Definición de una Ontología para la Representación de Comportamientos basados en Eventos

María Julia Blas[1], Silvio Gonnet[1],
Guido Tebes[2], Pablo Becker[2], Luis Olsina[2]

[1] INGAR, CONICET-UTN, Avellaneda 3657, Santa Fe, Argentina
{mariajuliablas,sgonnet}@santafe-conicet.gov.ar
[2] GIDIS_Web, Facultad de Ingeniería, UNLPam, General Pico, LP, Argentina
{guido_tebes,beckerp,olsinal}@ing.unlpam.edu.ar

Abstract. Este trabajo presenta los términos, propiedades, relaciones y axiomas de ParticularEventCO como un modelo semántico basado en la noción de eventos causados por los comportamientos de las entidades. La ontología propuesta se encuentra situada en el nivel "Core" de una arquitectura ontológica de cuatro capas llamada FCD-OntoArch (Foundational, Core, and Domain Ontological Architecture for Sciences), donde en el nivel fundacional se encuentra la ontología ThingFO. Luego, se detalla la forma en la cual los principales elementos del nivel fundacional han sido redefinidos en ParticularEventCO. Además, para ilustrar la utilidad de la propuesta, presentamos la instanciación de un ejemplo.

Keywords: Objetos y eventos, sistemas dinámicos de eventos discretos, fenómeno, modelo semántico, ontología.

1 Introducción

La definición de comportamientos dinámicos usando eventos es un área de interés en Modelado y Simulación (M&S). El M&S basado en eventos discretos parte de la noción de evento, el cual se define como un cambio en el estado del modelo [1]. Comúnmente, los modelos de simulación parten de la abstracción de un fenómeno contextualizado en un mundo particular dado. Luego, la definición de tales fenómenos se convierte en un problema de interés a nivel conceptual que contribuye a la posterior abstracción, formalización e implementación del modelo de simulación asociado [2].

En la literatura existen diversos modelos y ontologías que incorporan el concepto de "evento". Por ejemplo, en [3] se propone un modelo que conceptualiza eventos como entidades. En este trabajo, los autores argumentan que (al menos en los lenguajes orientados a objetos), el modelado de eventos como entidades proporciona beneficios sustanciales. Sin embargo, a nivel ontológico, esta decisión no parece tener fundamentos válidos (ya que los eventos no suelen corresponder a entidades u objetos). Desde una perspectiva diferente, los autores de [4] proponen un modelo donde los eventos se definen conceptualmente en una ontología fundacional denominada UFO-B. Una ontología fundacional es una ontología que es independiente de cualquier do-

minio. Luego, UFO-B les permite trabajar con eventos al mismo nivel que el resto de los elementos fundacionales definidos en UFO. Sin embargo, en un contexto fundacional diferente, los eventos pueden ser vistos como afirmaciones (assertions) que tienen lugar a partir de la existencia de determinadas cosas particulares o universales.

En este trabajo se presenta una ontología "core" denominada *ParticularEventCO* (*PEventCO*) que, sobre la base de la ontología fundacional *ThingFO* (*Thing Foundational Ontology*) [5], incorpora la noción de eventos como "assertions" que tienen lugar a causa del comportamiento de las entidades. Así, *PEventCO* se basa en los términos y relaciones existentes en *ThingFO* para la definición de nuevos elementos a nivel "Core" que facilitan la representación de comportamientos basados en eventos.

El resto del trabajo se encuentra estructurado de la siguiente manera. La Sección 2 detalla los fundamentos de la propuesta. La Sección 3 presenta la ontología en el contexto de la arquitectura FCD-OntoArch, detallando los términos, relaciones, propiedades y axiomas que la conforman. La Sección 4 describe una prueba de concepto. Finalmente, la Sección 5 se encuentra dedicada a las conclusiones y trabajos futuros.

2 Objetos & Eventos

En las ciencias aplicadas, los modelos se construyen sobre la base de la abstracción de un fenómeno. Es decir, un modelo puede ser concebido como una representación física, matemática y/o lógica de un sistema, entidad, o proceso sobre el cual se genera una abstracción la cual, subsecuentemente, es formalizada e implementada en un lenguaje particular para los fines propios de la ciencia. En este contexto, en el área de M&S, el sistema del mundo real a partir del cual se define un modelo es considerado el *fenómeno y/o sistema de interés* [6]. De acuerdo con los autores de [7], los modelos de simulación suelen asociarse de forma natural al mundo real (mucho más que los sistemas de software), ya que lidian con la representación de entidades de la parte del mundo real con la que tienen que tratar (y no con artefactos de implementación asociados). Luego, estas entidades existen en un mundo particular, donde el modelador entiende la realidad según sus propias construcciones mentales.

