrepresentación de estados para el metaobjeto sensor puede ser de nuevo una característica de clasificación que es señal analizada, una característica objetivo para especificar que se facilita el estado de resultado de detección, y como característica real, una ID de objeto que es ID de objeto igual a ID3456.

La figura 4 muestra ejemplos de modelado en una capa de modelo de uso o, de manera equivalente, capa de plantilla para definir agrupaciones de metamodelos de subsistemas.

En este caso, el objetivo es definir objetos de uso como agrupación de objetos de gemelo digital de tal manera que todos los objetos de gemelo digital dentro de una agrupación comparten un conjunto común de características descriptivas, un subconjunto común de eventos globalmente válidos, un conjunto común de estados y un conjunto común de modelos de estados.

En este caso, las agrupaciones de metamodelos de subsistemas usan una variedad de expresiones comunes, semántica y comportamiento de metamodelos de subsistemas así como un conjunto común de metadatos, características y propiedades.

Tal como se muestra en la figura 4, el modelado a nivel de uso difiere con respecto al modelado a metanivel en que es más específico con respecto al uso real de transportadores y sensores descritos de una manera abstracta a metanivel.

Por tanto, tal como se muestra en la figura 4, el modelo de uso de la línea de transportador menciona diferentes usos del transportador, es decir de un primer a un tercer uso. De manera similar, se describen cuatro usos de un sensor. Además, en la capa de uso, también se introducen los denominados vectores de estado externos que implican una operación tras la indicación de un determinado estado en el sistema global, por ejemplo la función de uso de transportador uno como activada cuando el estado del uso de sensor uno es de tal manera que indica que se analiza una señal.

Tal como se muestra en la figura 4, aún otro aspecto de modelado a nivel de uso según la presente invención que va a describirse en más detalle a continuación es la configuración de agrupaciones. En este caso, los usos identificados de diferentes componentes de sistema tales como transportador y sensores se resumen en denominadas agrupaciones de usos de tal manera que se asignan diferentes tipos de metaobjeto a una agrupación relacionada, respectivamente. Entonces, el intercambio de información entre agrupaciones de uso se logra en los denominados metaflujos.

Tal como se muestra en la figura 4, cada agrupación de uso envía señales externas como metaflujo, es decir la agrupación de uso transportador envía señales externas en el flujo transportador mientras que la agrupación de uso sensor establece señales externas en el flujo sensor. Además, cada agrupación de uso escucha para detectar señales externas a partir de al menos un metaflujo que es diferente de su propio metaflujo de envío, por ejemplo la agrupación de uso transportador escucha para detectar señales externas de la agrupación de uso sensor en el flujo sensor. De lo contrario, dentro de la agrupación de uso, también pueden intercambiarse señales internas.

El modelado a nivel de uso y la configuración de agrupaciones de uso y metaflujo relacionado entre las mismas según la presente invención tiene una importancia significativa para reducir la complejidad durante el control de procedimiento en tiempo real del sistema distribuido de eventos discretos global. El motivo de esto es que, a diferencia de una topología de enrutamiento de cualquiera a cualquiera, el número de componentes que van a abordarse viene dado por el número de agrupaciones, no por el número de componentes reales del sistema.

La figura 5 muestra ejemplos de modelado en una capa de gemelo para describir subsistemas como instancias de agrupaciones de metamodelos de subsistemas.

En este caso, cada instancia mapea el comportamiento lógico de un subsistema. A continuación, las instancias a nivel de gemelo digital se denominarán de manera equivalente gemelos digitales.

Por tanto, un tercer nivel de abstracción en el modelado de sistemas distribuidos de eventos discretos según la presente invención es un nivel de abstracción denominado nivel de gemelo digital. En este caso, el objetivo es definir objetos de gemelo digital de objetos de procedimiento del mundo real que modelan dispositivos, datos, mensajes, especificaciones de tarea, etc., del mundo real.

El resultado del modelado de capa de gemelo se muestra como ejemplo en la figura 5. En este caso, se supone que se configuran dos líneas de transportador.

Tal como se muestra en la figura 5, para el transportador hay tres situaciones de uso, no obstante, seis instanciaciones de gemelo de tal manera que del gemelo 1 al gemelo 3 del transportador configuran la línea 1 de transportador, mientras que del gemelo 4 al gemelo 6 del transportador configuran la línea 2 de transportador.

Además, para el uso de sensor, se especifican cuatro usos con respecto a la línea 1 y 2 de transportador, respectivamente. Entonces, para cada uso de sensor hay dos gemelos t1, t4, además t2, t5, además gemelo 3, gemelo 7, y finalmente gemelo 4, gemelo 8.

Tal como puede observarse a partir de la figura 5, sólo en la determinación del modelado en la capa de gemelo, se genera un modelo completo del sistema distribuido de eventos discretos.

La figura 6 muestra un resultado de modelado a un metanivel tal como se muestra en la figura 3 en combinación con el modelado a nivel de uso tal como se muestra en la figura 4 y en combinación con el modelado en una capa de gemelo tal como se muestra en la figura 5.

Tal como se muestra en la figura 6, el sistema completo mostrado en la figura 5 se modela a lo largo de tres capas de abstracción, es decir la representación de objeto de metamodelo usada para la configuración de agrupaciones tales como la agrupación transportador y la agrupación sensor, después, el modelado a nivel de uso para clasificar el uso de componentes de sistema caracterizado a metanivel según el tipo de metamodelado, y después seguido por la instanciación de las diferentes situaciones de uso para cada agrupación a nivel de gemelo.

Tal como se muestra en la figura 6, la interoperación entre diferentes agrupaciones de sistemas se logra entonces basándose en metaflujos en relación con las diferentes agrupaciones para reducir significativamente la carga global para el intercambio de información durante el control de procedimiento en tiempo real del sistema distribuido de eventos discretos global, que va a describirse a nivel más detallado a continuación.

La figura 7 muestra un contorno general de un modelo de espacio de estados usado según la presente invención para el modelado de sistemas distribuidos de eventos discretos.

Generalmente, según un modelo de estados, un sistema se describe mediante un vector de entrada $x = (x1, ..., xk)T \in X$, un vector de salida $y = (y1, ..., yn)T \in Y$, y un operador de sistema S que mapea el vector de entrada x al vector de salida y según $y = S(x)$.

En más detalle, el modelo matemático usa variables de estado para modelar estados de sistema que se resumen en un vector de variable de estado $z = (z1, ..., zm)T \in Z$.

La composición del vector de entrada x, el vector de variable de estado z, y el vector de salida y puede representarse usando al menos dos funciones:

Una primera función es una función de transición de estado $\Phi: Z \times X \rightarrow Z$ que determina una transición a un(os) estado(s) objetivo $z' \in Z$ en función de un(os) estado(s) inicial(es) $z \in Z$ y un vector de entrada $x \in X$ al producirse un evento de sistema para describir el comportamiento dinámico del sistema.

La segunda función es una función de salida $\Omega: Z \times X \rightarrow Y$ que determina un vector de salida $y \in Y$ en función del vector de variable de estado $z \in Z$ y el vector de entrada $x \in X$.

Opcionalmente, para algunos tipos de sistemas de eventos discretos puede definirse una tercera función que es una función de tiempo de retención $\tau: Z - R+$ que define el tiempo de retención con respecto a un vector de variable de estado $z \in Z$.

Además, debe observarse que, con respecto a la existencia y al tipo de vectores de entrada x, un vector de entrada vacío x corresponde a un sistema autónomo o, de manera equivalente, un generador de datos. Además, un vector de entrada vacío x en combinación con un vector de salida vacío y se refiere a sistemas que pueden describirse como un modelo de cadena de Markov.

Tal como se muestra en la figura 7, una característica según la presente invención es el tipo de representación de un estado en el modelo de estados. Según la presente invención, se sugiere tener diferentes tipos de abstracción para la representación de estados según metanivel, nivel objetivo, y nivel real, tal como se mostró anteriormente a modo de ejemplo en la figura 3.

Estos diferentes tipos de abstracción para la representación de estados según la presente invención permiten una reducción significativa de la complejidad en la manipulación del control de procedimiento del sistema distribuido de eventos discretos. El motivo para esto es que aunque, por ejemplo, el objetivo de un subsistema usado para especificar un comportamiento de control puede estar en el dominio de tiempo real y, por tanto, ser un número infinito de valores. No obstante, la abstracción de una manera de un intervalo real de este tipo para dar tipos de metanivel de, por ejemplo, velocidad baja, velocidad media, velocidad alta, conducirá entonces a una reducción de la complejidad en la manipulación, por ejemplo, de una línea de transportador tal como se describió anteriormente.

La figura 8 muestra una representación detallada adicional del modelo de espacio de estados mostrado en la figura 7.

Tal como se muestra en la figura 8, puede descomponerse una función de transición de estado Φ al menos en una transición de estado externa en Φex para describir transiciones de estados debidas a eventos externos al sistema, respectivamente.

En más detalle, tal como se muestra en la figura 8, según la presente invención se considera un estado en el que una transición de estado en un primer subsistema se ve influida por estados de subsistemas adicionales que no son idénticos al primer subsistema.

Tal como se muestra en la figura 8, puede considerarse un modelo de estados para un subsistema específico y una transición relacionada ΦEX desencadenada por un vector de estado externo que resume una identificación de subsistemas externos y estados relacionados, por ejemplo, para SS 1, st 1, ..., SS n, st n. Suponiendo que cada uno de estos subsistemas está en el estado especificado, entonces, según la presente invención, se considera una transición de estado externa dentro del subsistema correspondiente al vector de estado externo.

El concepto del vector de estado externo implica un factor desencadenante para la transición de estado dentro de un subsistema específico y soporta la paralelización en el procedimiento de modelado de sistema con respecto a la interacción entre subsistemas del sistema distribuido de eventos discretos. Los vectores de estado externo pueden configurarse de una manera flexible para aumentar o reducir la granularidad en el nivel de condiciones que fijan una transición de estado en un subsistema específico.

La figura 9 muestra una representación detallada adicional del modelo de espacio de estados mostrado en la figura 7.

Tal como se muestra en la figura 9, una función de transición de estado Φ puede descomponerse adicionalmente al menos en una transición de estado externa interna en Φin para describir transiciones de estados debidas a eventos internos de sistema.

Tal como se muestra en la figura 9, la transición de estado interna también se desencadena teniendo en cuenta la representación de estados de al menos un estado dentro del sistema. Suponiendo que prevalece una representación de estados que desencadena una transición de estado, entonces puede actualizarse la representación global de modelo de espacio de estados del subsistema. Debe observarse que el cambio de un estado interno dentro de un subsistema específico puede constituir un evento externo para un subsistema remoto y desencadenar una transición de estado en el subsistema remoto tal como se explicó anteriormente con respecto a la figura 4.

A la vista de lo anterior, a continuación se explicará un modelado de unos sistemas distribuidos de eventos discretos en general como base para entender los principios subyacentes de la presente invención.

En más detalle, el modelo de estados de un sistema distribuido de eventos discretos se define mediante una tupla de 7 elementos $E = <X, Z, Y, \Phi in, \Phi ex, \Omega,$ en la que

X es el conjunto finito de vectores de entrada $x \in X$;

Z es el conjunto finito de vectores de variables de estados $z \in Z$;

Y es el conjunto finito de vectores de salida $y \in Y$;

$\Phi in: Z \rightarrow Z$ es la función de transición de estado para eventos internos;

$\Phi ex: Z \times X \rightarrow Z$ es la función de transición de estado para eventos externos;

$\Omega: Z \times X \rightarrow Y$ es la función de salida; y

$\tau: Z \rightarrow R+$ es la función de tiempo de retención.

Según la presente invención, los vectores de entrada $x \in X$ para sistemas distribuidos de eventos discretos se denominan eventos externos. Tras producirse un evento externo, el estado de sistema $z \in Z$ puede cambiar a $z' \in Z$ según

$$z'(t) = \Phi ex\ (x(t),\ z0(t)).$$

En este caso, $z0(t) \in Z$ es el estado inicial del sistema distribuido de eventos discretos o, de manera resumida, el estado inicial y z'(t) es el estado objetivo del sistema distribuido de eventos discretos o, de manera resumida, el estado objetivo.

Además, tal como se explicó anteriormente de manera resumida con respecto a la figura 9, se generan transiciones de estados sin entrada debidas a eventos internos que pueden producirse a lo largo del tiempo. El sistema distribuido de eventos discretos permanece en el estado z a lo largo de un tiempo de retención $\tau(z)$. Al caducar el tiempo de retención $\tau(z)$, se genera un nuevo estado que se determina mediante la función de transición de estado interna según

$$\mathbf{z}'(t+\tau(\mathbf{z})) = \mathbf{\Phi in}\ (\mathbf{z}(t)).$$

Además, tras producirse una transición de estado, también se actualiza el vector de salida según

$$\mathbf{y}(t) = \mathbf{\Omega}(\mathbf{z}(t), \mathbf{x}(t))$$

donde z(t) es de nuevo el estado inicial. Debe observarse que x(t) puede ser el vector nulo cuando no prevalece ninguna entrada o, de manera equivalente, ningún evento externo.

Tal como se mencionó anteriormente, la función de tiempo de retención τ determina la duración de cada estado parcial $zi \in z = (z1, ..., zm)T$. Cualquier cambio en un estado parcial implica un cambio del vector de variable de estado $z \in Z$.

Suponiendo que ti es el tiempo inicial del estado parcial zi y que $\tau1, ..., im$ son los tiempos de retención para los estados parciales z1, ..., zm tal como se determinan en el tiempo inicial ti, entonces la función de tiempo de retención para el estado parcial zi en el tiempo ti puede determinarse según

$$\tau(zi) = \min\{\tau1, \ldots, \tau m\}.$$

Generalmente, según la presente invención se supone que los eventos en el sistema distribuido de eventos discretos se producen en puntos de tiempo aleatorios de modo que las transiciones de estados pueden calcularse usando modelos de estados estocásticos. Entonces, un caso especial de un sistema distribuido de eventos discretos puede ser el modelo de estados discretos en el tiempo determinista que tiene un tiempo de muestra tA en el que $\tau(zi) = tA$ para $i \in (1, ..., m)$ y $ti = tj$ para todos los $i, j \in (1, ..., m)$ con $i \neq j$.

Basándose en las explicaciones facilitadas anteriormente, a continuación se describirán principios de derivar un metamodelo para sistemas distribuidos de eventos discretos según la presente invención.

Tal como se mostró anteriormente a modo de ejemplo con respecto a la figura 3, a nivel de metamodelado hay tipos específicos de metamodelos denominados a continuación tipos de metaobjeto que sirven para modelar tipos específicos de subsistemas del sistema distribuido de eventos discretos.

En más detalle, según la presente invención un tipo de metaobjeto especifica un metamodelo de representaciones de estados y semántica relacionada. Los tipos de metaobjeto también especifican funciones de transformación o, de manera equivalente, una variedad de expresiones y un comportamiento lógico del subsistema que está modelándose mediante referencia a al menos un modelo de espacio de estados correspondiente.

A continuación se supone, sin pérdida de generalización, que se modela un subsistema SS con un metaobjeto MO que tiene un tipo de metaobjeto • específico pero aún no fijado.

Por tanto, debe entenderse • como marcador para un tipo de metaobjeto o, de manera equivalente, como que caracteriza una plantilla de tipo de metamodelo.

Entonces, la instanciación de un tipo de metaobjeto genérico para un uso específico del mismo se logra mediante parametrización del marcador • en el que diferentes situaciones de uso se gestionan mediante múltiples parametrizaciones.

Según la presente invención, un primer elemento de un metamodelo de un tipo de metaobjeto MO• es un objeto de representación MO•R que representa un conjunto $D\bullet = \{d1\bullet, ..., do\bullet\}$ de características descriptivas del tipo de metaobjeto MO•.

Además, según la presente invención, en el metamodelo el al menos un tipo de metaobjeto MO• puede percibirse en diferentes vistas en perspectiva sobre el tipo de metaobjeto MO•, describiéndose un conjunto $P\bullet = \{p1\bullet, ..., pv\bullet\}$ de perspectivas sobre un tipo de metaobjeto MO•.

Según la presente invención, un segundo elemento es al menos un tipo de metaobjeto MO• para modelar un subsistema del sistema distribuido de eventos discretos, en el que el al menos un tipo de metaobjeto MO• hace referencia a al menos un modelo de estados $\Sigma i\bullet\ i = 1, ...,s$ en relación con el subsistema.

En más detalle, según la presente invención, para cada tipo de metaobjeto MO• el metamodelo especifica al menos un modelo de estados Σi• i = 1, ...,s que representa un sistema de eventos discretos según Σi• = <Xi•, Zi•, Yi•, Φini•, Φexi•, Ωi•, τi•> de tal manera que el modelo de estados global del tipo de metamodelo MO• es Σ• = Ui Σi•.

El motivo para la referencia a un modelo de estados Σi• mediante un tipo de metaobjeto MO• es que un mismo tipo de metaobjeto MO• puede percibirse según diferentes criterios que forman, cada uno, una vista en perspectiva sobre el tipo de metaobjeto MO• y el subsistema subyacente que va a modelarse. A partir de esto, resulta significativo disponer de un modelo de estados Σi• por cada vista en perspectiva usada por el tipo de metaobjeto MO•. A continuación, el conjunto de perspectivas sobre un tipo de metaobjeto MO• se describe como P• = {p1•, ..., pv•}.

Además, para cada modelo de estados Σi• existe un vector de variable de estado zi• = (zi•1, ..., zi•z)T ∈ Zi•. Según la presente invención, cada k-ésimo estado en zi•, es decir el estado zi•k en cada modelo de estados Σi•, puede describirse usando diferentes niveles de abstracción de datos tales como metadatos, valores discretos o valores continuos para valores objetivo, y valores discretos o continuos para valores reales.

Por tanto, según la presente invención, cada estado zi•k tiene un metamodelo que es un conjunto de estados parciales zi•k,p descritos según

al menos un estado zi•k en el al menos un modelo de estados Σi• i = 1, ..., s tiene un metamodelo que es un conjunto de estados parciales zi•k,p usando diferentes niveles de abstracción de datos según

un conjunto de metacaracterísticas MFi•k,1 = {fc1, ..., fcc}i•k,1 que caracterizan un estado en un dominio de metadescripción;

un conjunto de características objetivo TFi•k,2 = {ft1, ..., ftt}i•k,2 que tienen un intervalo discreto o continuo de valores que caracterizan un objetivo para un estado considerado; y

un conjunto AFi•k,3 = {fa1, ..., faa}i•k,3 de características reales AFi•k,3 = {fa1, ..., faa}i•k,3 que tienen un intervalo discreto o continuo de valores que representan una constelación real de un estado considerado.

Debe observarse que las metacaracterísticas de uso que caracterizan un estado en un dominio de metadescripción según la presente invención logran una abstracción en comparación con la representación de espacio de estados convencional. La abstracción según la presente invención permite una metarrepresentación de espacio de estados. En este caso, cada metarrepresentación de espacio de estados puede resumir un número infinito de representaciones de espacio de estados en una única metarrepresentación de espacio de estados y, por tanto, puede reducir significativamente la complejidad de modelado.

Además, en el metamodelo una transición de estado en el i-ésimo modelo de estados Σi• del tipo de metaobjeto MO• desde un primer estado zi•a hasta un segundo estado zi•t se logra mediante un procedimiento descrito mediante un modelo funcional Φi•at: Zi• × Xi• → Zi•. Por tanto, cualquier transición de estado se asigna exactamente a un modelo de estados Σi• y hay exactamente una transición de estado desde zi•a hasta zi•t.

En este caso, según la presente invención, el modelo funcional representa un modelo de flujo de datos según metacaracterísticas, característica objetivo, y/o características reales. El modelo de flujo de datos describe un flujo de control que funciona en metadatos y un flujo de datos que funciona en datos que reflejan características objetivo y/o características reales.

Debe observarse que el modelo funcional Φi•at se aplica de manera general a transiciones de estados debidas a eventos externos y/o a eventos internos que son relevantes para un tipo de metaobjeto MO• específico según Φex,i•at y Φin,i•at, respectivamente.

En este caso, el modelo funcional Φex,i•at representa un metamodelo de una transición de estado desencadenada por al menos un evento externo relevante para un tipo de metaobjeto MO• específico. Además, describe un modelo de flujo de datos según diferentes niveles de abstracción de datos, es decir con respecto a metacaracterísticas, características objetivo, y/o características reales.

Según la presente invención, en el sentido más general, el modelo de flujo de datos describe un flujo de control que funciona en metadatos y un flujo de datos que funciona en datos que reflejan características objetivo o reales. La incorporación de flujo de datos en el modelo de flujo de datos permite una especificación de condiciones para transiciones de estados.

Además, según la presente invención, una primera entrada en el modelo funcional Φex,i•at es al menos un vector de estados externos o, de manera equivalente, vector de estado externo Θex,i•at,l = ((MOj, j ≠ •, zj), ..., (MOk, k ≠ •, zk))lT, donde l es un índice para el l-ésimo vector de estado externo y cada vector de estado externo resume respectivamente tuplas de subsistemas externos a un subsistema que está modelándose usando el tipo de metaobjeto MO• y estados correspondientes en los que se indica que prevalecen los subsistemas externos.

Por tanto, cuando un determinado conjunto de subsistemas que son externos a un tipo de subsistema que está modelándose usando el tipo de metamodelo MO• y en consideración están en estados específicos, respectivamente, esto puede desencadenar una transición de estado desde zi•a hasta zi•t en uno de los modelos de estados Σi• del tipo de metaobjeto MO• usado para modelar un subsistema en consideración. La identificación de los modelos de estados Σi• relevantes y del tipo de transiciones de estados se explicará en más detalle a continuación.

Además, el motivo para la consideración de más de un vector de estado externo Θex,i•at,l, 1 > 1, es que, aunque prevalezcan diferentes subsistemas en diferentes estados, respectivamente, esto puede desencadenar no obstante una misma transición de estado desde zi•a hasta zi•t en el modelo de estados de subsistema Σi• de un tipo de metaobjeto MO• en consideración.

Además, Πex,i•a = Ul Θex,i•at,l es el conjunto de todos los vectores de estado externo que pueden desencadenar una transición de estado con respecto al estado inicial zi•a en el modelo de estados Σi• de un tipo de metaobjeto MO•.