El M&S basado en eventos discretos se basa en la noción de evento, el cual se define como un cambio en el estado del modelo [1]. Un *evento* se produce en un instante determinado (llamado tiempo del evento) y hace que una parte componente del modelo se active para producir un cambio de estado (por ejemplo, que se genere un cambio en algún atributo de la entidad representada). En este contexto, el estado de una entidad es el conjunto de valores de todos sus atributos en un instante determinado. Los *sistemas de interés* al cual pertenecen este tipo de entidades (que pueden representarse con variables discretas sobre una base de tiempo continuo), se denominan Sistemas Dinámicos de Eventos Discretos (Discrete-Event Dynamic Systems, DEDS), y los diferentes métodos de M&S de DEDS se denominan métodos de M&S de eventos discretos. Aun existiendo métodos que dan soporte al M&S de este tipo de sistemas, de acuerdo con [1-2], es necesario definir nuevos métodos para modelar los fenómenos en los cuales las entidades cambian debido a la ocurrencia de eventos particulares y la evolución del *sistema de interés* depende de las interacciones de dichos eventos.

Una semántica claramente definida del modelo conceptual de un dominio conduce a una mayor calidad general del modelo construido sobre dicho modelo de dominio, dando lugar a una mejora en el nivel de comprensión, la capacidad de mantenimiento, la interoperabilidad y la capacidad de evolución [7]. El principal beneficio que se obtiene al establecer los fundamentos ontológicos de un dominio es un claro entendimiento de su semántica en el mundo real. Una ontología es una especificación explícita de una conceptualización, es decir, una visión abstracta y simplificada del mundo que incluye los objetos y sus relaciones en un dominio de interés [8]. En este contexto, las ontologías como mecanismo de soporte para la definición de los DEDS surgen naturalmente como una solución factible de ser utilizada.

El término Simulación de Eventos Discretos (Discrete Event Simulation, DES) se ha establecido como un término general que engloba distintos enfoques de simulación. Estos enfoques se basan en la idea general de representar un DEDS modelando su estado como una estructura compuesta de variables de estado y, luego, modelar su dinámica en base a los eventos responsables de los cambios de estado [9]. Pegden [10] caracteriza estos enfoques según "worldviews". El paradigma denominado "event worldview" ve al *sistema de interés* como una serie de eventos instantáneos que cambian su estado en el tiempo. Sin embargo, de acuerdo con [11], este paradigma es ontológicamente incompleto. Esto se debe a que el mundo real consiste esencialmente en objetos y eventos, lo que da lugar a un nuevo paradigma denominado "object-event worldview". No es posible concebir los eventos como una serie de ocurrencias aisladas que no tienen ninguna vinculación con los objetos, ya que son los objetos los actores primarios de los eventos. En este contexto, las propiedades de estado de un objeto por medio de sus capacidades son las que influyen en la evolución de su comportamiento y, viceversa, los comportamientos en reacción a acciones externas dan lugar a cambios en sus propiedades.

Con el objetivo de describir esta dinámica a nivel semántico, en este trabajo se presenta la ontología *PEventCO*. Esta ontología es incorporada como parte de la arquitectura FCD-OntoArch, donde la ontología fundacional es *ThingFO* [5,12]. La ontología propuesta es un primer paso hacia la definición de un modelo para el nivel de "phenomenon" propuesto en [2]. En dicho trabajo, los autores proponen una conceptualización multicapa del formalismo de M&S Discrete-Event System Specification (DEVS) [6], donde la capa superior (denominada *phenomenon*) busca representar los distintos aspectos de un sistema dinámico de interés de forma independiente a la estrategia de simulación subyacente.

3 PEventCO: Ontología Core en FCD-OntoArch

3.1 Niveles de la Arquitectura FCD-OntoArch

La arquitectura FCD-OntoArch ha sido propuesta en [5] como una estructura ontológica basada en cuatro niveles conceptuales que dan lugar a la definición de distintos tipos de ontologías según el nivel de abstracción al cual pertenecen. Se trata de una estructura basada en capas, en la cual cada capa se corresponde con un nivel de abs-

tracción específico, a saber: *Foundational*, *Core*, *Domain* e *Instance*. El nivel *Domain* a su vez se divide en dos subniveles específicos: *Top-domain* y *Low-domain*. Esto permite distinguir los modelos ontológicos de dominio general de los modelos ontológicos de ámbito específico.