Además, según la presente invención, una segunda entrada para el modelo funcional se representa como un vector Δex,i•a que resume un conjunto de condiciones que deben cumplirse antes de desencadenar una transición de estado con respecto al estado inicial zi•a en el modelo de estados Σi• de un tipo de metaobjeto MO• debida a un evento externo. Normalmente, tales condiciones se especificarán basándose en datos que reflejan características objetivo y características reales.

Además, para concluir con el metamodelado de tipos de metaobjeto MO• según la presente invención, se introduce un conjunto de contextos C = {con_ID1, ..., con_IDk}.

En este caso, cada contexto modela una posible relación entre dos objetos gemelos que representan instanciaciones digitales de tipos de metaobjeto MO•, o, dicho de otro modo, una posible relación entre instanciaciones de componentes del sistema distribuido de eventos discretos a nivel de gemelo tal como se explicará en más detalle a continuación.

Generalmente, se definen contextos en relación con modelos funcionales y son válidos para todos los tipos de metaobjeto MO• que hacen referencia a los modelos funcionales. Además, se asignan contextos de una manera inequívoca a una perspectiva.

A la vista de lo anterior, según la presente invención, el modelo funcional Φex,i•at puede modelarse como un flujo de datos de metamodelo según:

$$\Phi_{ex,i\bullet at}: zi\bullet a \in Zi\bullet \times \Pi_{ex,i\bullet a} \times \Delta_{ex,i\bullet a} \times con_IDa \times MFi\bullet a,1 \times TFi\bullet a,2 \times AFi\bullet a,3 \rightarrow zi\bullet t \in Zi\bullet \times cor_IDt \times MFi\bullet t,1 \times TFi\bullet t,2 \times AFi\bullet t,3$$

Además, según la presente invención, un modelo funcional similar Θin,i•at puede modelarse como un flujo de datos de metamodelo con respecto a una transición de estado interna en el tipo de metaobjeto MO• y su modelo de estados Σi• según:

$$\Phi_{in,i\bullet at}: zi\bullet a \in Zi\bullet \times \Pi_{in,i\bullet a} \times \Delta_{in,i\bullet a} \times con_IDa \times MFi\bullet a,1 \times TFi\bullet a,2 \times AFi\bullet a,3 \rightarrow zi\bullet t \in Zi\bullet \times cor_IDt \times MFi\bullet t,1 \times TFi\bullet t,2 \times AFi\bullet t,3$$

En los modelos funcionales anteriores, pueden identificarse elementos de representación de la siguiente manera:

zi•a ∈ Zi• es el estado de origen seleccionado de un conjunto de todos los estados Zi• del al menos un modelo de estados Σi• i = 1, ...,s) del tipo de metaobjeto MO•;

zi•t ∈ Zi• es el estado objetivo seleccionado de un conjunto de todos los estados Zi• del al menos un modelo de

estados Σi• i = 1, ..., s del tipo de metaobjeto MO•;

Πex,ini•a es el conjunto de los vectores de estado externo o interno que desencadenan la transición de estado desde el estado de origen zi•a hasta el estado objetivo zi•a;

Δex,ini•a es el vector de condiciones que desencadena la transición de estado desde el estado de origen zi•a hasta el estado objetivo zi•t;

con_IDa es identificación de contexto del contexto que prevalece para el subsistema que está modelándose mediante el tipo de metaobjeto MO• antes de la transición de estado desde el estado de origen zi•a hasta el estado objetivo zi•t;

con_IDt es identificación de contexto del contexto que prevalece para el subsistema que está modelándose mediante el tipo de metaobjeto MO• después de la transición de estado desde el estado de origen zi•a hasta el estado objetivo zi•t;

MFi•a,1 × TFi•a,2 × AFi•a,3 es una representación de valores para la metacaracterística, característica objetivo, y característica real en el estado de origen zi•a antes de la transición de estado al estado objetivo zi•t; y

MFi•t,1 × TFi•t,2 × AFi•t,3 es una representación de valores para la metacaracterística, característica objetivo, y característica real en el estado objetivo zi•t tras la transición de estado al estado objetivo zi•t.

En conclusión, según la presente invención, independientemente de cómo se realice una transformación específica, funcionará a dos niveles de abstracciones, es decir un flujo de control que funciona en metadatos y un flujo de datos que logra un vínculo entre nivel de metadatos, condiciones objetivo y condiciones reales, por ejemplo, captadas mediante sensores. Por tanto, la introducción de un flujo de control que funciona en metadatos permite gestionar un número infinito de estados de una manera computacionalmente muy eficiente.

Además, debe observarse que una transición de estado también puede desencadenarse mediante una combinación de al menos un evento externo y al menos un evento interno.

A continuación, se describirán diferentes aspectos de metamodelado implementado por ordenador de un sistema distribuido de eventos discretos y el uso relacionado del metamodelo implementado por ordenador para control de procedimiento en tiempo real del sistema distribuido de eventos discretos.

La figura 10 muestra una visión general para un sistema adaptado para modelar un sistema distribuido de eventos discretos y para ejecutar control de procedimiento en tiempo real de los sistemas distribuidos discretos uniformes según la presente invención.

Tal como se muestra en la figura 10, el sistema 10 según la presente invención comprende un motor 12 de modelado, un motor 14 de servicio, y un motor 16 de gemelos virtuales. El motor 16 de gemelos virtuales forma un homólogo virtual de objetos de procedimiento en el mundo real.

En este caso, los objetos de procedimiento son, por ejemplo, dispositivos de campo que tienen que controlarse durante el funcionamiento del sistema distribuido de eventos discretos. No obstante, según la presente invención, no se impone ninguna restricción sobre el tipo de objeto de procedimiento que también puede estar relacionado con tareas que van a lograrse dentro del sistema distribuido de eventos discretos y, por ejemplo, también modelar diferentes tipos de información, etc.

En cuanto al funcionamiento, el motor 12 de modelado impulsa un modelo a metanivel del sistema distribuido de eventos discretos tal como se explicó anteriormente de manera resumida. Una vez que está disponible un modelo a metanivel de este tipo para el sistema distribuido de eventos discretos, el motor 12 de modelado emite una petición de servicio al motor 14 de servicio.

Adicionalmente en cuanto al funcionamiento, el servicio 14 logra un modelado del sistema distribuido de eventos discretos a nivel de uso tal como se explicó anteriormente de manera resumida y repara adicionalmente una estructura principal de software ejecutable que refleja el resultado del modelado a nivel de uso del sistema distribuido de eventos discretos.

Tal como también se muestra en la figura 10, el motor 12 de modelado y el motor 14 de servicio tienen una interfaz con respecto al motor 16 de gemelos virtuales.

En cuanto al funcionamiento, el motor 14 de servicio reenvía las estructuras principales de software ejecutable en relación con diferentes tipos de agrupaciones de componentes tal como se explicó anteriormente al motor 16 de gemelos virtuales para su preinstalación en el mismo. Además, el motor 14 de servicio reenvía la topología de enrutamiento derivada durante el modelado a nivel de uso del sistema distribuido de eventos discretos al motor 16

de gemelos virtuales para el establecimiento de procedimientos de comunicación relacionados dentro del motor 16 de gemelos virtuales tal como se explicó anteriormente.

Tal como se muestra en la figura 10, en cuanto al funcionamiento, el motor 12 de modelado también puede inyectar componentes de modelado en el motor 16 de gemelos virtuales en tiempo real y de manera dinámica. Esto sirve para modificar el modelo del sistema distribuido de eventos discretos global en tiempo real para aumentar la flexibilidad del método de modelado y control de procedimiento tal como se explicará en más detalle a continuación.

La figura 11 muestra un diagrama de flujo básico de funcionamiento para el sistema adaptado para modelar un sistema distribuido de eventos discretos y para ejecutar control de procedimiento en tiempo real de los sistemas distribuidos discretos uniformes según la presente invención.

Tal como se muestra en la figura 11, inicialmente, en una etapa S10 ejecutada por el motor 12 de modelado, se genera al menos un tipo de metaobjeto correspondiente al modelo de datos estáticos previamente configurado que se explicará a continuación con respecto a la figura 12.

Después, en una etapa S12 ejecutada por el motor 14 de servicio, se agrupan subsistemas del sistema distribuido de eventos discretos según el/los tipo(s) de metaobjeto. Además, el motor 14 de servicio ejecuta una etapa S16 por cada objeto de agrupación para generar agrupaciones secundarias en relación con usos de tipo(s) de metaobjeto. Además, el motor 14 de servicio logra una especificación de una topología de enrutamiento entre diferentes agrupaciones.

Tal como se muestra en la figura 11, entonces sigue una etapa S16 ejecutada por el motor de gemelos virtuales para ejecutar el modelo generado en tiempo real. En este caso, el motor 16 de gemelos virtuales asignará al menos un objeto gemelo por uso(s) de tipo de metaobjeto y también usará diferentes niveles de abstracción con respecto a la representación de datos tal como se explicará en más detalle a continuación.

La figura 12 muestra un diagrama esquemático de un metamodelo implementado por ordenador que se usa por el motor de modelado según la presente invención tal como se muestra en la figura 10.

Tal como se muestra en la figura 12, un metamodelo implementado por ordenador según la presente invención se divide en un metamodelo estático y en un metamodelo dinámico. En este caso, el metamodelo estático comprende una pluralidad de entidades que son respectivamente un repositorio para acumulación de un modelo con capacidad de tiempo real de un sistema distribuido de eventos discretos.

Por otro lado, la parte de metamodelo dinámico del metamodelo implementado por ordenador hará referencia al menos parcialmente a las entidades del metamodelo estático para la transferencia de opción de modelado proporcionada por las diferentes entidades a un metamodelo real de un subsistema del sistema distribuido discreto uniforme.

Tal como se muestra en la figura 12, el metamodelo estático del metamodelo 18 implementado por ordenador para modelar al menos un subsistema de un sistema distribuido de eventos discretos comprende al menos una entidad 20 de metaobjeto que describe un tipo de metaobjeto MO• usado para modelar un subsistema del sistema distribuido de eventos discretos.

Tal como se muestra en la figura 12, la al menos una entidad 20 de metaobjeto hace referencia a una entidad 22 de modelo de espacio de estados que especifica al menos un modelo de estados $\Sigma i\bullet$ con $i \geq 1$ en relación con el subsistema. En la entidad 22 de modelo de espacio de estados el al menos un estado $zi\bullet k$ en el al menos un modelo de estados $\Sigma i\bullet$ se representa mediante un metamodelo que es un conjunto de estados parciales $zi\bullet k,p$ usando diferentes niveles de abstracción de datos según

un conjunto de metacaracterísticas $MFi\bullet k,1 = \{fc1, ..., fcc\}i\bullet k,1$ que caracterizan un estado en un dominio de metadescripción;

un conjunto de características objetivo $TFi\bullet k,2 = \{ft1, ..., ftt\}i\bullet k,2$ que tienen un intervalo discreto o continuo de valores que caracterizan un objetivo para un estado considerado; y

un conjunto $AFi\bullet k,3 = \{fa1, ..., faa\}i\bullet k,3$ de características reales $AFi\bullet k,3 = \{fa1, ..., faa\}i\bullet k,3$ que tienen un intervalo discreto o continuo de valores que representan una constelación real de un estado considerado.

Además, la al menos una entidad 20 de metaobjeto hace referencia a una entidad de representación que especifica al menos un objeto de representación MO•R que representa un conjunto $D\bullet = \{d1\bullet, ..., do\bullet\}$ de características descriptivas de un tipo de metaobjeto (MO•).

Además, el al menos un tipo de metaobjeto MO• puede percibirse en diferentes vistas en perspectiva sobre el tipo de metaobjeto MO• que está describiéndose como un conjunto $P\bullet = \{p1\bullet, ..., pv\bullet\}$ de perspectivas sobre el al menos un

tipo de metaobjeto (MO•).

Tal como se muestra en la figura 12, el metamodelo estático del metamodelo implementado por ordenador comprende una entidad 24 funcional que representa una transición de estado en el al menos un modelo de estados Σi• desde un estado de origen zi•a hasta un estado objetivo zi•t mediante un modelo funcional Φi•at: Zi• × Xi• → Zi• que representa un modelo de flujo de datos según metacaracterísticas, característica objetivo, y/o características reales.

Según la presente invención, la entidad 24 funcional representa el modelo de flujo de datos como un flujo de control que funciona en metadatos y un flujo de datos que funciona en datos que reflejan características objetivo y/o características reales. Preferiblemente, en el que la entidad 24 funcional representa el modelo funcional como un flujo de metadatos Φex,i•at o Φin,i•at tal como se explicó anteriormente, en el que ambos se resumen en Φ(ex,in),i•at a continuación.

Tal como se muestra en la figura 12, el metamodelo estático del metamodelo implementado por ordenador comprende una entidad 26 de vector de entrada que representa una primera entrada en el modelo funcional Φ(ex,in)i•at como al menos un vector de estado Θ(ex,in),i•at,l que resume al menos una tupla de una identificación de un subsistema y un estado en el que prevalece el subsistema identificado.

Además, la entidad 26 de vector de entrada representa una segunda entrada en el modelo funcional Φ(ex,in),i•at como vector que resume al menos una condición Δ(ex,in),i•at que debe cumplirse antes de una transición de estado desde un estado de origen zi•a hasta un estado objetivo zi•t.

Además, la entidad 26 de vector de entrada representa la al menos una condición como datos que reflejan al menos una característica objetivo y/o al menos una característica real.

Tal como se muestra en la figura 12, el metamodelo estático del metamodelo implementado por ordenador comprende una entidad 28 de contexto que representa al menos una posible relación entre dos objetos gemelos que representan instanciaciones digitales de tipos de metaobjeto MO• como un conjunto de contextos C = {con_ID1, ..., con_IDk} en el sentido anteriormente explicado de manera resumida.

Además de lo anterior, aparte de metamodelo estático, el metamodelo 18 implementado por ordenador según la presente invención comprende una parte de metamodelo dinámico que usa las opciones de modelado disponibles a partir de las entidades de los metamodelos estáticos para obtener un metamodelo específico de un subsistema del sistema distribuido de eventos discretos.

Tal como se muestra en la figura 12, el metamodelo dinámico del metamodelo 18 implementado por ordenador para modelar al menos un subsistema de un sistema distribuido de eventos discretos comprende un módulo 30 de uso de tipo de metaobjeto adaptado para seleccionar un tipo de metaobjeto MO• para el modelado del al menos un subsistema del sistema distribuido de eventos discretos. Tal como se explicó anteriormente de manera resumida, el tipo de metaobjeto MO• hace referencia a al menos un modelo de estados Σi• i ∈ {1, ...,s} para el modelado del al menos un subsistema del sistema distribuido de eventos discretos.

Tal como se muestra en la figura 12, el metamodelo dinámico del metamodelo 18 implementado por ordenador comprende un módulo 32 de uso de modelo de espacio de estados adaptado para modelar el al menos un modelo de estados Σi• i ∈ {1, ...,s} al que hace referencia el tipo de metaobjeto MO• seleccionado.

Tal como se muestra en la figura 12, el modelado del al menos un modelo de estados Σi• i ∈ {1, ...,s} se logra mediante referencia a las opciones de modelado proporcionadas por la entidad 22 de modelo de espacio de estados en general que después se transfieren a usos de modelo a metanivel por el módulo 32 de uso de modelo de espacio de estados.

Además, el módulo 32 de uso de modelo de espacio de estados está adaptado para describir al menos un estado en el al menos un modelo de estados seleccionado Σi• i = 1, ..., s con un metamodelo que es un conjunto de estados parciales zi•k,p que usa diferentes niveles de abstracción de datos tal como se explicó anteriormente de manera resumida. Tal como se explicó anteriormente de manera resumida, diferentes modelos de estados pueden tener diferentes perspectivas asociadas.

Tal como se muestra en la figura 12, el metamodelo dinámico del metamodelo 18 implementado por ordenador comprende un módulo de uso de representación adaptado para modelar un objeto de representación MO•R que representa un conjunto D• = {d1•, ..., do•} de o características descriptivas del al menos un subsistema modelado. En este caso, el modelo de uso de representación también hace referencia a la entidad de representación del metamodelo estático para instanciar al menos una de las características descriptivas descritas de manera general en la entidad de representación con respecto a una tarea de modelado específica.

Tal como se muestra en la figura 12, el metamodelo dinámico del metamodelo 18 implementado por ordenador

comprende un módulo 34 de uso de función adaptado para modelar una transición de estado en el al menos un modelo de estados Σi• desde un estado de origen zi•a hasta un estado objetivo zi•t mediante el modelo funcional que representa un modelo de flujo de datos según metacaracterísticas, característica objetivo, y/o características reales tal como se describió anteriormente.

Tal como se muestra en la figura 12, el modelado del modelo funcional se logra mediante referencia a la entidad 24 de función del metamodelo estático que proporciona opciones generales para funciones de modelo. Las opciones generales se transfieren entonces a un modelo de función más específico mediante indicación de transformaciones de función aplicables e indicación de argumentos aplicables para realizar el modelo funcional Φex,i•at y/o el modelo funcional Φin,i•at tal como se explicó anteriormente.

Además, al menos una posible relación entre al menos dos objetos gemelos que representan instanciaciones digitales de tipos de metaobjeto puede modelarse mediante al menos un contexto tal como se explicó anteriormente.

Tal como se muestra en la figura 12, el metamodelo dinámico del metamodelo 18 implementado por ordenador comprende un módulo 36 de uso de vector de entrada adaptado para modelar una primera entrada en el modelo funcional como vector de estado externo o vector de estado interno tal como se explicó anteriormente de manera resumida. Hasta ahora, se hace referencia a la entidad 26 de vector de entrada del modelo de datos estáticos del metamodelo implementado por ordenador que proporciona plantillas de vector de entrada y parámetros relacionados. Se aplica un enfoque similar al modelado del segundo vector que especifica condiciones para al menos una transición de estado.

Tal como se muestra en la figura 12, la parte de metamodelo dinámico del metamodelo 18 implementado por ordenador según la presente invención también comprende un módulo 38 de uso de gemelo/gemelo.

Proporcionar un módulo 38 de uso de gemelo de este tipo sirve para preparar ya un modelado de componentes del sistema distribuido de eventos discretos que se sabe que forman parte en sí mismos del sistema distribuido de eventos discretos.

Por tanto, los objetos gemelos relacionados pueden clasificarse como previamente conocidos o estáticos, a diferencia del objeto gemelo digital que va a asignarse en tiempo real durante el modelado a nivel de gemelo del sistema distribuido de eventos discretos que va a describirse a continuación.

La figura 13 muestra un diagrama de flujo de funcionamiento para el motor de modelado según la presente invención tal como se muestra en la figura 10.

Generalmente, el motor 12 de modelado según la presente invención ejecuta el metamodelo implementado por ordenador explicado anteriormente con respecto a la figura 12.

Tal como se muestra en la figura 13, el motor 12 de modelado ejecuta una etapa S18, ejecutando el módulo 30 de uso de tipo de metaobjeto, para seleccionar un tipo de metaobjeto MO• para el modelado del sistema distribuido de eventos discretos. El tipo de metaobjeto MO• hace referencia a al menos un modelo de estados Σi• i ∈ {1, ...,s} para el modelado del al menos un subsistema del sistema distribuido de eventos discretos.

Tal como se muestra en la figura 13, el motor 12 de modelado ejecuta una etapa S20, ejecutando el módulo 32 de uso de modelo de espacio de estados, para seleccionar al menos un modelo de estados Σi• i ∈ {1, ..., s} al que hace referencia el tipo de metaobjeto MO• para su uso real durante el modelado del al menos un subsistema. En la etapa S20 el al menos un estado zi•k en el al menos un modelo de estados seleccionado Σi• i ∈ {1, ...,s} se describe con un metamodelo tal como se explicó anteriormente de manera resumida.

Tal como se muestra en la figura 13, el motor 12 de modelado ejecuta una etapa S22, ejecutando un módulo de uso relacionado, para modelar un objeto de representación MO•R y seleccionar al menos una vista en perspectiva, etapa S22 que es opcional.

Tal como se muestra en la figura 13, el motor 12 de modelado ejecuta una etapa S24, ejecutando el módulo 32 de uso de función, para modelar una transición de estado en el al menos un modelo de estados Σi• desde un estado de origen zi•a hasta un estado objetivo zi•t mediante un modelo funcional que representa un modelo de flujo de datos según metacaracterísticas, característica objetivo, y/o características reales tal como se explicó anteriormente. En este caso, el modelo de flujo de datos describe un flujo de control que funciona en metadatos y un flujo de datos que funciona en datos que reflejan características objetivo o reales.

Tal como se muestra en la figura 13, el motor 12 de modelado ejecuta una etapa S26, ejecutando el módulo 36 de uso de vector de entrada, para modelar vectores de entrada en el modelo funcional que representan vectores de estado externo y/o vectores de estado interno tal como se explicó anteriormente de manera resumida.

Tal como se muestra en la figura 13, el motor 12 de modelado ejecuta una etapa S28, ejecutando el módulo 36 de

uso de vector de entrada, para modelar condiciones para transiciones de estados, etapa S28 que es opcional.

La figura 14 muestra una división de un sistema distribuido de eventos discretos según la presente invención, en el que la división se define como un conjunto de agrupaciones de subsistemas de tal manera que todos los subsistemas asignados a una agrupación de subsistemas específica se modelan con un mismo tipo de metaobjeto.

Tal como se explicó anteriormente de manera resumida, según la presente invención los diferentes niveles abstracción en el modelado de sistemas distribuidos de eventos discretos (es decir, metanivel, nivel de uso, y nivel de gemelo) implican una división de sistemas distribuidos de eventos discretos en subsistemas que a su vez están compuestos por gemelos digitales.

Suponiendo la noción de sistemas distribuidos de eventos discretos, subsistemas relacionados y flujos entre los subsistemas, un asunto que debe resolverse para el modelado del sistema distribuido de eventos discretos es cómo dividir la estructura global del sistema distribuido de eventos discretos en el al menos un subsistema que pueda entonces modelarse usando un tipo de metaobjeto MO• relacionado tal como se describió anteriormente.

Hasta ahora, inicialmente debe determinarse cuál de los tipos de metaobjeto MO• disponibles son realmente relevantes para un caso particular de división de un sistema distribuido de eventos discretos.

Tal como se muestra en la figura 14, según la presente invención la división basada en tipo de metaobjeto según el tipo de metaobjeto MO• implica la definición de agrupaciones de subsistemas MOC = {MO1, ..., MOp} de tal manera que cada agrupación de subsistemas MOi ∈ MOC comprende al menos un subsistema que está modelándose con un mismo tipo de metamodelo MO•.