Esta arquitectura multicapa promueve el uso de diferentes niveles de abstracción para la definición de ontologías. Por medio del uso de estos niveles ontológicos, se da lugar a la correcta ubicación de los modelos conceptuales dentro del esquema general. Por lo tanto, la arquitectura fomenta la modularidad, extensibilidad y reutilización de los elementos ontológicos en todos los niveles. A fin de brindar una identificación de los modelos, las ontologías que pertenecen a un nivel de abstracción específico llevan una denominación asociada a su ubicación dentro de FCD-OntoArch. Por ejemplo, las ontologías de nivel *Foundational* se denominan *Foundational Ontologies* (FO), mientras que los modelos semánticos de nivel *Core* se denominan *Core Ontologies* (CO).

Existen dos premisas que guían la incorporación de modelos a FCD-OntoArch:

- *Premisa #1*: Toda nueva ontología situada a nivel N debe garantizar una correspondencia de sus elementos con los elementos definidos en el nivel inmediato superior (es decir, el nivel N-1). Por ejemplo, para introducir una nueva ontología a nivel *Core* se debe garantizar que sus elementos tengan una correspondencia con los elementos definidos en el nivel *Foundational*. Esto permite que los términos y relaciones de las ontologías de los niveles inferiores se enriquezcan semánticamente con los términos y relaciones de las ontologías de los niveles superiores.
- *Premisa #2*: Las ontologías de un mismo nivel pueden relacionarse entre sí, pero debe garantizarse que en su definición conjunta (como un todo) no se violen los principios del nivel inmediato superior. Esto implica que, si una ontología de nivel Core utiliza elementos de otra ontología del mismo nivel, en conjunto ambos modelos deben garantizar una correcta definición respecto al nivel *Foundational*. Esto permite que los términos y relaciones de las ontologías de un mismo nivel puedan complementarse entre sí, manteniendo una correspondencia con las definiciones de las ontologías de los niveles superiores.

La Figura 1 presenta la arquitectura FCD-OntoArch. La estructura de paquetes situada en la parte izquierda de la imagen presenta los niveles de abstracción propuestos. La estructura de paquetes en gris (situada en la parte derecha de la figura) refiere al conjunto de ontologías ya incluidas en la arquitectura. A nivel *Foundational*, los autores proponen una única ontología: *ThingFO* [5]. En cuanto al resto de los niveles, las ontologías ya incluidas son: *i)* a nivel *Core*: *ProcessCO*, *GoalCO*, *SituationCO*, y *ProjectCO*; *ii)* a nivel *Top-domain*: *TestTDO*, *FRsTDO* (Functional Requirements), *NFRsTDO* (Non-Functional Requirements), y *MEvalTDO* (Measurement and Evaluation); y *iii)* a nivel *Low-domain*: *MetricsLDO* e *IndicatorsLDO*.

En este contexto, la ontología *PEventCO* (paquete en verde, situado en la parte derecha de la Figura 1) se encuentra situada a nivel *Core* y, en esta instancia de definición, no interactúa con otras ontologías del mismo nivel.

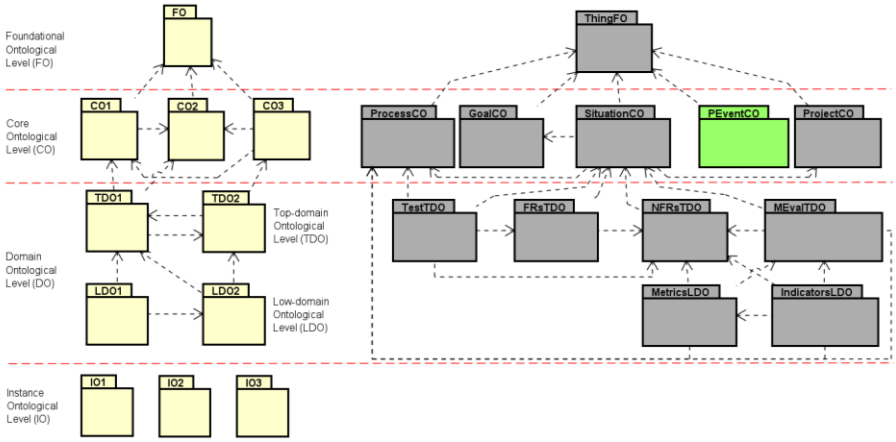

Fig. 1. Ubicación de la ontología PEventCO dentro de la arquitectura FCD-OntoArch (Foundational, Core, and Domain Ontological Architecture for Sciences) propuesta en [5].

ThingFO. Las ontologías fundacionales son representaciones sobre construcciones primitivas de nivel superior independientes del dominio. En este contexto, *ThingFO* ha sido propuesta como una ontología de nivel FO en FCD-OntoArch la cual se vale de un conjunto reducido de términos y relaciones que refieren tanto a elementos particulares como universales del mundo. Su objetivo es identificar el conjunto mínimo de términos, propiedades, relaciones y restricciones que representen el mundo, permitiendo que los mismos sean reutilizados y/o especializados en todos los dominios de las distintas ciencias. Una primera versión de *ThingFO* ha sido presentada en [5]. Esta versión ha sido actualizada en [12], habiendo sido validada por los presentes autores.