En más detalle, MOT = {tipo1, ...,tipot} es un conjunto de todos los tipos de metaobjeto MO• disponibles para el modelado de un sistema distribuido de eventos discretos y el mapeo MOT(MOi): MOi → tipo de MOi ∈ MOT es una función que mapea una identificación de subsistema MOi a su tipo de metamodelo MO• relacionado.

Entonces, la división basada en tipo de metaobjeto P(x) = G [SS, STR] (x) del sistema distribuido de eventos discretos significa que diferentes subsistemas de una agrupación de subsistemas tienen un mismo tipo de metamodelo MO•, es decir, que para todos los MOi, MOj ∈ MOCk con i ≠ j, MOT (MOi) = MOT (MOj) = MO•.

Además, una división del sistema distribuido de eventos discretos *P*(x) puede representarse como un dígrafo según *P*(x) = G [SS, STR] (x) . En más detalle, SS = {SS1, ...,SSp} es un de agrupaciones de subsistemas según tipos de metaobjeto {MO1, ...,MOp} y STR = {STR1, ..., STRq} es un conjunto de flujos entre los subsistemas del sistema distribuido de eventos discretos. En este caso, las agrupaciones de subsistemas tienen una variedad de expresiones comunes, semántica, y comportamiento de metamodelo de subsistemas, así como un conjunto común de metadatos, rasgos y características.

Además, según la presente invención, dentro de cada agrupación de subsistemas que refleja un determinado tipo de metaobjeto MO•, se introducen agrupaciones secundarias de subsistemas según diferentes usos de los subsistemas que están modelándose mediante el tipo de metaobjeto MO•.

Además, con respecto a cada agrupación de subsistemas que tiene el tipo de metamodelo i, se describen diferentes situaciones de uso del tipo de metamodelo i subyacente, situaciones de uso que se modelan como agrupaciones de uso de subsistema MOUCi = {MOUC1, ..., MOUCu}i.

Un resultado de la división y agrupación basadas en tipo de metaobjeto es que se agrupan subsistemas del sistema distribuido de eventos discretos según el tipo de metaobjeto MO• relacionado. Dentro de cada agrupación de subsistemas se describen opcionalmente diferentes agrupaciones de uso de subsistema que tienen un mismo tipo de metaobjeto MO• pero diferentes usos del tipo de metamodelo MO•.

Una vez que está disponible una división del sistema distribuido de eventos discretos, un punto adicional que debe resolverse a nivel de modelado a nivel de uso es la configuración de una topología de enrutamiento para el intercambio de mensajes de flujo continuo entre agrupaciones de subsistemas MOCi ∈ MOC.

Tal como se muestra en la figura 14, se define una topología de enrutamiento con respecto a cada agrupación de subsistemas MOCi ∈ MOC. En más detalle, para cada agrupación de subsistemas MOCi se define un conjunto MOC_rec_i de al menos un agrupación de subsistemas a partir del cual la agrupación de subsistemas MOCi recibe mensajes de flujo continuo. Además, para cada agrupación de subsistemas MOCi se define un conjunto MOC_send_i de al menos una agrupación de subsistemas a que la agrupación de subsistemas MOCi envía mensajes de flujo continuo.

Entonces, sigue la configuración de una topología de enrutamiento entre agrupaciones de subsistemas del sistema distribuido de eventos discretos para avanzar en la transición desde el modelado a metanivel hasta el modelado a nivel de uso.

Además, una etapa final en el procedimiento de modelado a nivel de uso se refiere a una transferencia del resultado de modelado a nivel de uso a una estructura principal de software ejecutable. La estructura principal de software ejecutable sirve para ejecutar un motor de gemelos digitales para el control de un sistema distribuido discreto uniforme en tiempo real.

En conclusión, la división basada en tipo de metaobjeto reduce significativamente la cantidad de datos que van a transmitirse en flujo continuo entre diferentes agrupaciones de subsistemas en comparación con un enfoque de modelado en el que se logra una representación de composición a nivel de objetos gemelos digitales sin abstracción de modelado a un metadominio y dominio a nivel de uso.

El motivo para esto es que la transmisión en flujo continuo entre diferentes agrupaciones de subsistemas se realiza como metaflujo. En este caso, para cada agrupación de subsistemas, los objetos gemelos digitales relacionados (agrupados de manera secundaria según uso de tipo de metamodelo) emiten por radiodifusión datos en relación con eventos internos de la agrupación de subsistemas en un canal de transmisión en flujo continuo común asignado a la agrupación de subsistemas modelada con un tipo de metaobjeto MO• específico.

De lo contrario, cada otra agrupación de subsistemas recibe datos en relación con eventos externos de una agrupación de subsistemas de determinado tipo de metaobjeto MO• en un canal de metaflujo asignado para el otro tipo de metamodelo.

En conjunto, el número de canales de transmisión en flujo continuo depende únicamente del número de agrupaciones de subsistemas y tipos de metaobjeto MO• relacionados y no del número de subsistemas reales y objetos gemelos relacionados. Esto tiene una gran influencia sobre la complejidad de la topología de enrutamiento entre agrupaciones de subsistemas ya que el número de canales de comunicación para un procedimiento de comunicación completo dentro del sistema distribuido de eventos discretos aumenta linealmente con el número de tipos de metaobjeto MO• y no exponencialmente con el número de subsistemas reales.

Además, debe observarse que los propios subsistemas pueden descomponerse para dar subsistemas adicionales relacionados conduciendo a la introducción de jerarquía en el procedimiento de modelado global.

La figura 15 muestra un diagrama esquemático del motor 14 de servicio según la presente invención mostrado en la figura 10, en el que el motor 14 de servicio genera un modelo de uso de al menos un subsistema de un sistema distribuido de eventos discretos correspondiente a un tipo de metaobjeto usado para modelar el al menos un subsistema.

Tal como se muestra en la figura 15, el motor de servicio comprende un módulo 40 de división adaptado para dividir el sistema distribuido de eventos discretos en al menos dos agrupaciones de subsistemas que comprenden, cada una, al menos un subsistema del sistema distribuido de eventos discretos tal como se explicó con respecto a la figura 14. En este caso, todos los subsistemas asignados a una agrupación de subsistemas específica MOCi $\in$ MOC se modelan con un mismo tipo de metaobjeto MO•.

Tal como se muestra en la figura 15, el motor de servicio comprende un módulo 42 de enrutamiento adaptado para definir una topología de enrutamiento para el intercambio de mensajes de flujo continuo entre agrupaciones de subsistemas MOCi $\in$ MOC.

Tal como se muestra en la figura 15, el motor de servicio comprende un módulo 44 de agrupación de uso adaptado para modelar diferentes agrupaciones de uso de subsistema MOUCi = {MOUC1, ..., MOUCu}i para todos los subsistemas asignados a una agrupación de subsistemas del sistema distribuido de eventos discretos.

Preferiblemente, la topología de enrutamiento define, para cada agrupación de subsistemas MOCi, al menos una agrupación de subsistemas externos de la que se reciben mensajes de flujo continuo y al menos una agrupación de subsistemas externos a la que se envían mensajes de flujo continuo.

Tal como se muestra en la figura 15, el motor de servicio comprende un módulo 46 de configuración de software adaptado para transformar el modelo de uso de al menos un subsistema de un sistema distribuido de eventos discretos en al menos un núcleo de software de modelado ejecutable que puede ejecutarse en un motor de gemelos digitales para el control en tiempo real de un sistema distribuido discreto uniforme.

Preferiblemente, el módulo 46 de configuración de software está adaptado para:

configurar el al menos un núcleo de software de modelado ejecutable por cada agrupación de sistemas MOCi mediante parametrización de software ejecutable disponible a partir de una biblioteca de software; en el que

el al menos un núcleo de software de modelado ejecutable está configurado para ejecutar al menos un modelo de uso de subsistema en relación con cada uso de subsistema de la agrupación de subsistemas MOCi; y

el al menos un núcleo de software de modelado ejecutable está configurado para ejecutar una interfaz de entrada y una interfaz de salida para el intercambio de flujos de mensajes que tienen formato de mensaje de metanivel hacia y desde el al menos un núcleo de software de modelado ejecutable.

Además, según la presente invención, el módulo 46 de configuración de software está adaptado para configurar una topología de enrutamiento entre al menos dos núcleos de software de modelado ejecutables conectando interfaces de entrada e interfaces de salida relacionadas mediante al menos un canal de comunicación de metaflujo.

Además, según la presente invención el módulo 46 de configuración de software está adaptado para cargar el al menos un núcleo de software de modelado ejecutable en el motor de gemelos digitales para el control de un sistema distribuido discreto uniforme en tiempo real.

La figura 16 muestra un diagrama de flujo de funcionamiento para el modelado a nivel de uso según la presente invención tal como se muestra en la figura 14 y la figura 15, respectivamente.

Tal como se muestra en la figura 16, para lograr el modelado a nivel de uso según la presente invención se ejecuta una etapa S30, ejecutando el módulo 40 de división mostrado en la figura 15, para dividir el sistema distribuido de eventos discretos en al menos dos agrupaciones de subsistemas que comprenden, cada una, al menos un subsistema del sistema distribuido de eventos discretos, en el que todos los subsistemas asignados a una agrupación de subsistemas específica MOCi ∈ MOC se modelan con un mismo tipo de metaobjeto MO•.

Tal como se muestra en la figura 16, tras la etapa S30 se ejecuta una etapa S32, ejecutando el módulo 42 de enrutamiento mostrado en la figura 15, para definir una topología de enrutamiento para el intercambio de mensajes de flujo continuo entre agrupaciones de subsistemas MOCi ∈ MOC.

Preferiblemente, la etapa 32 define una topología de enrutamiento definiendo para cada agrupación de subsistemas MOCi al menos una agrupación de subsistemas externos de la que se reciben mensajes de flujo continuo y al menos una agrupación de subsistemas externos a la que se envían mensajes de flujo continuo.

Tal como se muestra en la figura 16, tras la etapa S32 se ejecuta una etapa S34, ejecutando el módulo 42 de agrupación de uso mostrado en la figura 15, para definir diferentes agrupaciones de uso de subsistema MOUCi = {MOUC1, ..., MOUCu}i) para todos los subsistemas asignados a una agrupación de subsistemas del sistema distribuido de eventos discretos.

Tal como se muestra en la figura 16, tras la etapa S34 se ejecuta una etapa S36, ejecutando el módulo 46 de configuración de software mostrado en la figura 15, para configurar software para transformar el modelo de uso de al menos un subsistema de un sistema distribuido de eventos discretos en al menos un núcleo de software de modelado ejecutable que puede ejecutarse en un motor de gemelos digitales para el control en tiempo real de un sistema distribuido discreto uniforme.

Los datos específicos de la configuración de software según la etapa S36 se han explicado anteriormente de manera resumida con respecto a la figura 15 y, por tanto, no se repetirán en este caso.

Finalmente, durante la ejecución de la etapa S36, el al menos un núcleo de software de modelado ejecutable se carga en el motor 16 de gemelos virtuales para el control de un sistema distribuido discreto uniforme en tiempo real.

Aunque anteriormente se han descrito diferentes aspectos de la presente invención como que están relacionados con el modelado de un sistema distribuido de eventos discretos, a continuación se explicará el modelo generado para control de procedimiento en tiempo real del sistema distribuido de eventos discretos.

La figura 17 muestra un diagrama esquemático de un motor 16 de gemelos virtuales según la presente invención.

A continuación se supondrá que el motor 16 de gemelos virtuales tiene preinstalado al menos un núcleo de software de modelado ejecutable preparado por el motor 15 de servicio tal como se muestra en la figura 15.

Tal como se explicó anteriormente de manera resumida, esto implica una preinstalación de al menos un núcleo de software de modelado por cada agrupación de sistemas del sistema distribuido discreto uniforme y tipo de metaobjeto MO• relacionado usado para el modelado a metanivel de la agrupación de sistemas.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende un módulo 48 de procesamiento de modelo adaptado para ejecutar en tiempo real al menos un modelo de uso de subsistema en relación con la agrupación de subsistemas mediante la ejecución de un núcleo de software de modelado relacionado que modela el al menos un modelo de uso de subsistema.

Según la presente invención, el módulo 48 de procesamiento de modelo está adaptado para hacer funcionar al

menos un gemelo digital con respecto al al menos un modelo de uso de subsistema como homólogo virtual de un objeto de procedimiento relacionado que se hace funcionar en el sistema distribuido discreto uniforme y que tiene una funcionalidad correspondiente al modelo de uso de subsistema.

Además, según la presente invención, el módulo 48 de procesamiento de modelo está adaptado para hacer funcionar el al menos un gemelo digital de una manera pasiva mediante acceso en tiempo real al núcleo de software de modelado que modela el modelo de uso de subsistema del al menos un gemelo digital.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende una interfaz 50 de motor de modelado adaptada para inyectar información de modelado en el módulo 48 de procesamiento de módulo activando al menos un tipo de metaobjeto MO• a partir de todos los tipos de metaobjeto MO• disponibles soportados por el motor de modelado en el motor de gemelos digitales.

Preferiblemente, la interfaz 50 de motor de modelado está adaptada para inyectar un conjunto de modelos de estados Σij, en el que i es el índice para el modelo de estados y j es el índice para el uso relacionado, que representa el que comprende al menos modelo de estados que representa el tipo de metaobjeto MO• en el módulo 48 de procesamiento de módulo.

De manera adicionalmente preferible, la interfaz 50 de motor de modelado está adaptada para inyectar al menos una perspectiva Pj en el al menos un modelo de estados inyectado en el módulo 48 de procesamiento de módulo.

De manera adicionalmente preferible, la interfaz 50 de motor de modelado está adaptada para inyectar diferentes niveles de abstracción de datos según metadatos, datos objetivo, y datos reales para cada modelo de estados y variables de estado relacionadas en el módulo 48 de procesamiento de módulo.

De manera adicionalmente preferible, la interfaz 50 de motor de modelado está adaptada para inyectar al menos una agrupación de uso de subsistema con un tipo de metamodelo predeterminado y al menos un gemelo digital relacionado. En este caso, el al menos un gemelo digital sirve como homólogo digital para un objeto de procedimiento del mundo real correspondiente.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende una interfaz 52 de motor de servicio adaptada para inyectar código de software que implementa transiciones de estados externas y/o internas en el módulo 48 de procesamiento de módulo.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende un módulo 54 de control de modelado adaptado para inicializar al menos un gemelo digital relacionado en relación con al menos una agrupación de uso de subsistema. Preferiblemente, el módulo 54 de control de modelado está adaptado para inicializar el al menos un gemelo digital antes del inicio del motor de gemelos digitales o en tiempo real durante el funcionamiento del motor de gemelos digitales.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende un módulo 56 de procesamiento de flujo de datos adaptado para gestionar el intercambio de datos hacia/desde el motor de gemelos virtuales.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende un módulo 58 de control de flujo de datos adaptado para el intercambio de datos de control hacia/desde el motor de gemelos virtuales clasificando datos de comunicación según diferentes niveles de abstracción en al menos flujo de datos de control y/o flujo de datos de uso tal como se explicará en más detalle a continuación.

Tal como se muestra en la figura 17, el motor 16 de gemelos virtuales según la presente invención comprende un módulo 60 de entrada/salida de metaflujo para el intercambio de mensajes de metaflujo hacia y desde el motor 16 de gemelos virtuales.

La figura 18 muestra un diagrama de flujo de funcionamiento para el motor 16 de gemelos virtuales según la presente invención tal como se muestra en la figura 17.

Tal como se muestra en la figura 18, para lograr el control de procedimiento en tiempo real de un sistema distribuido de eventos discretos de modelo se ejecuta una etapa S38, ejecutando el módulo 48 de uso de procesamiento de modelo, para ejecutar en tiempo real al menos un modelo de uso de subsistema en relación con la agrupación de subsistemas mediante la ejecución de un núcleo de software de modelado relacionado que modela el al menos un modelo de uso de subsistema.

Tal como se muestra en la figura 18, para lograr el control de procedimiento en tiempo real de un sistema distribuido de eventos discretos de modelo se ejecuta una etapa S40, ejecutando el módulo 48 de uso de procesamiento de modelo, para hacer funcionar al menos un gemelo digital con respecto al al menos un modelo de uso de subsistema como homólogo virtual de un objeto de procedimiento relacionado que se hace funcionar en el sistema distribuido

discreto uniforme y que tiene una funcionalidad correspondiente al modelo de uso de subsistema.

Según la presente invención, la etapa S40 de hacer funcionar el al menos un gemelo digital se ejecuta de una manera pasiva mediante acceso en tiempo real al núcleo de software de modelado que modela el modelo de uso de subsistema del al menos un gemelo digital.

Tal como se muestra en la figura 18, para lograr el control de procedimiento en tiempo real de un sistema distribuido de eventos discretos de modelo se ejecuta una etapa S42, ejecutando el módulo 48 de uso de procesamiento de modelo, para modificar un modelo del sistema distribuido de eventos discretos en tiempo real mediante inserción/deleción de componentes de modelado y/o gemelos digitales.

Preferiblemente, la etapa S42 se ejecuta para inyectar información de modelado activando al menos un tipo de metaobjeto MO• a partir de todos los tipos de metaobjeto MO• disponibles soportados por el motor 12 de modelado en el módulo 48 de procesamiento de módulo del motor 16 de gemelos virtuales.

Preferiblemente, la etapa S42 se ejecuta para inyectar al menos modelos de estados Σij que comprenden al menos un modelo de estados que representa el tipo de metaobjeto MO• en el módulo 48 de procesamiento de módulo del motor 16 de gemelos virtuales.

Preferiblemente, la etapa S42 se ejecuta para inyectar al menos una perspectiva Pj en el al menos un modelo de estados inyectado en el módulo 48 de procesamiento de módulo del motor 16 de gemelos virtuales.

Preferiblemente, la etapa S42 se ejecuta para inyectar diferentes niveles de abstracción de datos según metadatos, datos objetivo, y datos reales para cada modelo de estados y variables de estado relacionadas en el módulo 56 de procesamiento de flujo de datos de gemelo motor 16.

Preferiblemente, la etapa S42 se ejecuta para inyectar código de software que implementa transiciones de estados externas y/o internas en el módulo 48 de procesamiento de módulo de la máquina 16 de gemelos virtuales.

Preferiblemente, la etapa S42 se ejecuta para inyectar al menos una agrupación de uso de subsistema con un tipo de metaobjeto predeterminado en el módulo 48 de procesamiento de módulo de la máquina 16 de gemelos virtuales.

Tal como se muestra en la figura 18, la etapa S42 también se ejecuta para inicializar al menos un gemelo digital en relación con una agrupación de uso de subsistema, en la que el al menos un gemelo digital sirve como homólogo digital para un objeto de procedimiento del mundo real correspondiente. Preferiblemente, el al menos un gemelo digital se inicializa antes del inicio del motor de gemelos virtuales o durante el funcionamiento del motor de gemelos virtuales para control de procedimiento en tiempo real del sistema distribuido de eventos discretos.

Tal como se muestra en la figura 18, para lograr el control de procedimiento en tiempo real de un sistema distribuido de eventos discretos de modelo se ejecuta una etapa S44, ejecutando el módulo de control de flujo de datos 88, para el intercambio de datos de control hacia/desde el motor de gemelos digitales clasificando datos de comunicación según diferentes niveles de abstracción en al menos flujo de datos de control y/o flujo de datos de uso.

Preferiblemente, en la etapa S44 se evalúa al menos un vector de estado que representa al menos un evento desencadenante para al menos una transición de estado de un gemelo digital.

De manera adicionalmente preferible, en la etapa S44 el flujo de datos de control se controla generando al menos un mensaje de metaflujo en relación con una transición de estado de un gemelo digital para el intercambio de datos de control relacionados entre diferentes agrupaciones de subsistemas.

La figura 19 muestra detalles de funcionamiento adicionales para el módulo 48 de procesamiento de módulo tal como se hace funcionar en el motor 16 de gemelos virtuales.

Tal como se muestra en la figura 19, en la unidad 48 de procesamiento de módulo se ejecuta código de software funcional inyectado que implementa tipos de metamodelo específicos y modelos de uso relacionados de los mismos.

Después, para cada agrupación de uso de subsistema en cada agrupación de subsistemas se introduce al menos un gemelo digital como homólogo digital en el dominio digital de un objeto de procedimiento en el mundo real en el dominio de procedimiento. Dicho de otro modo, cada gemelo digital refleja el comportamiento lógico de un objeto de procedimiento en el dominio que va a modelarse.

En más detalle, para cada agrupación de uso de subsistema MOUCk ∈ {MOUC1, ..., MOUCu} de una agrupación de subsistemas que tienen el tipo de metamodelo i se instancia un conjunto de al menos un objeto gemelo digital TOk,i = {to1, ..., tol}k,i que realiza un homólogo digital de un objeto de procedimiento del mundo real.

Debe observarse que la introducción de gemelos digitales puede ejecutarse o bien antes del inicio del motor de gemelos digitales o bien durante el control de procedimiento en tiempo real del sistema distribuido de eventos discretos. Esto tiene una importancia significativa ya que esta capacidad de configuración en tiempo real del motor de gemelos digitales logra una flexibilidad y configurabilidad mucho mayores en comparación con sistemas de control de procedimiento cableados.

Además, dado que durante la configuración inicial aún no se dispone de valores objetivo de modo que se declaran según el tipo de datos y se asigna una memoria relacionada. De manera similar, también se consideran valores reales mediante declaración de tipo de datos y asignación de memoria.

El resultado de la configuración inicial es un conjunto de núcleos de software ejecutables en relación con agrupaciones de uso de subsistema.

Después de eso, la ejecución del motor de gemelos virtuales implica ejecutar modelos de uso de subsistema en relación con usos de subsistema de agrupaciones de subsistemas en tiempo real mediante la ejecución del núcleo de software de modelado que modela el uso de subsistema.

Además, se hace funcionar al menos un gemelo digital con respecto a cada modelo de uso de subsistema como homólogo digital y en una relación de uno a uno con respecto a al menos un objeto de procedimiento que se hace funcionar en el sistema distribuido discreto uniforme y que tiene una funcionalidad modelada mediante el gemelo digital.

En particular, el al menos un gemelo digital se hace funcionar mediante acceso en tiempo real al núcleo de software de modelado relacionado que se ejecuta en tiempo real. Por tanto, el comportamiento de un objeto gemelo no se modela explícitamente sino que se hereda a partir de la etapa de modelado a nivel de uso.

Además, según la presente invención es posible encontrar un compromiso entre el tamaño de agrupaciones de uso de subsistema y el número de gemelos. En un primer caso, el tamaño de cada agrupación de uso es uno y las agrupaciones de uso de subsistema cubren exactamente un objeto gemelo. Esto se denomina modelado explícito. Alternativamente, una agrupación de uso de subsistema puede cubrir más de un objeto gemelo según una relación de 1:N. Esto se denomina modelado implícito.

La figura 20 muestra un diagrama esquemático detallado adicional que ilustra el funcionamiento del motor 16 de gemelos virtuales según la presente invención tal como se muestra en la figura 17.

Tal como se muestra en la figura 20, otro aspecto de ejecutar el motor 16 de gemelos virtuales según la presente invención se refiere al control de flujo de datos con respecto al intercambio de datos hacia y desde el motor 16 de gemelos virtuales.

Tal como se muestra en la figura 20, según la presente invención el control de flujo de datos se basa en dividir los datos en diferentes clases, por ejemplo, flujo de datos de control, flujo de datos a nivel de uso en relación con la ejecución en tiempo real del/de los tipo(s) de metamodelo, tipo(s) de modelo de uso y objeto(s) gemelo(s) relacionado(s). Una categoría adicional de datos puede estar relacionada con datos de procedimiento, por ejemplo, datos de sensor/accionador alimentados hacia y/o desde el motor de gemelos digitales.

Debe observarse que, según la presente invención, el flujo de datos a nivel de uso puede usarse para influir en el flujo de datos de control. Además, el flujo de datos a nivel de uso puede especificarse en el nivel de uso del modelado y puede evaluarse en el nivel de gemelo del modelado. Finalmente, las condiciones operativas relacionadas con transición de estado pueden especificarse mediante condiciones a nivel de uso o condiciones a metanivel.

A la vista de lo anterior, la separación entre datos de control y datos relacionados con modelo según la presente invención corresponde al metamodelo implementado por ordenador explicado anteriormente de manera resumida y permite lograr un aumento de eficiencia significativo durante el funcionamiento del sistema global.

Tal como se muestra en la figura 20, un primer elemento de control de flujo de datos relacionado con datos de control se refiere a la evaluación de vectores de estado externos y/o internos que representan eventos desencadenantes para transiciones de estados en los modelos de uso que modelan el funcionamiento de los gemelos digitales.

Tal como se muestra en la figura 20, un segundo elemento de control de flujo de datos relacionado con datos de control se refiere a la generación de mensajes de metaflujo para el intercambio con agrupaciones de subsistemas que no son idénticas a la agrupación de subsistemas en la que se produce una transición de estado en consideración.

Tal como se muestra en la figura 20, un primer elemento de control de flujo de datos relacionado con modelo se

refiere a la evaluación de vectores de condición que representan condiciones para transiciones de estados en los modelos de uso que modelan el funcionamiento de los gemelos digitales. En más detalle, tales condiciones se evalúan teniendo en cuenta datos que representan valores reales u objetivo y en asociación con diferentes estados del modelo de estados a nivel de uso que se hace funcionar en relación con gemelos digitales.

Tal como se muestra en la figura 20 y suponiendo que se cumplen condiciones para una transición de estado particular, un segundo elemento de control de flujo de datos relacionado con modelo se refiere a lograr la transición real desde el estado inicial hasta el estado objetivo. Esto implica la aplicación de la función de transformación asociada a datos que representan el estado inicial para la generación de datos que representan el estado objetivo. Además, una transición desde el estado inicial hasta el estado objetivo necesita al menos una actualización de información de contexto en correspondencia con el objeto gemelo para el que se gestiona la transición de estado.

Tal como se muestra en la figura 20, un tercer elemento de control de flujo de datos relacionado con modelo se refiere al procesamiento del/de los resultado(s) de transición de estado. En este caso, se presenta información relacionada con el estado objetivo al segundo elemento de control de flujo de datos de control, es decir la generación de mensajes de metaflujo.

Una vez terminada una transición de estado, se genera un mensaje de metaflujo relacionado para informar a otras agrupaciones de subsistemas con diferente tipo de metamodelo sobre la transición de estado.

Además, una vez terminada una transición de estado, también es necesario actualizar el la representación de modelo de espacio de estados relacionada dentro de la agrupación de subsistemas en la que se ejecuta el modelo de espacio de estados relacionado.

Tal como se indicó anteriormente, una categoría adicional de datos puede estar relacionada con datos de procedimiento, por ejemplo, datos de sensor/accionador alimentados hacia y/o desde el motor de gemelos digitales. En este caso, se intercambian datos relacionados en tiempo real entre las representaciones de objeto gemelo e interfaces de sensor/accionador del motor de gemelos digitales.

La figura 21 muestra un diagrama de flujo de funcionamiento de detalle adicional para el módulo de procesamiento de flujo de datos mostrado en la figura 17 y 20, respectivamente.

Tal como se muestra en la figura 21, según la presente invención el módulo 52 de control de flujo de datos está adaptado para controlar un módulo 62 de evaluación de vector de estado que funciona en el flujo de datos de control para evaluar al menos un vector de estado que representa al menos un evento desencadenante para al menos una transición de estado de un gemelo digital.

Tal como se muestra en la figura 21, el módulo 58 de control de flujo de datos está adaptado para controlar un módulo 64 de generación de mensaje que funciona en el flujo de datos de control para generar al menos un mensaje de metaflujo en relación con una transición de estado de un gemelo digital para el intercambio de datos de control relacionados entre diferentes agrupaciones de subsistemas.

Tal como se muestra en la figura 21, el módulo 58 de control de flujo de datos está adaptado para controlar un módulo 66 de evaluación de vector de condición que funciona en el flujo de datos de uso para evaluar al menos un vector de condición que comprende al menos un valor real y/o al menos un valor objetivo en asociación con la al menos una transición de estado del al menos un gemelo digital para la evaluación de una condición de transición relacionada.

Tal como se muestra en la figura 21, el módulo 58 de control de flujo de datos está adaptado para controlar un módulo 68 de evaluación de error que funciona en el flujo de datos de uso para evaluar al menos una indicación de condición para un error en el funcionamiento en el motor 16 de gemelos virtuales. Si se produce un error, entonces un módulo 70 de gestión de error o bien resolverá el problema de modo que pueda reanudarse el funcionamiento o bien terminará el funcionamiento global del motor 16 de gemelos virtuales.

Tal como se muestra en la figura 21, el módulo 58 de control de flujo de datos está adaptado para controlar un módulo 72 de transformación que funciona en el flujo de datos de uso para aplicar al menos una función de transformación a datos que representan un estado inicial antes de una transición de estado para generar datos que representan un estado objetivo después de la transición de estado.

Tal como se muestra en la figura 21, el módulo 58 de control de flujo de datos está adaptado para controlar un módulo 74 de actualización que funciona en el flujo de datos de uso para actualizar información relacionada con representación de estados y opcionalmente información de contexto en relación con una transición de estado.

En conclusión, según la presente invención se propone una división de flujo de datos en flujo de datos de control y flujo de datos de uso. Esto reduce la complejidad en la comunicación y mejora una capacidad en tiempo real dado que las operaciones en flujo de datos de control y flujo de datos de uso pueden paralelizarse, por ejemplo. Además,

permite aumentar la flexibilidad dado que los datos de uso pueden tener un impacto sobre el flujo de control.

La figura 22 muestra un ejemplo de un enfoque de caja de semillas usado para la evaluación de vectores de estado y vectores de condición en el módulo de procesamiento de flujo de datos según la presente invención tal como se muestra en la figura 17.

Tal como se explicó anteriormente de manera resumida, según la presente invención se consideran vectores de estado externo que desencadenan una transición de estado en un modelo de espacio de estados tras producirse un evento que es externo a la agrupación de subsistemas en la que se usa el modelo de estados.

Tal como se explicó anteriormente de manera resumida, un l-ésimo vector de estado externo Θex,i•at,l = ((MOj, j ≠ •, zj), ..., (MOk, k ≠ •, zk))Tl respectivamente resume tuplas de subsistemas externos a un subsistema que está modelándose usando el tipo de metaobjeto MO• y estados correspondientes en los que los subsistemas externos indicados prevalecen. Además, anteriormente se explicó que, para diferentes transiciones de estados puede existir una pluralidad de vectores de estado externo.

Según la presente invención, puede suponerse sin pérdida de generalidad que se establecen diferentes vectores de estado externo por transición de estado en relación con diferentes contextos.

Además, según la presente invención se preparan metamodelos como base para la configuración de modelos a nivel de uso. Entonces, para cada uso dedicado de un metamodelo específico, se instancia al menos un gemelo digital como homólogo digital de un objeto de procedimiento en el mundo real.

A partir de lo anterior, debe quedar claro que vectores de estado externo tal como se especifican en el metanivel se replican según varios gemelos digitales relacionados y diferentes contextos relacionados que acompañan a una transición de estado de los gemelos digitales.

Además, para cada modelo de uso que está haciéndose funcionar en el motor de gemelos virtuales se hace funcionar al menos un modelo de espacio de estados relacionado. Suponiendo que para cada modelo de uso se instancia al menos un gemelo digital, entonces para cada estado en el modelo de espacio de estados relacionado se considera al menos un uso de estado por gemelo o, de manera equivalente, uso de estado por gemelo digital.

Dicho de otro modo, suponiendo que para un modelo de uso particular se instancia una pluralidad de gemelos, entonces para cada estado del modelo de espacio de estados relacionado en el nivel de abstracción de uso se considerará una pluralidad correspondiente de usos de estado en el nivel de abstracción de gemelo.

A partir de lo anterior y tal como se muestra en la figura 22, la conclusión es que, para cada conjunto de vectores de estado externo por tipo de subsistema y un uso de tipo de subsistema relacionado pueden organizarse en una forma de matriz en la que las filas corresponden a los usos de estado con respecto a gemelos digitales del modelo de uso de subsistema y las columnas representan un(os) contexto(s) diferente(s) para transiciones de estados consideradas en el nivel de gemelo de abstracción de modelo.

La figura 23 muestra un ejemplo adicional de un enfoque de caja de semillas para la evaluación de vectores de estado y vectores de condición en el módulo de procesamiento de flujo de datos según la presente invención tal como se muestra en la figura 17.

Tal como se muestra en la figura 23, según la presente invención se preparan cajas de semillas para cada agrupación de subsistemas del modelo a nivel de uso de un sistema distribuido de eventos discretos.

Según la presente invención, la preparación de las cajas de semillas corresponde a la topología de enrutamiento en la abstracción a nivel de uso del modelado.

En más detalle y tal como se muestra en la figura 23, para cada agrupación de subsistemas se preparará una caja de semillas en relación con cada una de las otras agrupaciones de subsistemas a las que escucha la agrupación de subsistemas considerada. Además, se preparará una caja de semillas en relación con cada tipo de metaobjeto que es aplicable en la otra agrupación de subsistemas a la que escucha la agrupación de subsistemas considerada. Finalmente, también se prepararán cajas de semillas para la gestión de transiciones de estados internas de modelos de espacio de estados que se hacen funcionar de manera interna dentro de una agrupación de subsistemas.

La figura 24 muestra un ejemplo de actualización de una caja de semillas tal como se muestra en las figuras 22 y 23 tras la ejecución de una transición de estado con respecto a un objeto gemelo.

Tal como se muestra en la figura 24, una manera eficiente de representar una matriz de usos de estado frente a contextos tal como se muestra en la figura 22 de una manera dispersa es usar una lista vinculada de pares de gemelo/contexto con respecto a cada uso de estado. Después, una vez que un gemelo ejecuta una transición de estado, esto implica la cancelación del par de gemelo/contexto relacionado a partir de la lista en relación con el

estado de origen de la transición de estado y la adición de un par de gemelo/contexto actualizado a una segunda lista en relación con el estado objetivo de la transición de estado.

Aunque anteriormente se ha descrito la presente invención con referencia a los dibujos y figuras de realizaciones preferidas de la invención, debe observarse que, evidentemente, la presente invención también puede implementarse usando variaciones y modificaciones de la misma que resultarán evidentes y pueden realizarse fácilmente por los expertos en la técnica siempre que permanezcan dentro del alcance de la invención tal como se define por las reivindicaciones adjuntas. Por ejemplo, las funcionalidades descritas anteriormente pueden realizarse en software, en hardware, o una combinación de los mismos.

Glosario:

Vectores y conjuntos:

x = (x1, ..., xk) T vector de entrada

X conjunto de vectores de entrada

y = (y1, ..., yn)T vector de salida

Y conjunto de vectores de salida

z = (z1, ..., zm)T vector de variable de estado

Z conjunto de vectores de variables de estados

SS = {SS1, ...,SSp} conjunto de subsistemas mutuamente separados de un sistema distribuido de eventos discretos

STR = {STR1, ..., STRq} conjunto de flujos entre subsistemas de un sistema distribuido de eventos discretos

MOC = {MOC1, ..., MOCp} conjunto de agrupaciones de subsistemas que describen una división de un sistema distribuido de eventos discretos

MOC_rec_i conjunto de al menos una agrupación de subsistemas a partir de la cual la agrupación de subsistemas MOCi recibe mensajes de flujo continuo

MOC_send_i conjunto de al menos una agrupación de subsistemas a la que la agrupación de subsistemas MOCi envía mensajes de flujo continuo

MOT = {MOT1, ...,MOTp} conjunto de tipos de metaobjeto disponibles para el modelado de un sistema distribuido de eventos discretos

MOUCi = {MOUC1, ..., MOUCui}i agrupaciones de uso de subsistema de subsistemas dentro de un subsistema que está modelándose con un tipo de metaobjeto específico tipo i

MO•R objeto de representación de tipo de metaobjeto MO•

D• = {d1, ..., do} conjunto de características descriptivas del tipo de metaobjeto MO•

Vector de estado externo para el tipo de metaobjeto MO•: Θex,i•at,I = ((SSTj, j ≠ •, zj), ..., (SSTk, k ≠ •, zk))IT

Πex,i•at = UI Θex,i•at,I conjunto de todos los vectores de estado externo que desencadenan una transición de estado debida a un evento externo que es relevante para el tipo de metaobjeto MOT•

Δex,i•a conjunto de condiciones que deben cumplirse antes de desencadenar una transición de estado debida a un evento externo que es relevante para el tipo de metaobjeto MOT•

Πin,i•at conjunto de vectores de estado interno que desencadenan una transición de estado debida a un evento interno que es relevante para el tipo de metaobjeto MOT•

Δin,i•a conjunto de condiciones que deben cumplirse antes de desencadenar una transición de estado debida a un evento interno que es relevante para el tipo de metaobjeto MOT•

C = (con_ID1, ..., con_IDn) conjunto de contextos

Funciones:

Φ función de transición de estado

Φin transiciones de estados debidas a eventos internos

Φex transiciones de estados debidas a eventos externos

Ω función de salida

τ función de tiempo de retención

Operadores y modelos:

S operador de sistema

Σ = <X, Z, Y, Φin, Φex, Ω, τ> modelo de estados de un sistema distribuido de eventos discretos

P(x) = G [SS, STR] (x) representación en dígrafo de una división de composición de un sistema distribuido de eventos discretos

Σi• i-ésimo modelo de estados usado en el modelo de datos estáticos del subsistema parcial TS•

• marcador para un elemento arbitrario pero fijado

REIVINDICACIONES

1. Método de ejecución de un motor de gemelos virtuales para el control de un sistema de fabricación distribuido de eventos discretos en tiempo real, en el que el motor de gemelos virtuales tiene instalado al menos un núcleo de software de modelado ejecutable que se prepara mediante

generar una representación de metamodelo para al menos un subsistema de un sistema de fabricación distribuido de eventos discretos seleccionando a partir de un metamodelo un tipo de metaobjeto (MO•) que hace referencia a al menos un modelo de estados (Σi• i ∈ {1, ...,s}) para el modelado del al menos un subsistema del sistema de fabricación distribuido de eventos discretos, en el que al menos un estado (zi•k) en el al menos un modelo de estados seleccionado (Σi• i ∈ {1, ...,s}) se describe con un metamodelo que es un conjunto de estados parciales (zi•k,p) usando diferentes niveles de abstracción de datos según un conjunto de metacaracterísticas que caracterizan un estado en un dominio de metadescripción, un conjunto de características objetivo que tienen un intervalo discreto o continuo de valores que caracterizan un objetivo para un estado considerado, y un conjunto de características reales que tienen un intervalo discreto o continuo de valores que caracterizan una constelación real de un estado considerado;

dividir el sistema de fabricación distribuido de eventos discretos en al menos dos agrupaciones de subsistemas que comprenden, cada una, al menos un subsistema del sistema de fabricación distribuido de eventos discretos que tiene la misma representación de metamodelo;

definir una topología de enrutamiento para el intercambio de mensajes de flujo continuo entre agrupaciones de subsistemas;

modelar al menos una agrupación de uso de subsistema para al menos un subsistema asignado a una agrupación de subsistemas del sistema de fabricación distribuido de eventos discretos; y

transferir el resultado de modelado a al menos un núcleo de software de modelado ejecutable para ejecutar la topología de enrutamiento y al menos un modelo de uso de subsistema en relación con cada uso de subsistema de la agrupación de subsistemas, en el que el al menos un núcleo de software de modelado ejecutable se configura mediante parametrización de software ejecutable disponible a partir de una biblioteca de software;

reduciendo el método de ejecución del motor de gemelos virtuales la carga global para el intercambio de información durante el control de procedimiento en tiempo real del sistema de fabricación distribuido discreto uniforme global, comprendiendo el método:

una etapa de ejecutar en tiempo real al menos un modelo de uso de subsistema en relación con la agrupación de subsistemas mediante la ejecución de un núcleo de software de modelado relacionado que modela el al menos un modelo de uso de subsistema;

una etapa de hacer funcionar al menos un gemelo digital con respecto al al menos un modelo de uso de subsistema como homólogo virtual de un objeto de procedimiento relacionado que se hace funcionar en el sistema de fabricación distribuido discreto uniforme y que tiene una funcionalidad correspondiente al modelo de uso de subsistema; en el que

la etapa de hacer funcionar el al menos un gemelo digital se ejecuta de una manera pasiva mediante acceso en tiempo real al núcleo de software de modelado que modela el modelo de uso de subsistema del al menos un gemelo digital; y

una etapa de controlar el intercambio de datos hacia/desde el motor de gemelos virtuales clasificando datos de comunicación según diferentes niveles de abstracción al menos en flujo de datos de control y/o flujo de datos de uso.

2. Método según la reivindicación 1, que comprende una etapa de inyectar al menos un elemento en la máquina de estados virtuales, elemento que se selecciona de un grupo que comprende:

información de modelado que refleja al menos un tipo de metaobjeto (MO•) en el motor de gemelos virtuales;

un conjunto de modelos de estados (Σij) que comprende al menos un modelo de estados al que hace referencia el tipo de metaobjeto (MO•) en el motor de gemelos virtuales;

al menos una perspectiva que representa una vista en el al menos un modelo de estados inyectado en el motor de gemelos virtuales;

diferentes niveles de abstracción de datos según metadatos, datos objetivo, y datos reales para cada modelo de estados y variables de estado relacionadas en el motor de gemelos virtuales;

código de software que implementa transiciones de estados externas y/o internas en la máquina de gemelos virtuales; y

al menos una agrupación de uso de subsistema con un tipo de metamodelo predeterminado.

3. Método según la reivindicación 1 ó 2, que comprende una etapa de inicializar al menos un gemelo digital en relación con una agrupación de uso de subsistema, en el que el al menos un gemelo digital sirve como homólogo digital para un objeto de procedimiento del mundo real correspondiente.

4. Método según la reivindicación 3, en el que la etapa de inicializar al menos un gemelo digital se ejecuta antes del inicio del motor de gemelos virtuales o durante el funcionamiento del motor de gemelos virtuales para control de procedimiento en tiempo real del sistema distribuido de eventos discretos.

5. Método según la reivindicación 1, que comprende una etapa de controlar el flujo de datos de control mediante evaluación de al menos un vector de estado que representa al menos un evento desencadenante para al menos una transición de estado de un gemelo digital.

6. Método según la reivindicación 1 ó 5, que comprende una etapa de controlar el flujo de datos de control generando al menos un mensaje de metaflujo en relación con una transición de estado de un gemelo digital para el intercambio de datos de control relacionados entre diferentes agrupaciones de subsistemas.

7. Método según una de las reivindicaciones 1, 5 ó 6, que comprende una etapa de controlar el flujo de datos de uso evaluando al menos un vector de condición que comprende al menos un valor real y/o al menos un valor objetivo en asociación con la al menos una transición de estado del al menos un gemelo digital para la evaluación de una condición de transición relacionada.

8. Método según una de las reivindicaciones 1 ó 5 a 7, que comprende una etapa de controlar el flujo de datos de uso ejecutando al menos una etapa seleccionada de un grupo que comprende

aplicar al menos una función de transformación a datos que representan un estado inicial antes de una transición de estado para generar datos que representan un estado objetivo después de la transición de estado;

actualizar al menos una información de contexto en relación con la transición de estado; y

procesar un resultado de transición de estado para la generación de al menos un mensaje de metaflujo para la actualización de al menos un modelo de estados en una agrupación de uso de subsistema que es externa a la agrupación de uso de subsistema del gemelo digital que ejecuta la transición de estado.