Los principales términos incluidos en esta ontología son: *Thing*, *Property* y *Power* para particulares, *Thing Category* para universales y *Assertions* como elemento clave para representar las afirmaciones que surgen cuando un agente humano representa y modela intencionadamente elementos del mundo que corresponden tanto a cosas particulares (*Assertion on Particulars*) como a universales (*Assertion on Universals*). En *PEventCO*, los principales términos y relaciones reutilizados del nivel FO son aquellos que corresponden a particulares y sus afirmaciones.

La base de la representación de particulares en *ThingFO* es que una *Thing* (objeto tangible o intangible de un mundo particular dado), representa una entidad a partir de la cual se pueden generar individuos (o instancias). Sin embargo, una *Thing* no existe sin sus *Properties* y *Powers*. Una *Property* refiere a la constitución intrínseca, la estructura o las partes de una *Thing* concreta, mientras un *Power* refiere a lo que una *Thing* concreta hace o puede hacer. Luego, los términos *Property* y *Power* son incluidos como componentes mandatorios de *Thing* (no pudiendo existir de forma aislada en el mundo particular modelado).

En un mundo a representar, las *Things*, *Thing Categories* y *Assertions* no se especifican sin conexión. Por este motivo, en *ThingFO* se incluye un conjunto de relaciones que garantizan el principio de no aislamiento de las *Things*, la posible pertenencia de

una *Thing* a una o más *Thing Categories*, la definición de *Assertions* por parte de las *Things*, la agregación de distintas *Assertions*, y jerarquías de *Things* y *Thing Categories*, entre otras.

Para el caso de *Assertions*, en *ThingFO* se definen 12 subtipos que representan distintas clases de afirmaciones. Por ejemplo, se define *Time-related Assertion* como una afirmación que implica la especificación de límites temporales, entre otros aspectos, como parte de diferentes situaciones y/o eventos. Otro ejemplo es *Action-related Assertion*, la cual es definida como una afirmación relacionada con la interacción de las *Things* (ya que los *Powers* de una *Thing* dan lugar a cualquier tipo de acontecimiento que pueda ocurrir en un mundo dado). Para mayores detalles de *ThingFO*, se sugiere consultar [12].

3.2 PEventCO: Términos, Propiedades y Relaciones

Haciendo uso de un diagrama de clases UML, la Figura 2 presenta el conjunto de términos, propiedades y relaciones que compone la ontología *PEventCO*. Es importante notar que, dada la arquitectura *FCD-OntoArch*, debe existir una correspondencia entre los elementos que conforman el modelo *PEventCO* y los elementos definidos en *ThingFO* (premisa #1). Esta premisa es válida tanto para los términos como así también para las relaciones y propiedades.

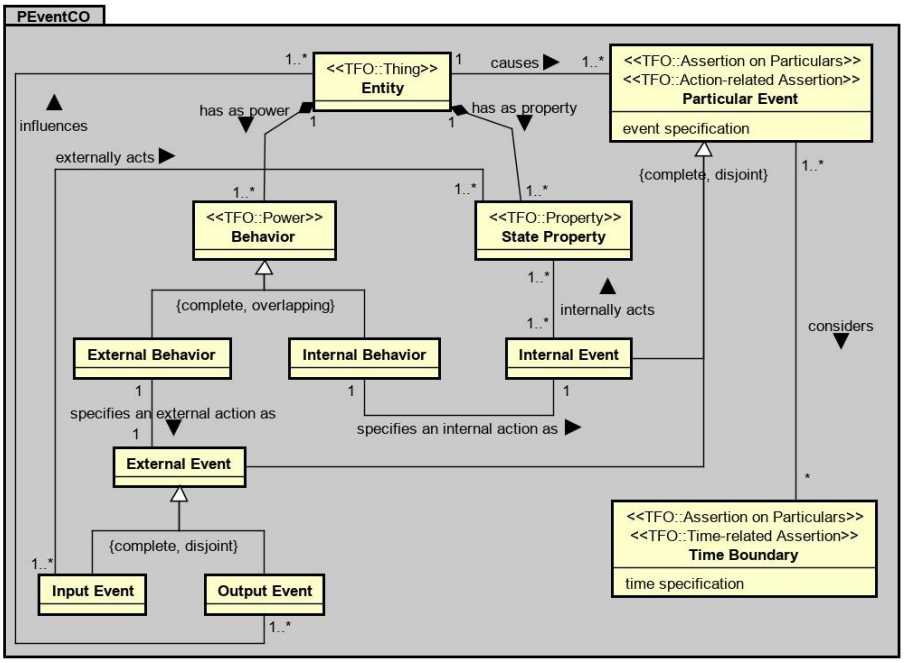

Fig. 2. Diagrama UML de los términos, relaciones y propiedades que componen la ontología PEventCO. Los estereotipos refieren a términos definidos en ThingFO (TFO).