9. Motor de gemelos virtuales para el control de un sistema de fabricación distribuido discreto uniforme en tiempo real, en el que el motor de gemelos virtuales tiene instalado al menos un núcleo de software de modelado ejecutable preparado mediante

generar una representación de metamodelo para al menos un subsistema de un sistema de fabricación distribuido de eventos discretos seleccionando a partir de un metamodelo un tipo de metaobjeto (MO•) que hace referencia a al menos un modelo de estados ($\Sigma i\bullet\ i \in \{1, ...,s\}$) para el modelado del al menos un subsistema del sistema de fabricación distribuido de eventos discretos, en el que al menos un estado ($zi\bullet k$) en el al menos un modelo de estados seleccionado ($\Sigma 1\bullet\ i \in \{1, ...,s\}$) se describe con un metamodelo que es un conjunto de estados parciales ($zi\bullet k,p$) usando diferentes niveles de abstracción de datos según un conjunto de metacaracterísticas que caracterizan un estado en un dominio de metadescripción, un conjunto de características objetivo que tienen un intervalo discreto o continuo de valores que caracterizan un objetivo para un estado considerado, y un conjunto de características reales que tienen un intervalo discreto o continuo de valores que caracterizan una constelación real de un estado considerado;

dividir el sistema de fabricación distribuido de eventos discretos en al menos dos agrupaciones de subsistemas que comprenden, cada una, al menos un subsistema del sistema de fabricación distribuido de eventos discretos que tiene la misma representación de metamodelo;

definir una topología de enrutamiento para el intercambio de mensajes de flujo continuo entre agrupaciones de subsistemas;

modelar al menos una agrupación de uso de subsistema para al menos un subsistema asignado a una agrupación de subsistemas del sistema de fabricación distribuido de eventos discretos; y

transferir el resultado de modelado a al menos un núcleo de software de modelado ejecutable para ejecutar la topología de enrutamiento y al menos un modelo de uso de subsistema en relación con cada uso de subsistema de la agrupación de subsistemas, en el que el al menos un núcleo de software de modelado ejecutable se configura mediante parametrización de software ejecutable disponible a partir de una biblioteca de software;

estando el motor de gemelos virtuales configurado para reducir la carga global para el intercambio de información durante el control de procedimiento en tiempo real del sistema de fabricación distribuido de eventos discretos global, comprendiendo el motor de gemelos virtuales:

un módulo de procesamiento de modelo adaptado para ejecutar en tiempo real al menos un modelo de uso de subsistema en relación con la agrupación de subsistemas mediante la ejecución de un núcleo de software de modelado relacionado que modela el al menos un modelo de uso de subsistema; en el que

el módulo de procesamiento de modelo está adaptado para hacer funcionar al menos un gemelo digital con respecto al al menos un modelo de uso de subsistema como homólogo virtual de un objeto de procedimiento relacionado que se hace funcionar en el sistema de fabricación distribuido discreto uniforme y que tiene una funcionalidad correspondiente al modelo de uso de subsistema;

el módulo de procesamiento de modelo está adaptado para hacer funcionar el al menos un gemelo digital de una manera pasiva mediante acceso en tiempo real al núcleo de software de modelado que modela el modelo de uso de subsistema del al menos un gemelo digital; y

un módulo de control de flujo de datos está adaptado para controlar el intercambio de datos hacia/desde el motor de gemelos virtuales clasificando datos de comunicación según diferentes niveles de abstracción al menos en flujo de datos de control y/o flujo de datos de uso.

10. Motor de gemelos virtuales según la reivindicación 9, que comprende una interfaz de motor de modelado adaptada para inyectar al menos un elemento en la máquina de estados virtuales, elemento que se selecciona de un grupo que comprende:

información de modelado que refleja al menos un tipo de metaobjeto (MO•) en el motor de gemelos virtuales;

un conjunto de modelos de estados (Σij) que comprende al menos un modelo de estados al que hace referencia el tipo de metaobjeto (MO•) en el motor de gemelos virtuales;

al menos una perspectiva que representa una vista en el al menos un modelo de estados inyectado en el motor de gemelos virtuales;

diferentes niveles de abstracción de datos según metadatos, datos objetivo, y datos reales para cada modelo de estados y variables de estado relacionadas en el motor de gemelos virtuales;

código de software que implementa transiciones de estados externas y/o internas en la máquina de gemelos virtuales; y

al menos una agrupación de uso de subsistema con un tipo de metamodelo predeterminado.

11. Motor de gemelos virtuales según la reivindicación 9 ó 10, que comprende un módulo de control de modelado adaptado para inyectar al menos un gemelo digital en relación con una agrupación de uso de subsistema, en el que el al menos un gemelo digital sirve como homólogo digital para un objeto de procedimiento del mundo real correspondiente.

12. Motor de gemelos virtuales según la reivindicación 11, en el que módulo de control de modelado está adaptado para inicializar el al menos un gemelo digital antes del inicio del motor de gemelos digitales o en tiempo real durante el funcionamiento del motor de gemelos virtuales.

13. Motor de gemelos virtuales según una de las reivindicaciones 9 a 12, que comprende una interfaz de motor de servicio adaptada para inyectar código de software que implementa transiciones de estados externas y/o internas en el motor de gemelos virtuales.

14. Motor de gemelos virtuales según la reivindicación 9, en el que el módulo de control de flujo de datos está adaptado para controlar un módulo de evaluación de vector de estado que funciona en el flujo de datos de

control para evaluar al menos un vector de estado que representa al menos un evento desencadenante para al menos una transición de estado de un gemelo digital.

15. Motor de gemelos virtuales según la reivindicación 9 ó 14, en el que el módulo de control de flujo de datos está adaptado para controlar un módulo de evaluación de vector de estado que funciona en el flujo de datos de control para generar al menos un mensaje de metaflujo en relación con una transición de estado de un gemelo digital para el intercambio de datos de control relacionados entre diferentes agrupaciones de subsistemas.

16. Motor de gemelos virtuales según una de las reivindicaciones 9, 14 ó 15, en el que el módulo de control de flujo de datos está adaptado para controlar un módulo de evaluación de vector de condición que funciona en el flujo de datos de uso para evaluar al menos un vector de condición que comprende al menos un valor real y/o al menos un valor objetivo en asociación con la al menos una transición de estado del al menos un gemelo digital para la evaluación de una condición de transición relacionada.

17. Motor de gemelos virtuales según una de las reivindicaciones 9 ó 14 a 16, en el que el módulo de control de flujo de datos está adaptado para controlar un módulo de transformación que funciona en el flujo de datos de uso para aplicar al menos una función de transformación a datos que representan un estado inicial antes de una transición de estado para generar datos que representan un estado objetivo después de la transición de estado.

18. Motor de gemelos virtuales según una de las reivindicaciones 9 ó 14 a 17, en el que el módulo de control de flujo de datos está adaptado para controlar un módulo de actualización que funciona en el flujo de datos de uso para actualizar al menos una información de contexto en relación con la transición de estado.

19. Motor de gemelos virtuales según una de las reivindicaciones 9 ó 14 a 18, en el que el módulo de control de flujo de datos está adaptado para controlar un módulo de generación de mensaje que funciona en el flujo de datos de control para generar al menos un mensaje de metaflujo para la actualización de al menos un modelo de estados en una agrupación de uso de subsistema externa a la agrupación de uso de subsistema del gemelo digital que ejecuta la transición de estado.

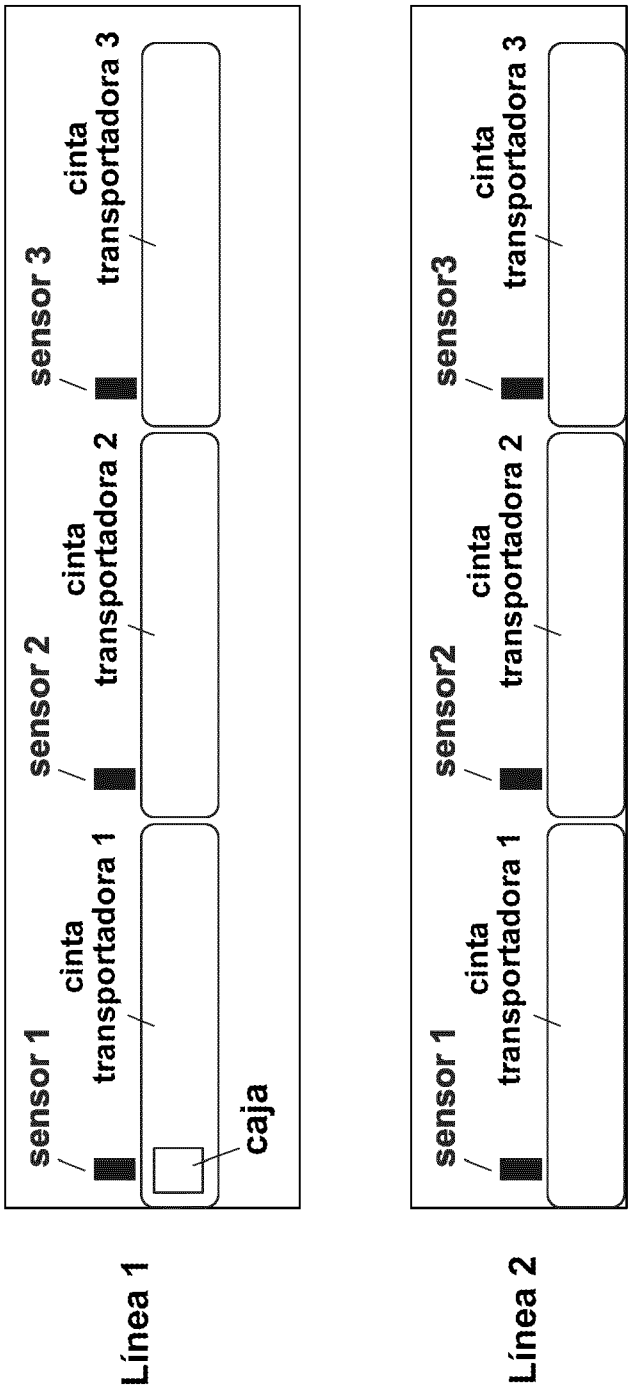

sensor 1
cinta transportadora 1
sensor 2
cinta transportadora 2
sensor 3
cinta transportadora 3
caja
Línea 1
sensor1
cinta transportadora 1
sensor2
cinta transportadora 2
sensor3
cinta transportadora 3
Línea 2

FIG. 1

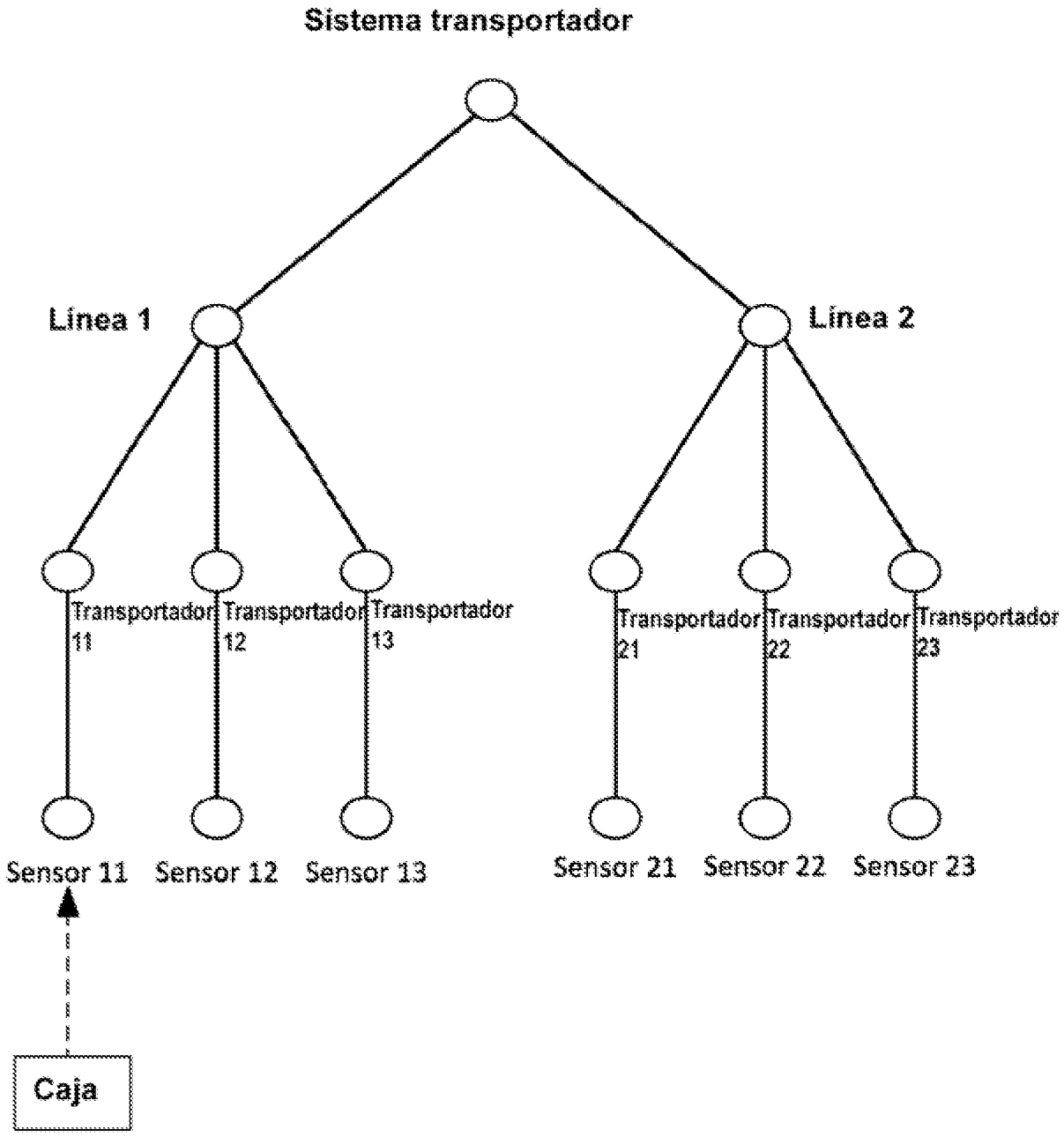

Sistema transportador
Línea 1
Línea 2
Transportador 11
Transportador 12
Transportador 13
Transportador 21
Transportador 22
Transportador 23
Sensor 11
Sensor 12
Sensor 13
Sensor 21
Sensor 22
Sensor 23
Caja

FIG. 2

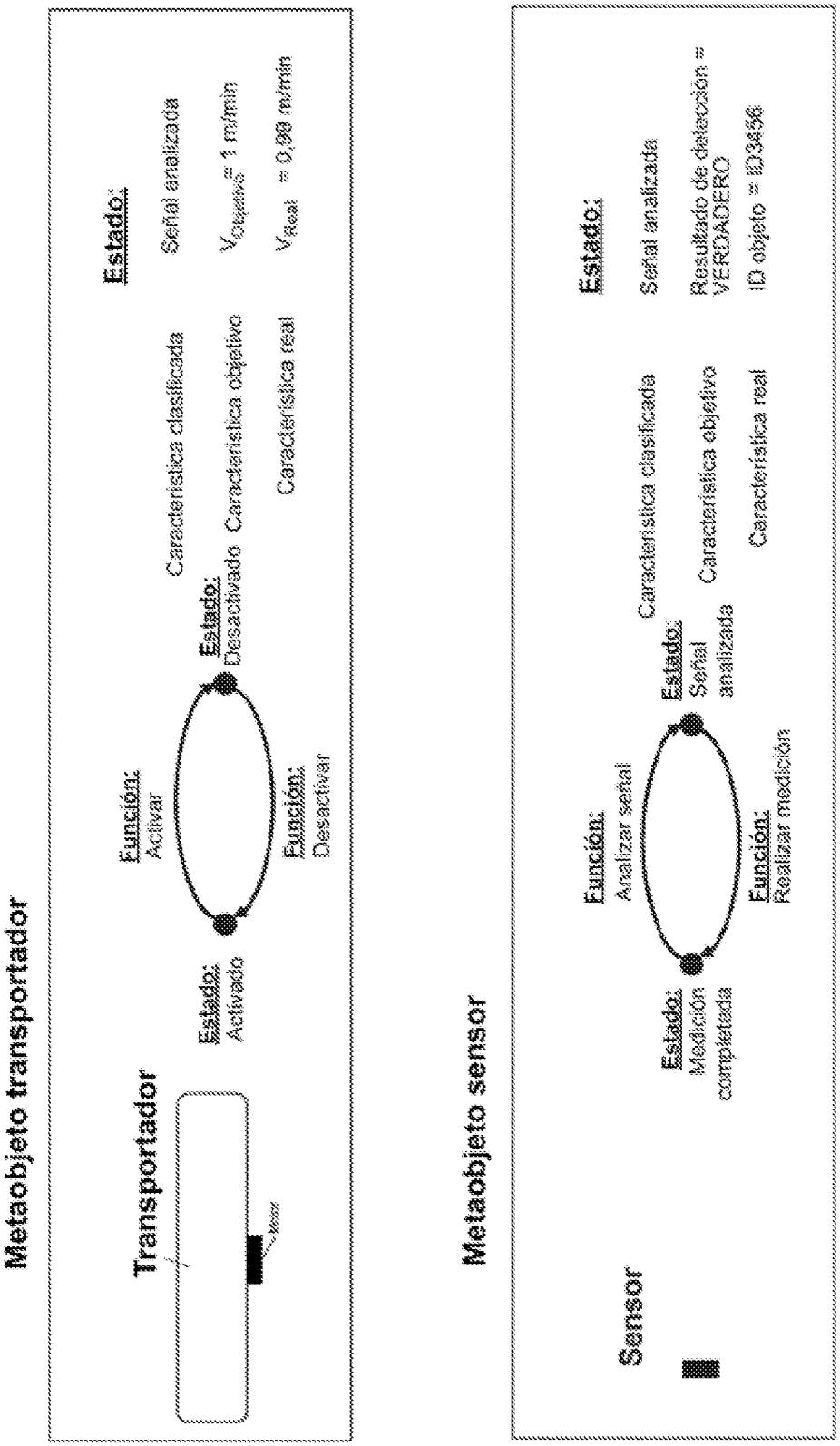
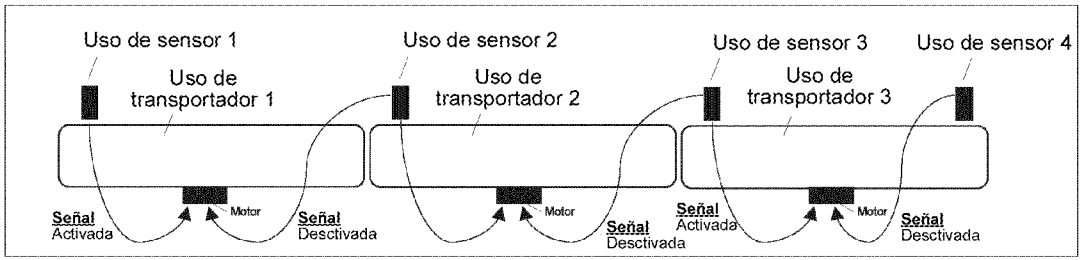

Metaobjeto transportador
Transportador
Estado:
Señal analizada
Característica clasificada
Estado:
Activado
Función:
Activar
Estado:
Desactivado
Característica objetivo
Función:
Desactivar
Característica real
$V_{Objetivo}$ = 1 m/min
V_{Real} = 0,98 m/min
Metaobjeto sensor
Sensor
Estado:
Señal analizada
Característica clasificada
Función:
Analizar señal
Estado:
Señal analizada
Característica objetivo
Estado:
Medición completada
Función:
Realizar medición
Característica real
Resultado de detección = VERDADERO
ID objeto = ID3456

FIG. 3

Modelo de uso "línea de transportador"



Vector de estado externo

Uso de transportador 1, función activar

Vector de estado externo

[Uso de sensor 1: estado: señal analizada]

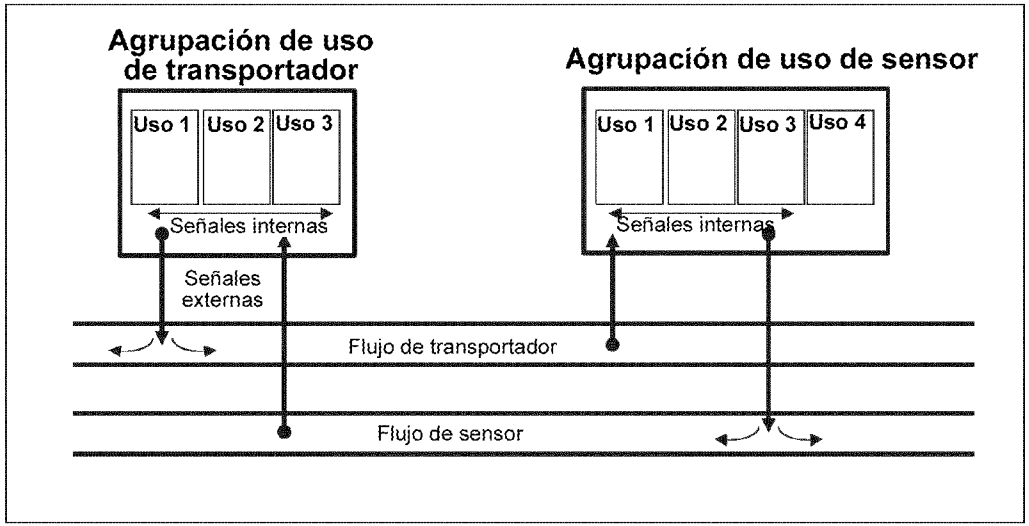


FIG. 4

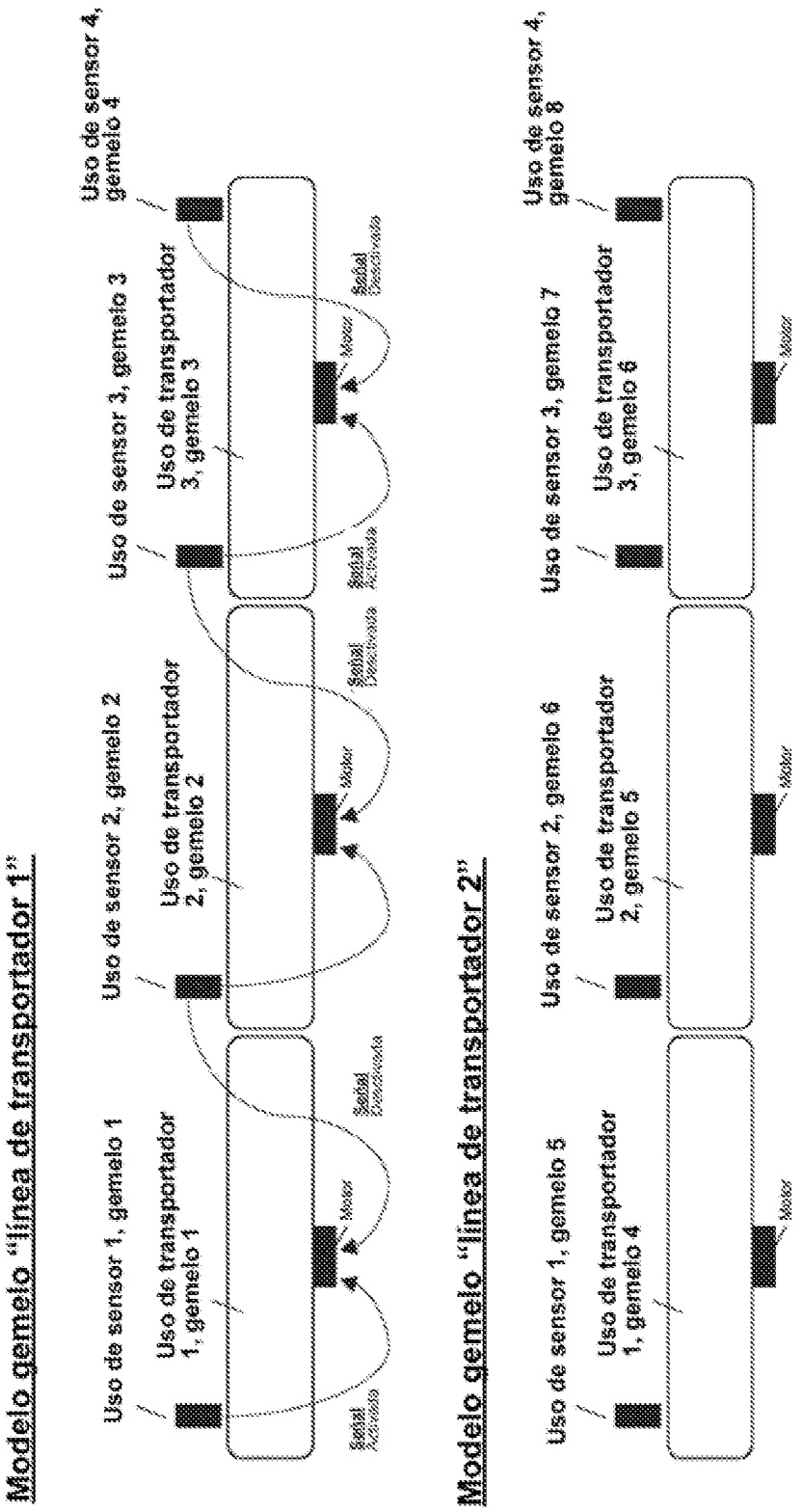

Modelo gemelo "línea de transportador 1"
Uso de sensor 1, gemelo 1
Uso de transportador 1, gemelo 1
Uso de sensor 2, gemelo 2
Uso de transportador 2, gemelo 2
Uso de sensor 3, gemelo 3
Uso de transportador 3, gemelo 3
Uso de sensor 4, gemelo 4
Motor
Señal
Modelo gemelo "línea de transportador 2"
Uso de sensor 1, gemelo 5
Uso de transportador 1, gemelo 4
Uso de sensor 2, gemelo 6
Uso de transportador 2, gemelo 5
Uso de sensor 3, gemelo 7
Uso de transportador 3, gemelo 6
Uso de sensor 4, gemelo 8
Motor

FIG. 5

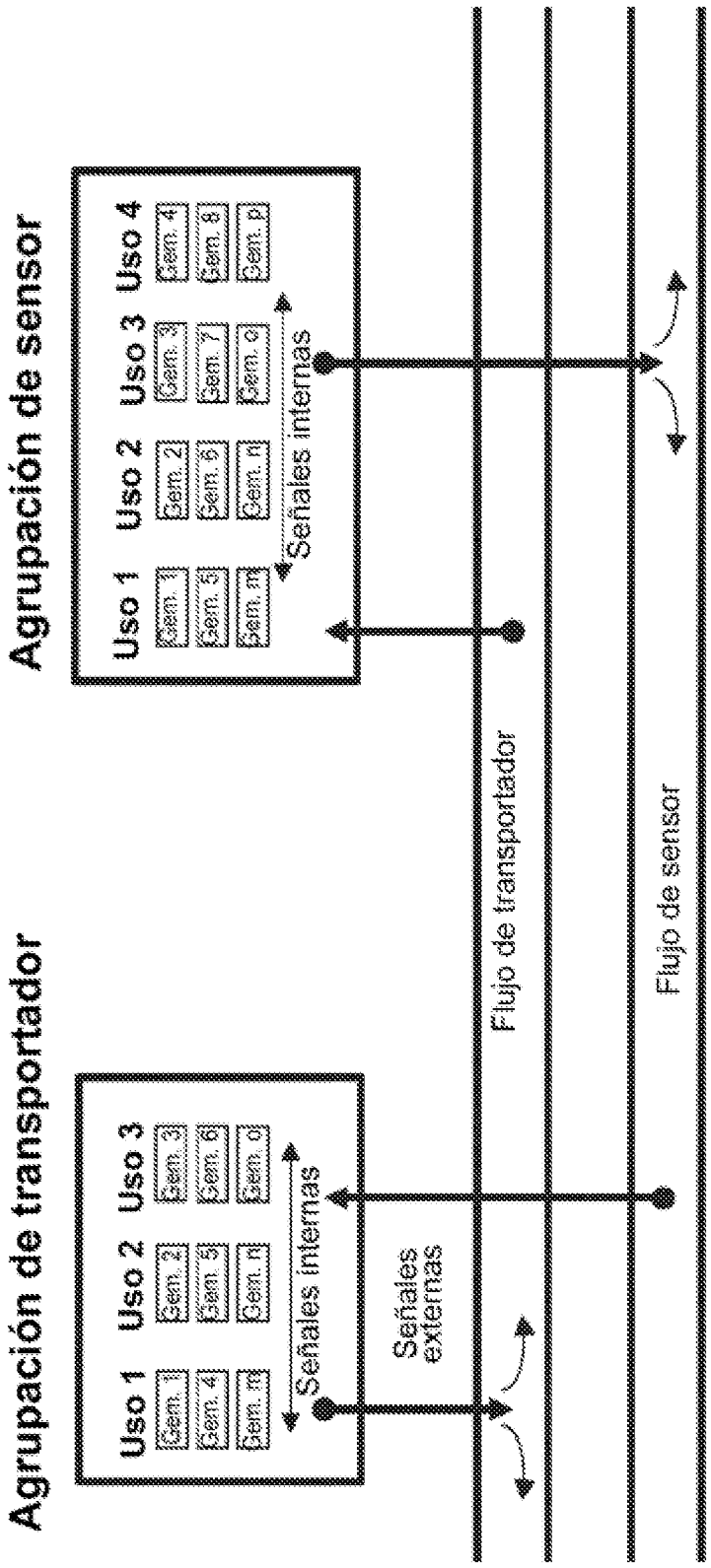

Agrupación de transportador
Uso 1
Uso 2
Uso 3
Sem. 1
Sem. 2
Sem. 3
Sem. 4
Sem. 5
Sem. 6
Sem. m
Sem. n
Sem. o
Señales internas
Señales externas
Flujo de transportador
Flujo de sensor
Agrupación de sensor
Uso 1
Uso 2
Uso 3
Uso 4
Sem. 1
Sem. 2
Sem. 3
Sem. 4
Sem. 5
Sem. 6
Sem. 7
Sem. 8
Sem. m
Sem. n
Sem. o
Sem. p
Señales internas

FIG. 6

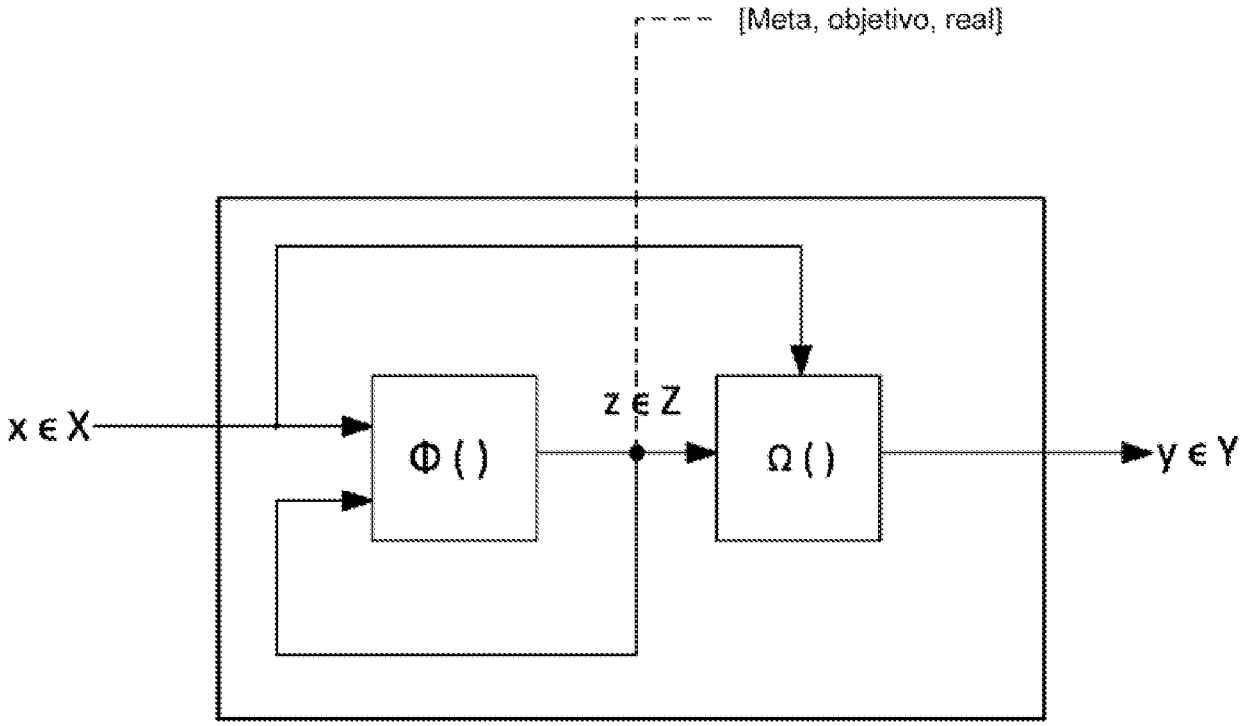


FIG. 7

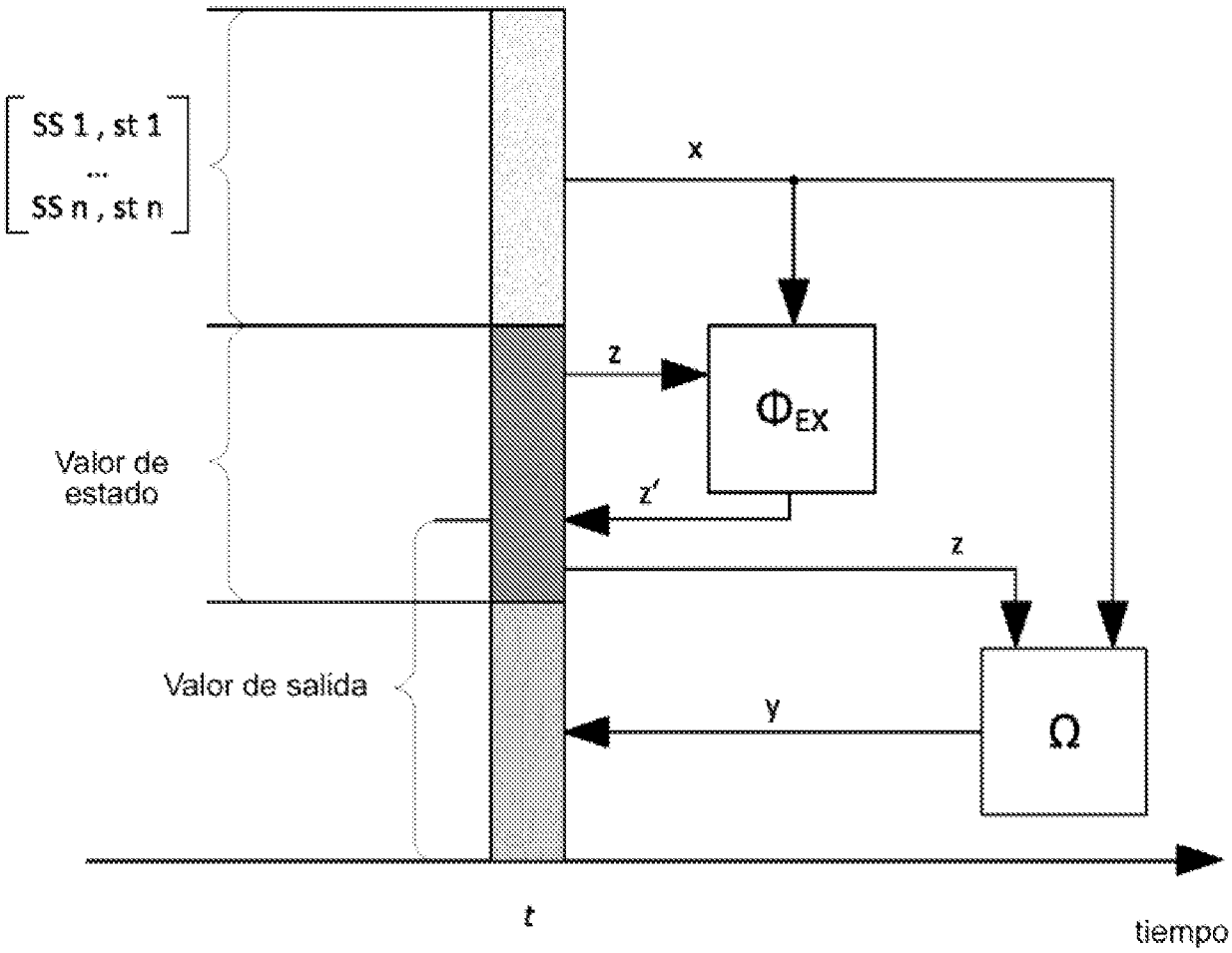

SS 1 , st 1
...
SS n , st n
x
z
Φ_{EX}
z'
z
Valor de estado
Valor de salida
y
Ω
t
tiempo

FIG. 8

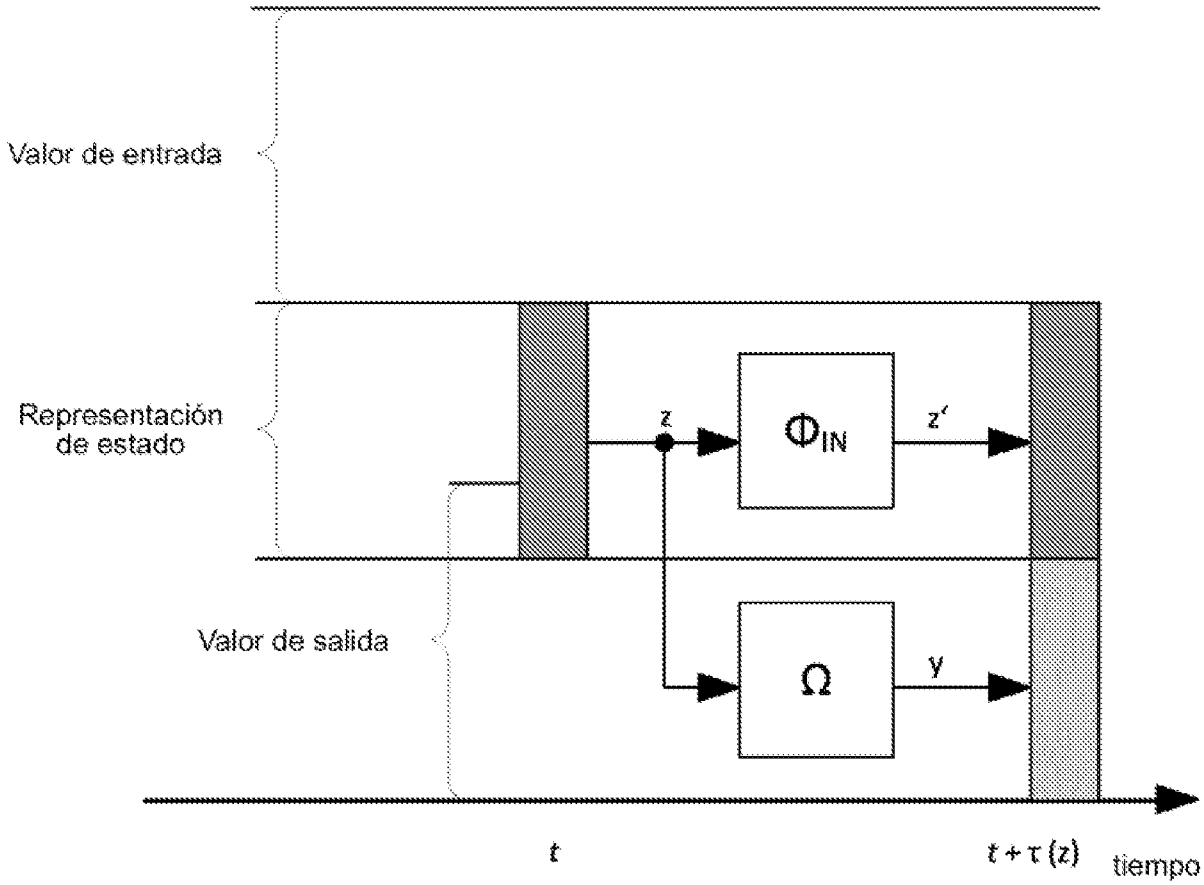

Valor de entrada
Representación de estado
Valor de salida
z
Φ_{IN}
z'
Ω
y
t
t + τ (z)
tiempo

FIG. 9

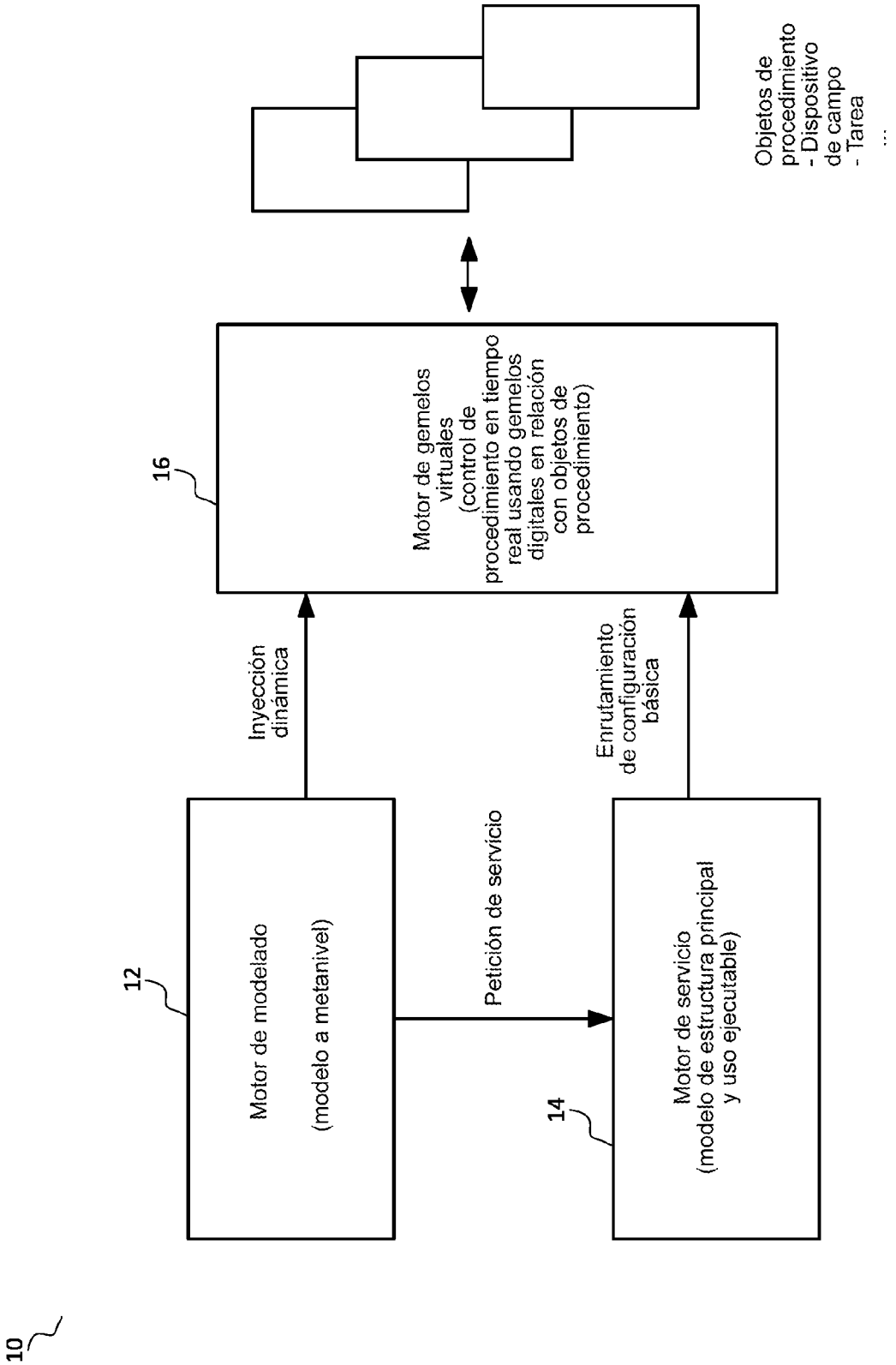

10
12
Motor de modelado
(modelo a metanivel)
Inyección dinámica
Petición de servicio
Enrutamiento de configuración básica
14
Motor de servicio
(modelo de estructura principal y uso ejecutable)
16
Motor de gemelos virtuales
(control de procedimiento en tiempo real usando gemelos digitales en relación con objetos de procedimiento)
Objetos de procedimiento
- Dispositivo de campo
- Tarea
...