De acuerdo con los lineamientos planteados en [13], cuando UML es utilizado como lenguaje para generar modelos ontológicos, se pueden emplear los estereotipos como mecanismo de enriquecimiento semántico para nuevos conceptos. El diagrama UML propuesto en la Figura 2 hace uso de este mecanismo a fin de establecer un vínculo entre términos definidos a nivel CO (es decir, en *PEventCO*) y términos situados a nivel FO (es decir, en *ThingFO*). Por otra parte, la única forma de establecer una correspondencia entre relaciones de nivel CO y FO, es por medio de la definición de axiomas. Tales axiomas son enunciados en la subsección 3.3.

En *PEventCO*, una *Entity* representa un objeto tangible o intangible para el cual se define un comportamiento dinámico. Luego, *Entity* tiene semántica de *Thing* (del nivel FO) ya que dicho comportamiento es descripto haciendo uso de sus *Powers* and *Properties*. Específicamente, las *Powers* que componen una *Entity* se definen como *Behaviors*. Esto permite considerar las capacidades de las entidades como comportamiento asociado a las mismas. A su vez, las *Properties* que componen una *Entity* se definen como *State Properties*. De esta manera, las entidades quedan definidas según las propiedades asociadas a su estado. En conjunto, ambas definiciones dan lugar a centrar la atención en la forma en la cual los *Behaviors* actúan en las *State Properties*. De acuerdo con Fleetwood [14], las capacidades son la forma de actuar de las propiedades de las cosas (es decir, las capacidades son las propiedades de las cosas en acción). Luego, este conjunto de cosas, propiedades y capacidades causa cualquier evento que pueda ocurrir. En *PEventCO*, los eventos son modelados a partir del concepto *Particular Event*. La tupla *Entity-Behavior-State Property-Particular Event* es la base para conceptualizar la existencia de eventos.

Todos los eventos se corresponden con un *Particular Event*. En este contexto, un evento particular tiene semántica de *Assertion on Particulars* y *Action-related Assertion* de *ThingFO*. De esta manera, un evento particular declara y especifica (como una aserción) la ocurrencia de una acción en una entidad. Se relaciona con la forma en la cual las entidades interactúan, ya que los comportamientos causan cualquier evento que tenga lugar. Es decir, un evento particular establece la ocurrencia de la acción de algún objeto concreto en el contexto de un lugar determinado durante un intervalo o instante de tiempo particular. En este sentido, los mecanismos de eventos pueden requerir considerar límites temporales. Luego, un *Time Boundary* tiene semántica de *Assertion on Particulars* y *Time-related Assertion* de *ThingFO* con el objetivo de especificar las restricciones temporales a partir de las cuales se dan eventos particulares.

En *PEventCO*, existen dos tipos de *Behaviors* que pueden formar parte de una *Entity*, a saber: *Internal Behavior* y *External Behavior*. Un *Internal Behavior* representa un comportamiento que refiere a lo que una entidad particular puede hacer para actuar sobre su estado. Los comportamientos internos actúan sobre las *State Properties* por medio de acciones internas que dan lugar a *Internal Events*. Es decir, un comportamiento puede modificar y/o consultar las propiedades de estado haciendo uso de eventos internos que se desprenden de su definición. Por su parte, un *External Behavior* representa un comportamiento que refiere a lo que una entidad particular puede hacer para influenciar otras entidades y/o en respuesta a la influencia de otras entidades. Los comportamientos externos dan lugar a acciones externas que se presentan como *External Events*. Existen dos tipos de *External Events*, a saber: *Input Event* y *Output Event*. Un *Input Event* es un evento externo asociado a un comportamiento

externo que refiere a la ocurrencia de una acción externa que actúa sobre las propiedades de estado de una entidad. Esto es, un evento de entrada se asocia a la forma en la cual la entidad reacciona ante estímulos externos por medio de un comportamiento propio definido. Este evento, actúa sobre las propiedades de estado ya sea modificando o consultando su estructura. En contraposición, un *Output Event* es un evento externo que declara y especifica explícitamente la ocurrencia de una acción externa en una entidad (a partir de un comportamiento externo) que busca generar algún impacto sobre las propiedades de otras entidades. En este punto, es importante comprender que una entidad no puede afectar a otra de forma directa. Son las capacidades (en este caso, los comportamientos) de una entidad los que actúan sobre sus propiedades de estado. Luego, cuando una entidad afecta a otra lo hace por medio de un evento de salida. Este evento es producido por el comportamiento externo de la entidad origen de la interacción. En respuesta, la entidad destino actúa en consecuencia según su propio comportamiento externo asociado al evento de entrada. En este sentido, un evento de entrada puede ser visto como el estímulo que necesita una entidad para actuar en consecuencia a la influencia de otra entidad externa.