FIG. 10

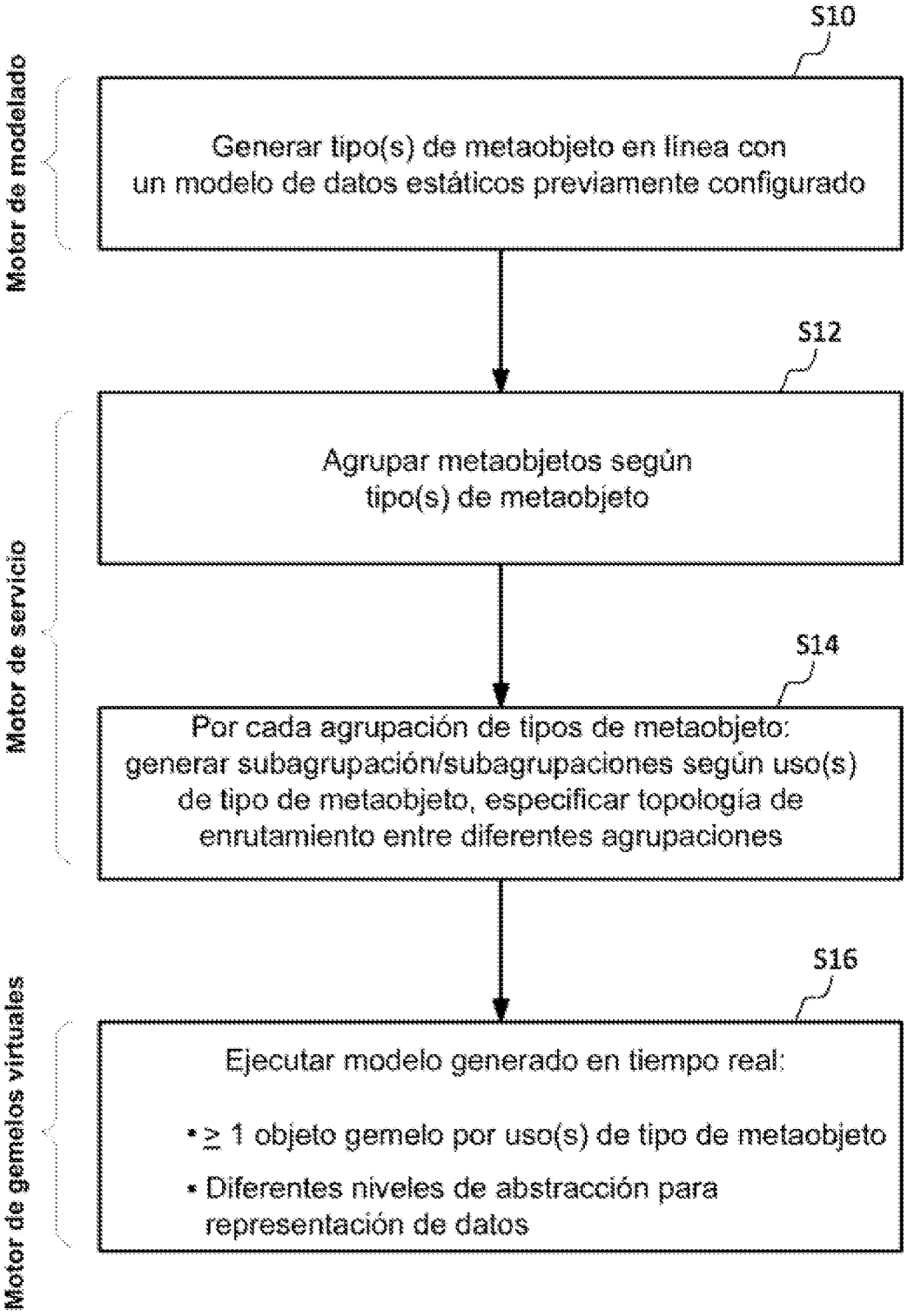

Motor de modelado
S10
Generar tipo(s) de metaobjeto en línea con un modelo de datos estáticos previamente configurado
Motor de servicio
S12
Agrupar metaobjetos según tipo(s) de metaobjeto
S14
Por cada agrupación de tipos de metaobjeto: generar subagrupación/subagrupaciones según uso(s) de tipo de metaobjeto, especificar topología de enrutamiento entre diferentes agrupaciones
Motor de gemelos virtuales
S16
Ejecutar modelo generado en tiempo real:
• ≥ 1 objeto gemelo por uso(s) de tipo de metaobjeto
• Diferentes niveles de abstracción para representación de datos

FIG. 11

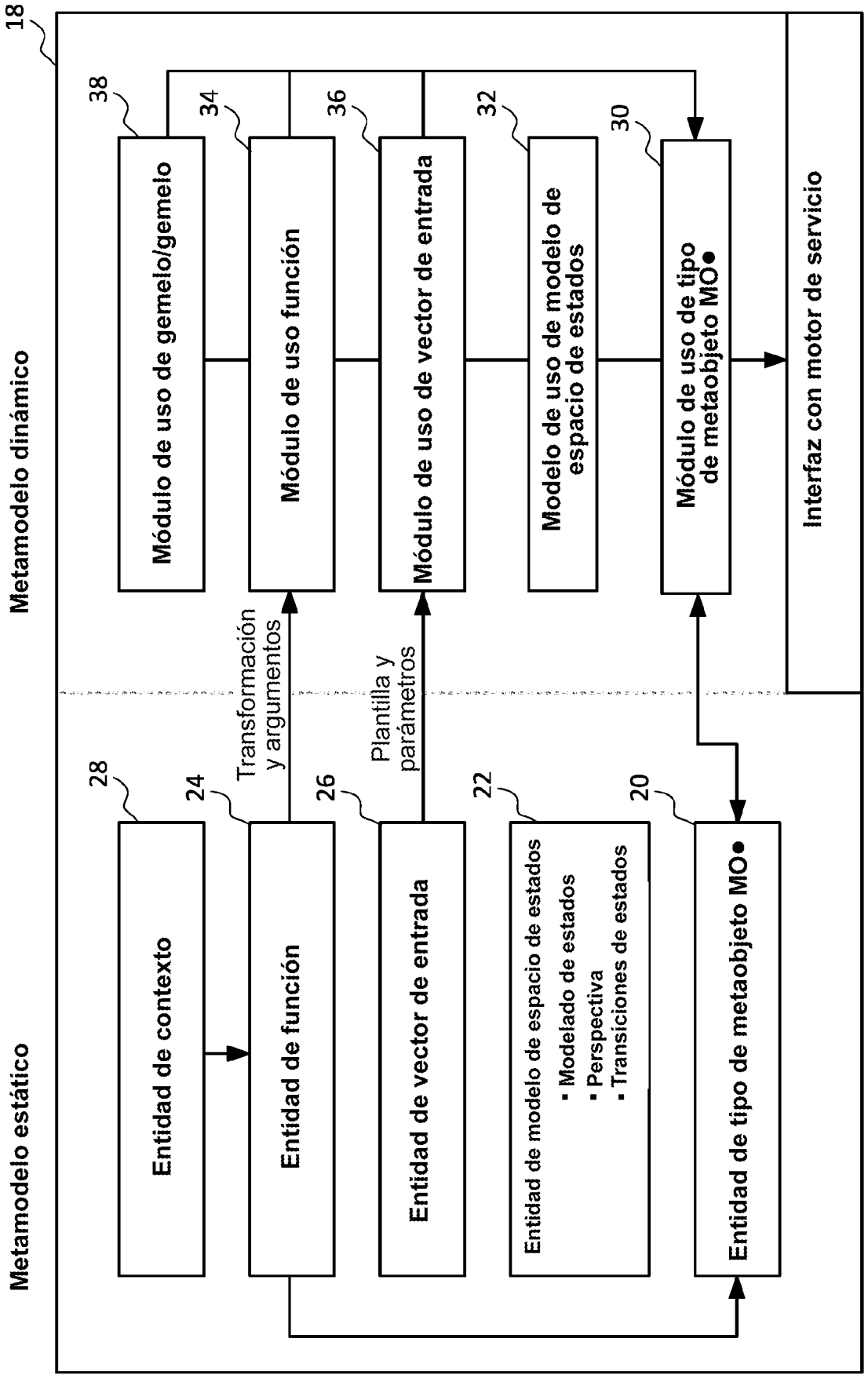

18
Metamodelo dinámico
Metamodelo estático
38
34
36
32
30
Módulo de uso de gemelo/gemelo
Módulo de uso función
Módulo de uso de vector de entrada
Modelo de uso de modelo de espacio de estados
Módulo de uso de tipo de metaobjeto MO●
Interfaz con motor de servicio
Transformación y argumentos
Plantilla y parámetros
28
24
26
22
20
Entidad de contexto
Entidad de función
Entidad de vector de entrada
Entidad de modelo de espacio de estados
▪ Modelado de estados
▪ Perspectiva
▪ Transiciones de estados
Entidad de tipo de metaobjeto MO●

FIG. 12

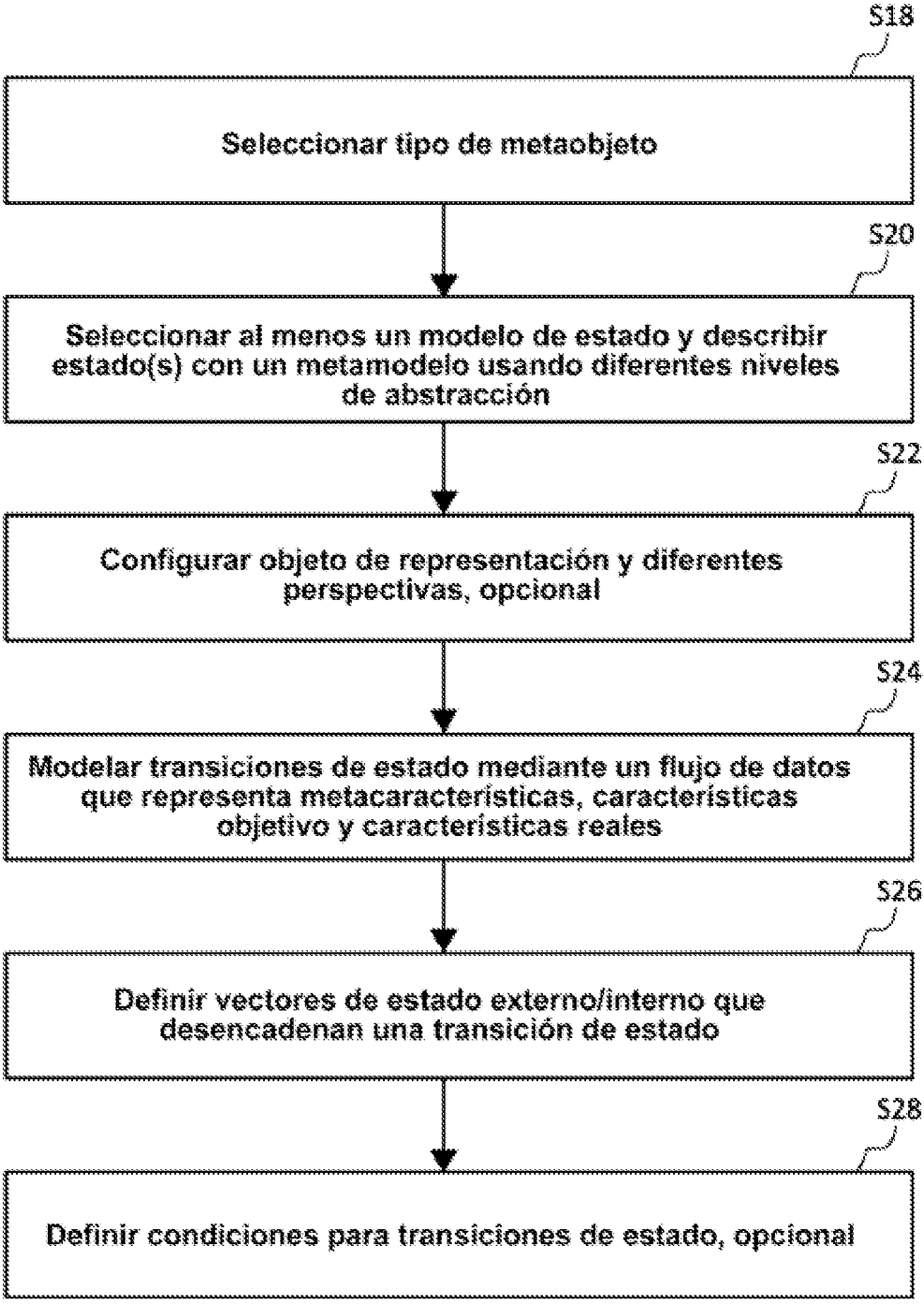

S18
Seleccionar tipo de metaobjeto
S20
Seleccionar al menos un modelo de estado y describir estado(s) con un metamodelo usando diferentes niveles de abstracción
S22
Configurar objeto de representación y diferentes perspectivas, opcional
S24
Modelar transiciones de estado mediante un flujo de datos que representa metacaracterísticas, características objetivo y características reales
S26
Definir vectores de estado externo/interno que desencadenan una transición de estado
S28
Definir condiciones para transiciones de estado, opcional

FIG. 13

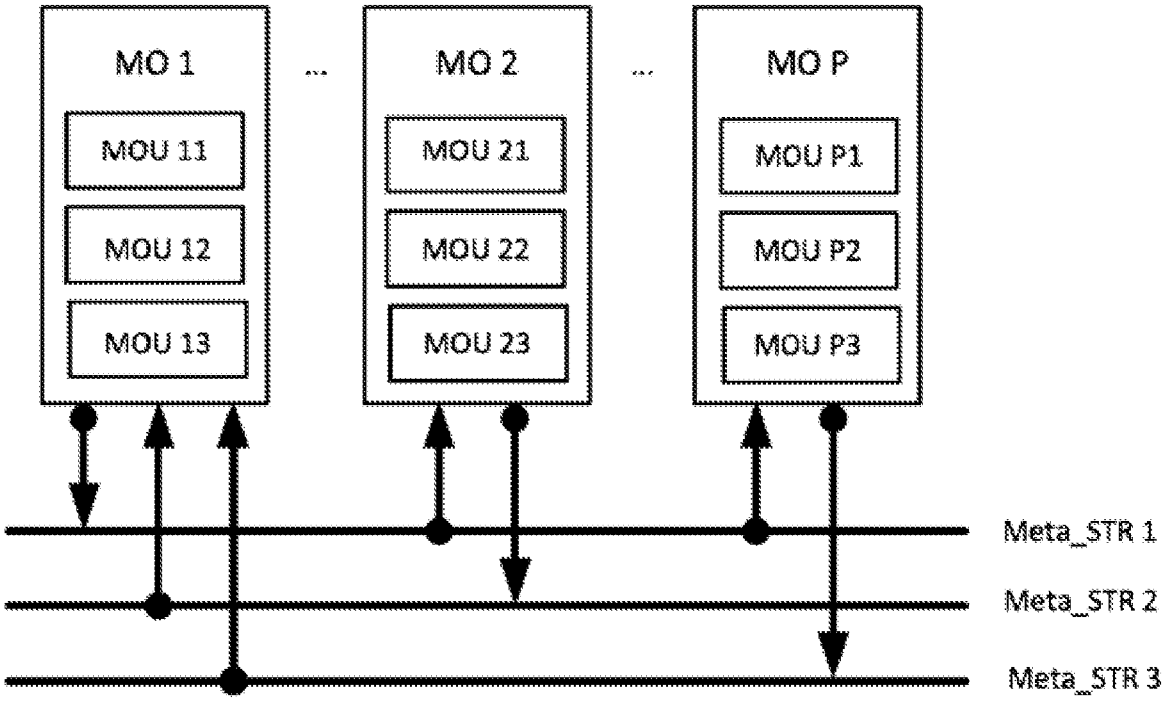

MO 1
...
MO 2
...
MO P
MOU 11
MOU 12
MOU 13
MOU 21
MOU 22
MOU 23
MOU P1
MOU P2
MOU P3
Meta_STR 1
Meta_STR 2
Meta_STR 3

FIG. 14

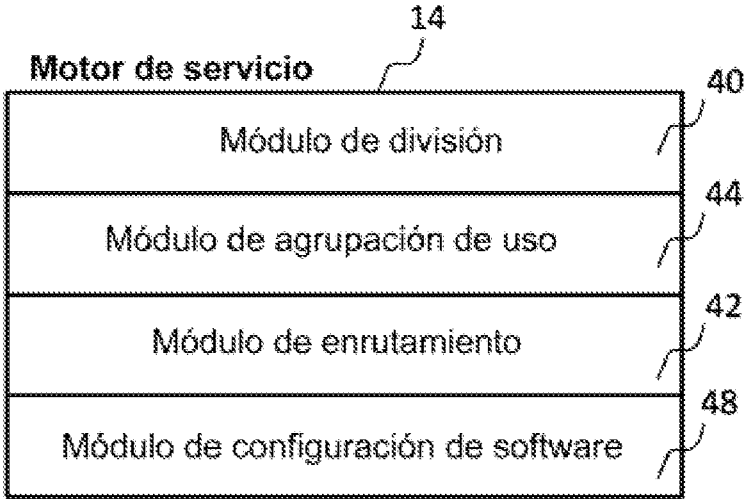

14
Motor de servicio
40
Módulo de división
44
Módulo de agrupación de uso
42
Módulo de enrutamiento
48
Módulo de configuración de software

FIG. 15

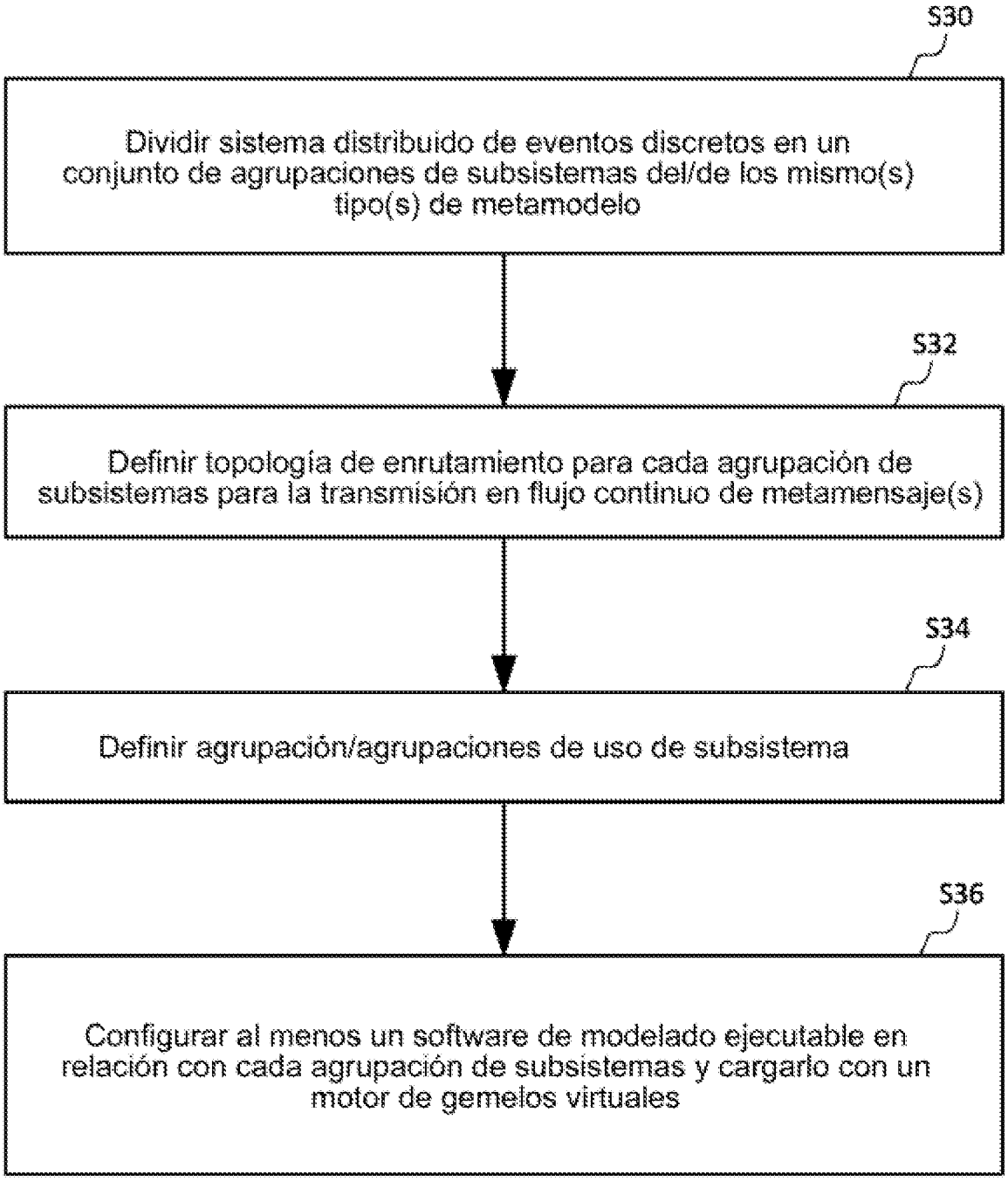

S30
Dividir sistema distribuido de eventos discretos en un conjunto de agrupaciones de subsistemas del/de los mismo(s) tipo(s) de metamodelo
S32
Definir topología de enrutamiento para cada agrupación de subsistemas para la transmisión en flujo continuo de metamensaje(s)
S34
Definir agrupación/agrupaciones de uso de subsistema
S36
Configurar al menos un software de modelado ejecutable en relación con cada agrupación de subsistemas y cargarlo con un motor de gemelos virtuales

FIG. 16

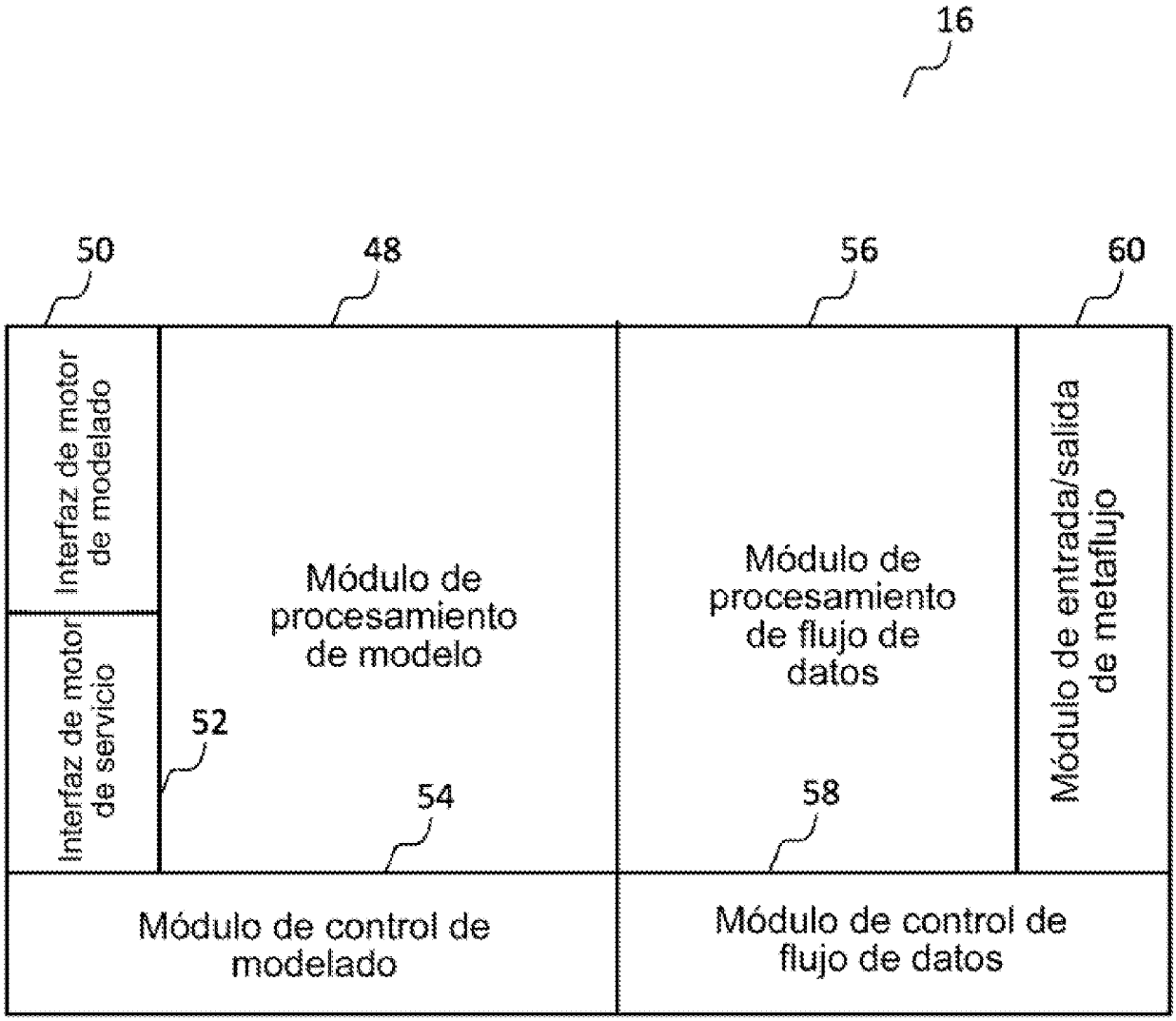

16
50
48
56
60
Interfaz de motor de modelado
Interfaz de motor de servicio
Módulo de procesamiento de modelo
52
54
Módulo de procesamiento de flujo de datos
58
Módulo de entrada/salida de metaflujo
Módulo de control de modelado
Módulo de control de flujo de datos

FIG. 17

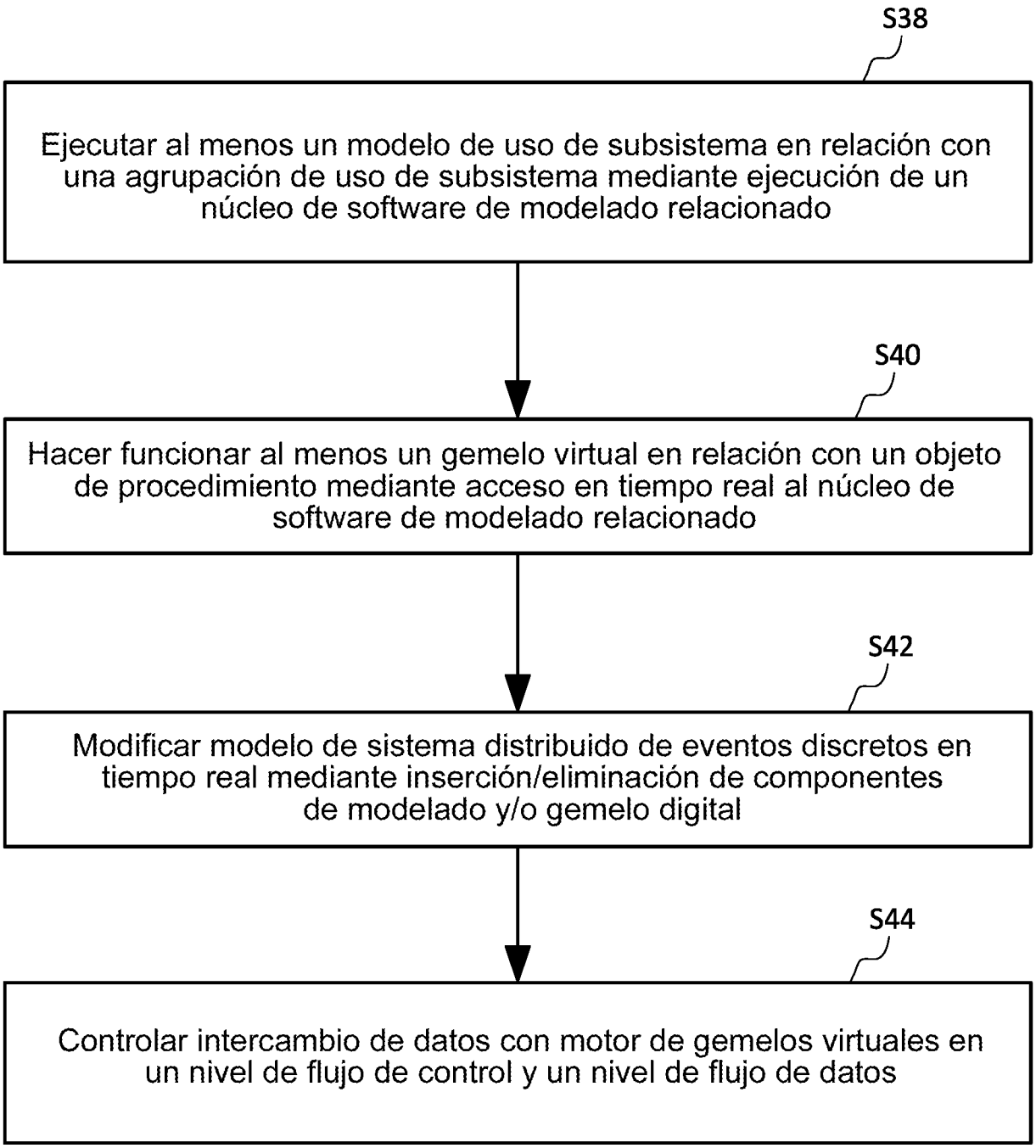

S38
Ejecutar al menos un modelo de uso de subsistema en relación con una agrupación de uso de subsistema mediante ejecución de un núcleo de software de modelado relacionado
S40
Hacer funcionar al menos un gemelo virtual en relación con un objeto de procedimiento mediante acceso en tiempo real al núcleo de software de modelado relacionado
S42
Modificar modelo de sistema distribuido de eventos discretos en tiempo real mediante inserción/eliminación de componentes de modelado y/o gemelo digital
S44
Controlar intercambio de datos con motor de gemelos virtuales en un nivel de flujo de control y un nivel de flujo de datos

FIG. 18

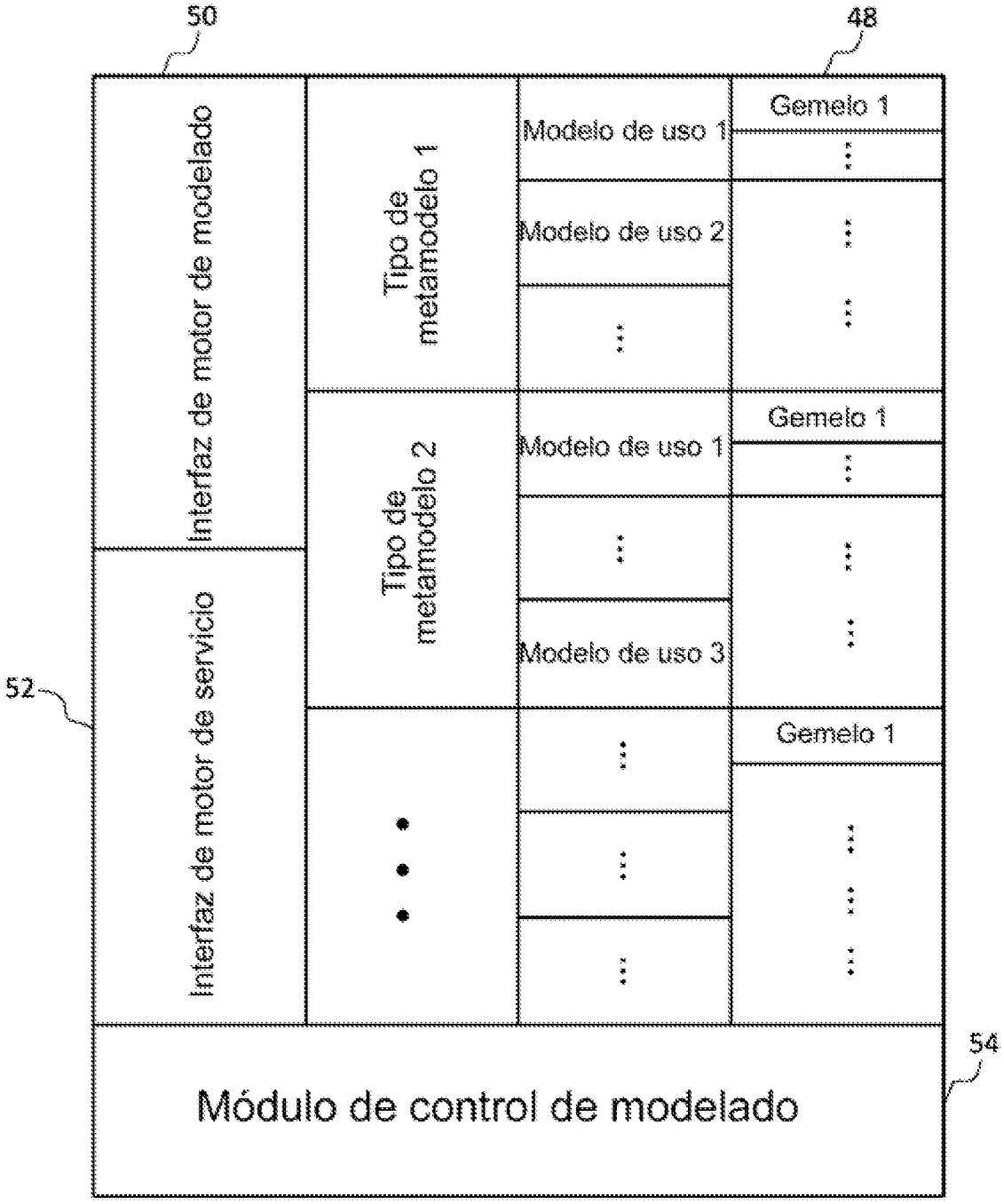

50
48
52
54
Interfaz de motor de modelado
Interfaz de motor de servicio
Tipo de metamodelo 1
Tipo de metamodelo 2
Modelo de uso 1
Modelo de uso 2
Modelo de uso 3
Gemelo 1
Módulo de control de modelado

FIG. 19

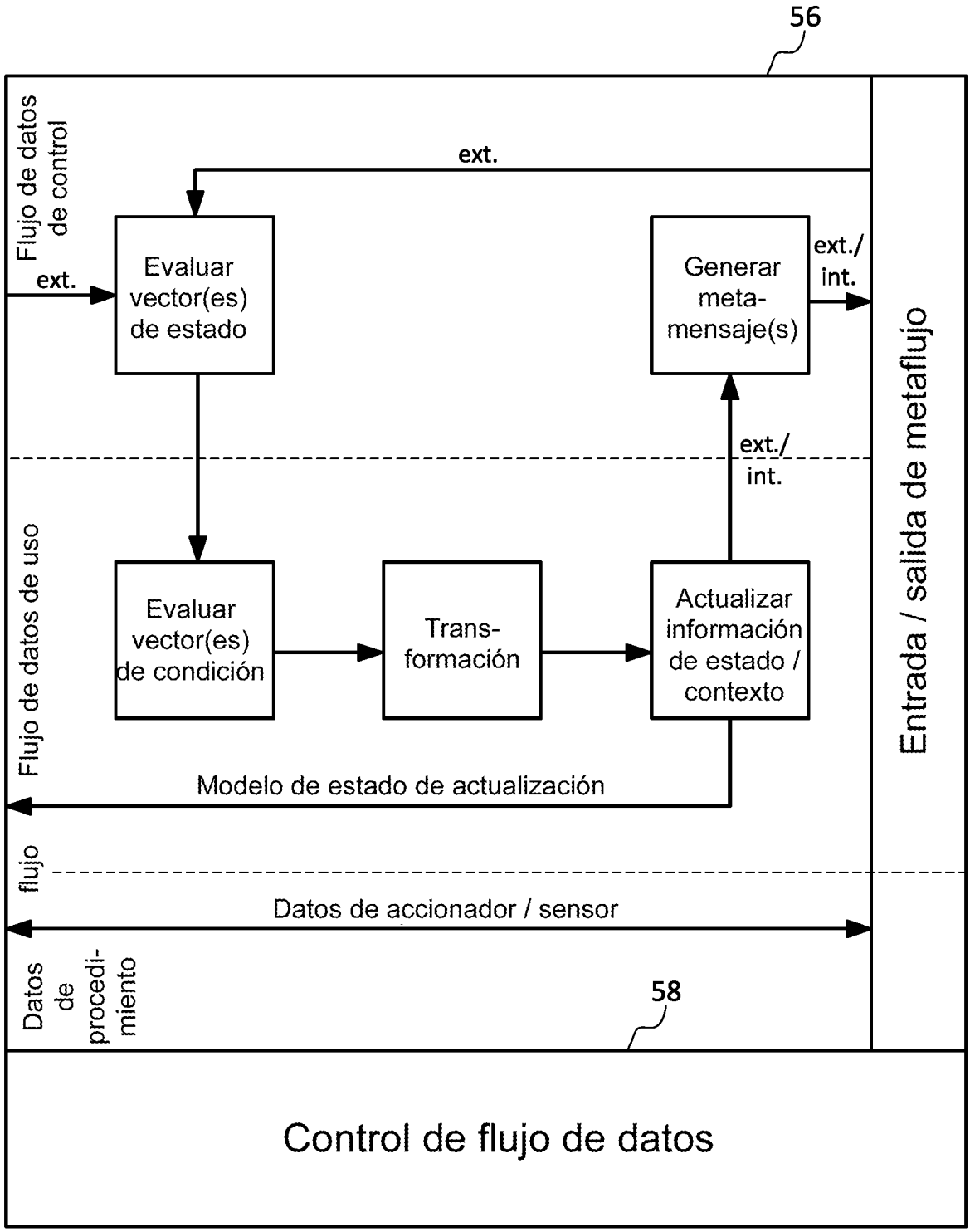

56
Flujo de datos de control
ext.
ext.
Evaluar vector(es) de estado
Generar meta-mensaje(s)
ext./ int.
ext./ int.
Flujo de datos de uso
Evaluar vector(es) de condición
Trans-formación
Actualizar información de estado / contexto
Modelo de estado de actualización
Entrada / salida de metaflujo
flujo
Datos de accionador / sensor
Datos de procedi-miento
58
Control de flujo de datos

FIG. 20

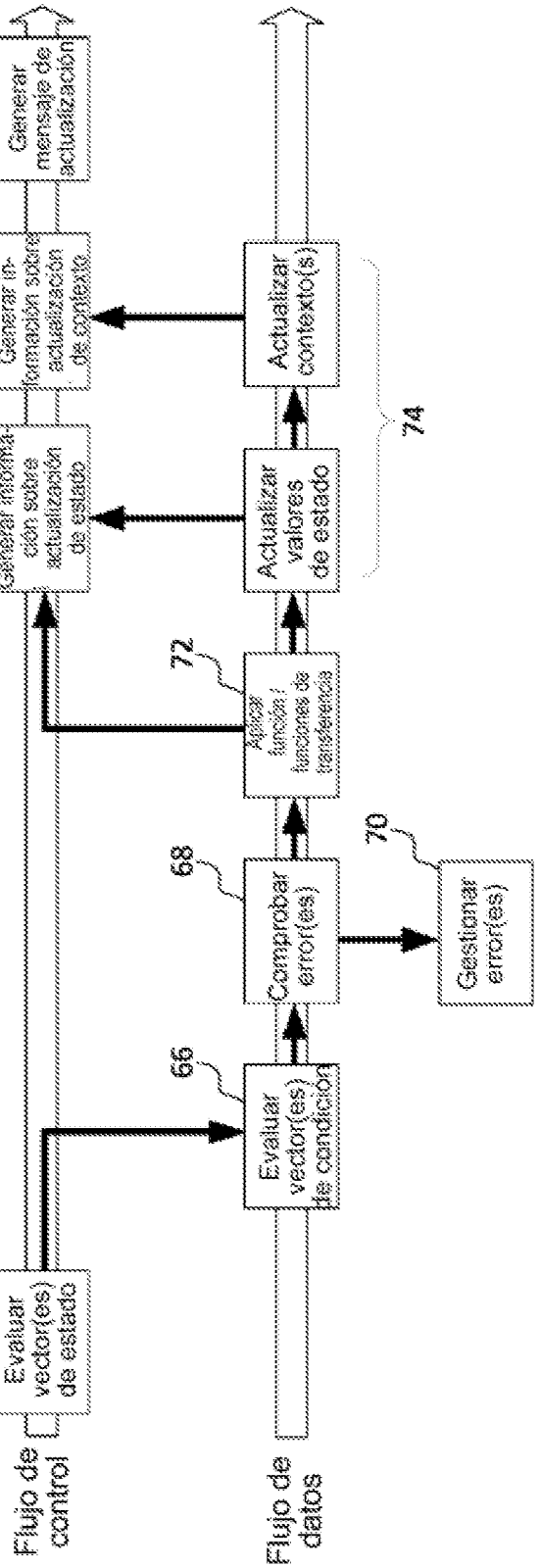
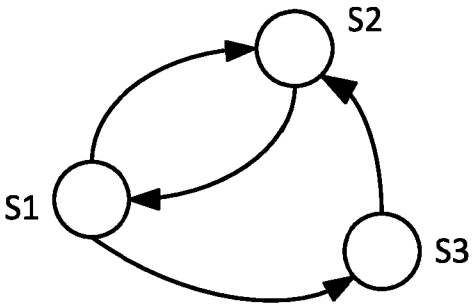

Flujo de control
Flujo de datos
62
Evaluar vector(es) de estado
66
Evaluar vector(es) de condición
68
Comprobar error(es)
70
Gestionar error(es)
72
Aplicar función / funciones de transferencia
Generar información sobre actualización de estado
Actualizar valores de estado
Generar información sobre actualización de contexto
Actualizar contexto(s)
Generar mensaje de actualización
64
74

FIG. 21



Contextos de gemelos

Gemelo 1: Gemelo 1 - Gemelo n = C1n

... ...

Gemelo i - Gemelo j = Cij

... ...

Gemelo n: Gemelo n - Gemelo k = Cnk

	C1n	Cij	Cnk
Uso 1-S1	Gemelo 1	Gemelo 1	Gemelo 1
Uso 2-S2	Gemelo 1	Gemelo 1	Gemelo 1
Uso 3-S3	Gemelo 1	Gemelo 1	Gemelo 1
Uso 4-S1	Gemelo 2	Gemelo 2	Gemelo 2
Uso 5-S2	Gemelo 2	Gemelo 2	Gemelo 2
Uso 6-S3	Gemelo 2	Gemelo 2	Gemelo 2
Uso 7-S1	Gemelo 3	Gemelo 3	Gemelo 3
Uso 8-S2	Gemelo 3	Gemelo 3	Gemelo 3
...	...	...	...
Uso n	Gemelo n	Gemelo n	Gemelo n

Cij

...

Gemelo i / Sm

...

Gemelo j / SM

...

= Uso i / Cij ; Uso j / Cij

FIG. 22

Uso de SM / MO	1	2	3
1	[SB 11]	[SB 12]	[SB 13]
2	[SB 21]	[SB 22]	[SB 23]
3	[SB 31]	...	...
...	...	...	...

FIG. 23

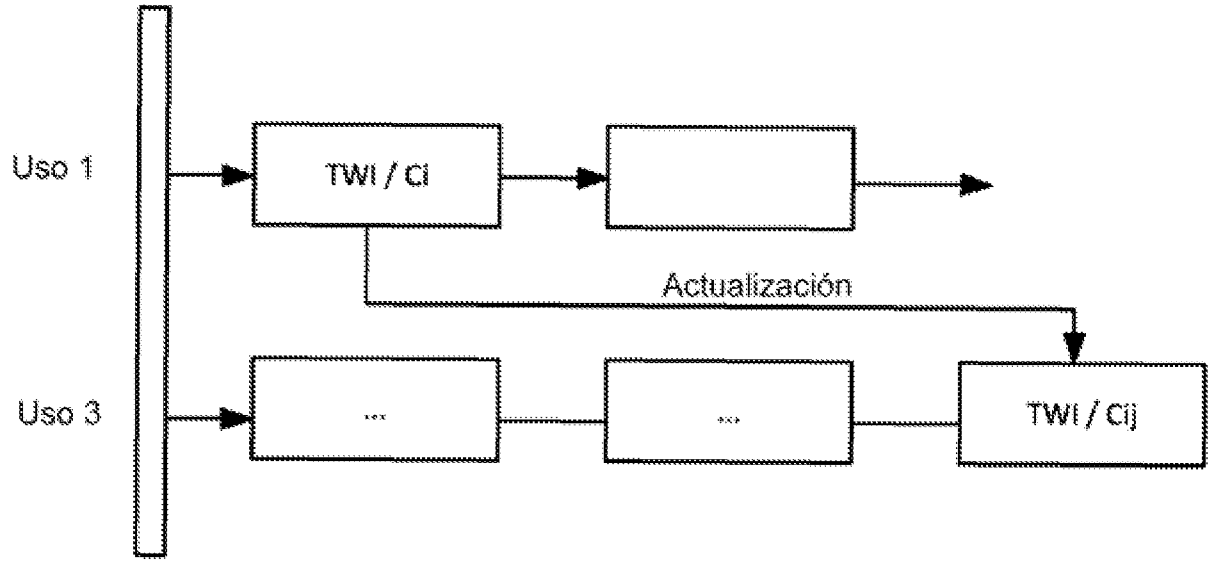


FIG. 24