Las Tablas 1 a 3 presentan, respectivamente, la definición de cada término, propiedad y relación que forma parte de *PEventCO*.

Tabla 1. Definición de los términos de PEventCO que han sido especificados en la Figura 2 haciendo uso de clases UML. Se detallan los sinónimos a fin de dar consistencia a la definición. La versión en inglés de estas definiciones se encuentra disponible aquí.

Término	Definición
Entity (sinónimo: **Dynamic Entity)**	Objeto particular o concreto, tangible o intangible, para el que se define de forma explícita un comportamiento dinámico con sus propiedades y capacidades.
Behavior	Refiere a la forma en la que se comporta una entidad en particular bajo condiciones establecidas.
State Property	Refiere a la estructura de estado intrínseca de una determinada entidad.
Particular Event	Aserción que declara y especifica explícitamente la ocurrencia de una acción en una entidad.
Time Boundary	Es una aserción relacionada con el tiempo que especifica los límites y restricciones temporales a partir de los cuales se dan los eventos particulares.
Internal Behavior (sinónimo: **Internal Power**)	Es un tipo de comportamiento que establece y especifica la ocurrencia de acciones internas en una entidad.
External Behavior (sinónimo: **External Power**)	Es un tipo de comportamiento que establece y especifica la ocurrencia de acciones externas sobre una entidad.
Internal Event (sinónimo: **Entity Internal Event**)	Evento particular que establece y especifica explícitamente la ocurrencia de una acción interna en una entidad que actúa sobre sus propiedades de estado.
External Event (sinónimo: **Entity External Event**)	Evento particular que declara y especifica explícitamente la ocurrencia de una acción externa en una entidad.
Input Event (sinónimo: **External Input Event**)	Es un evento externo que declara y especifica explícitamente la ocurrencia de una acción externa que actúa sobre las propiedades de estado de una entidad.
Output Event (sinónimo: **External Output Event**)	Es un evento externo que declara y especifica explícitamente la ocurrencia de una acción externa en una entidad que tiene algún impacto sobre otras entidades.

Tabla 2. Definición de las propiedades de los términos incluidos en PEventCO que han sido especificadas en la Figura 2 haciendo uso de atributos de clases UML. En este caso, estos atributos redefinen la propiedad "specification" de "Assertion" (término de ThingFO). La versión en inglés de estas definiciones se encuentra disponible aquí.

Término	Propiedad	Definición
Particular Event	**event specification**	Especifica el evento en un lenguaje dado.
Time Boundary	**time specification**	Especifica las relaciones y restricciones temporales para los eventos, ya que los eventos ocurren en el tiempo.

Tabla 3. Definición de las relaciones no taxonómicas de PEventCO que han sido especificadas en la Figura 2 haciendo uso de asociaciones UML. La correspondencia de estas relaciones con las de nivel FO se presenta en los axiomas de la subsección 3.3. La versión en inglés de estas definiciones se encuentra disponible aquí.

Relación	Definición
has as power	Una entidad tiene uno o más comportamientos que definen sus capacidades.
has as property	Una entidad tiene una o más propiedades de estado que definen sus propiedades estructurales.
causes	Una entidad dinámica causa uno o más eventos particulares.
specifies an internal action as	Un comportamiento interno especifica un evento interno.
specifies an external action as	Un comportamiento externo especifica un evento externo.
internally acts	Un evento interno actúa internamente sobre propiedades de estado.
externally acts	Un evento de entrada actúa externamente en propiedades de estado.
considers	Un evento particular puede requerir de límites temporales.
influences	Un evento de salida influencia a una o más entidades.

3.3 PEventCO: Axiomas

En *PEventCO* se definen 12 axiomas que se clasifican en dos grandes grupos. Por un lado, tal como se ha enunciado con anterioridad, se definen axiomas que garantizan la correspondencia entre relaciones de nivel CO y nivel FO. Por otro lado, se definen axiomas que, además de dar consistencia entre niveles, dan consistencia entre los elementos definidos a nivel CO. Al primer grupo, pertenecen los axiomas A1 a A8. Los axiomas restantes pertenecen al segundo grupo. La Tabla 4 resume todos los axiomas incluidos en el modelo.

El axioma A1 define que una *Entity* que causa un *Particular Event* es una *Thing* que define una *Assertion on Particulars*. Luego, la relación *causes* del nivel CO redefine la relación *defines* de nivel FO, donde *Entity* actúa como *Thing* y *Particular Event* como *Assertion on Particulars*. Lo mismo ocurre con *hasAsPower* y *hasAsProperty* respecto a las composiciones *Thing-Power* y *Thing-Property* (axiomas A2 y A3). El axioma A4, redefine la relación *relatesWith* de nivel FO entre dos *Assertions* como *considers* para el caso de las aserciones *ParticularEvent* y *TimeBoundary*.

En este mismo contexto, los axiomas A5 a A7 definen la forma en la cual las relaciones *actsUpon* e *interactsWithOther* del nivel FO pueden ser inferidas a partir de lo modelado a nivel CO. De acuerdo con el nivel FO, la relación *actsUpon* entre *Power* y *Property* es definida como la vinculación que se establece cuando "un *Power* actúa

sobre una o más *Properties*, pudiendo consultarlas o actualizar su estado". Luego, esta relación ayuda a representar los eventos que son producidos por un comportamiento cuyo efecto actuará sobre las propiedades de estado. En este sentido, hay dos tipos de eventos que actúan sobre las propiedades de estado: eventos internos y eventos de entrada. El axioma A5 define la forma en la cual las relaciones *specifiesAnInternalActionAs* e *internallyActs* entre *InternalBehavior*, *InternalEvent* y *StateProperty*, componen la relación *actsUpon* de nivel superior. Lo mismo ocurre en el axioma A6 con las relaciones *specifiesAnExternalActionAs* y *externallyActs* entre *ExternalBehavior*, *InputEvent* y *StateProperty*. Por su parte, en el nivel FO, la relación *interactsWithOther* se define como el vínculo que se establece cuando, debido al *Power* de una *Thing*, las cosas interactúan entre sí. Esta interacción ayuda a representar eventos de salida que son producidos por el comportamiento de una entidad pero que, a su vez, buscan afectar a otras entidades. El axioma A7 define la forma en la cual las relaciones *specifiesAnExternalActionAs* e *influences* entre *ExternalBehavior*, *OutputEvent* y *Entity* dan lugar a *interactsWithOther*.

El axioma A8 busca redefinir la relación *dealsWithParticulars* para el caso de *influences* entre *OutputEvent* y *Entity*.

Los axiomas A9 a A11 se encuentran vinculados a la forma en la cual los eventos son causados por una entidad. Específicamente, el axioma A9 indica que todo *InternalEvent* producto del *InternalBehavior* de una *Entity*, es causado por dicha *Entity* y, además, a nivel FO, define la relación *dealsWithParticulars*. De forma similar, el axioma A10 efectiviza la misma definición, pero para un *InputEvent* especificado a partir de un *ExternalBehavior*. Sin embargo, para el caso de *OutputEvent* aun cuando el evento es causado por la *Entity*, no se lo vincula a nivel FO por medio de la relación *dealsWithParticulars* con dicha *Entity* (axioma A11).

Tabla 4. Definición de los axiomas que complementan los términos y relaciones de la ontología PEventCO haciendo uso de lógica de primer orden. La definición completa de estos axiomas en inglés se encuentra disponible aquí.

Id.	Definición
A1	*∀e,∀ev:[Entity(e) ∧ ParticularEvent(ev) ∧ causes(e,ev) → defines(e,ev)]*
A2	*∀e,∀b: [Entity(e) ∧ Behavior(b) ∧ hasAsPower(e,b) → partOf(b,e)]*
A3	*∀e,∀s: [Entity(e) ∧ StateProperty(s) ∧ hasAsProperty(e,s) → partOf(s,e)]*
A4	*∀e,∀ib: [ParticularEvent(e) ∧ TimeBoundary(tb) ∧ considers(e,tb) → relatesWith(e,tb)]*
A5	*∀ib,∀ie,∀sp: [InternalBehavior(ib) ∧ specifiesAnInternalActionAs(ib,ie) ∧ InternalEvent(ie) ∧ internallyActs(ie,sp) ∧ StateProperty(sp) → actsUpon(ib,sp)]*
A6	*∀eb,∀ie,∀sp: [ExternalBehavior(eb) ∧ specifiesAnExternalActionAs(eb,ie) ∧ InputEvent(ie) ∧ externallyActs(ie,sp) ∧ StateProperty(sp) → actsUpon(eb,sp)]*
A7	*∀eb,∀oe,∀e: [ExternalBehavior(eb) ∧ specifiesAnExternalActionAs(eb,oe) ∧ OutputEvent(oe) ∧ influences(oe,e) ∧ Entity(e) → interactsWithOther(eb,e)]*
A8	*∀oe,∀e: [OutputEvent(oe) ∧ influences(oe,e) ∧ Entity(e) → dealsWithParticulars(oe,e)]*
A9	*∀e,∀ib,∀ie:[Entity(e) ∧ partOf(ib,e)∧ InternalBehavior(ib) ∧ specifiesAnInternalActionAs(ib,ie)∧ InternalEvent(ie) → causes(e,ie) ∧ dealsWithParticulars(ie,e)]*
A10	*∀e,∀eb,∀ie: [Entity(e) ∧ partOf(eb,e) ∧ ExternalBehavior(eb) ∧ InputEvent(ie) ∧ specifiesAnExternalActionAs(eb,ie) → causes(e,ie) ∧ dealsWithParticulars(ie,e)]*
A11	*∀e,∀eb,∀oe: [Entity(e) ∧ partOf(eb,e) ∧ ExternalBehavior(eb) ∧ OutputEvent(oe) ∧ specifiesAnExternalActionAs(eb,oe) → causes(e,oe) ∧ ¬dealsWithParticulars(oe,e)]*
A12	*∀e,∀tb,∀en: [ParticularEvent(e) ∧ TimeBoundary(tb) ∧ considers(e,tb) ∧ Entity(en) ∧ dealsWithParticulars(e,en) → dealsWithParticulars(tb,en)]*

Finalmente, el axioma A12 tiene como objetivo dar coherencia a la relación *dealsWithParticulars* de nivel FO una vez establecido un vínculo a nivel CO para *ParticularEvent*. Este axioma detalla que el *TimeBoundary* de un *ParticularEvent* que se encuentra vinculado a nivel FO por *dealsWithParticulars* a una *Entity*, también debe estar vinculado por *dealsWithParticulars* a dicha *Entity*.

4 Instanciando PEventCO: The Tape System

Con el objetivo de generar instancias del modelo semántico propuesto como *PEventCO* y, teniendo en cuenta que su definición depende de *ThingFO*, ambas ontologías fueron implementadas adoptando el lenguaje Telos. Telos es un lenguaje de modelado conceptual para representar el conocimiento sobre sistemas de información basado en los conceptos centrales de tecnología orientada a objetos, restricciones de integridad y reglas deductivas. Por lo tanto, se empleó el lenguaje Telos para especificar formalmente los conceptos y relaciones introducidos en la subsección 3.2 y los axiomas del modelo propuesto en la subsección 3.3. Dicha implementación fue realizada utilizando la herramienta ConceptBase [15], la cual es un administrador de base de datos deductivo orientado a objetos que, en nuestro caso, facilita la verificación y validación del modelo propuesto.

En la Figura 3 se ilustra la especificación en Telos del concepto *Particular Event* (Tabla1). En este lenguaje, cada uno de los axiomas modelados para deducir nuevos hechos en la ontología *PEventCO*, han sido definidos como reglas (*rules*). Por ejemplo, la expresión del axioma A4 (en Tabla 4) ha sido definida como la regla "rule A4_r" (Figura 3).

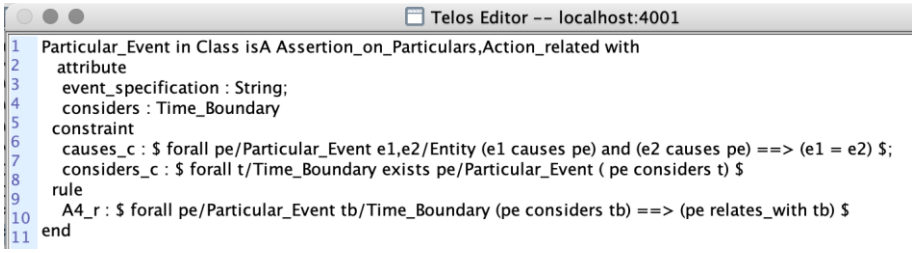

Fig. 3. Definición en Telos del término "Particular Event" junto con los axiomas asociados. Estos axiomas han sido definidos como "reglas".

Para comprobar que el modelo semántico propuesto es satisfacible (junto con *ThingFO*), se ha instanciado un caso clásico empleado en el área de M&S [6]: una máquina de Turing construida a partir de dos entidades denominadas TM Control y Tape System.

La Figura 4 se centra en la representación del elemento "tape" que refiere a la entidad Tape System. Dado el tamaño de las instancias definidas para este caso, la entidad TM Control sólo se visualiza como "tm_control". En la parte inferior de la figura

se ilustran las instancias. Los arcos sólidos entre instancias representan relaciones explícitamente enunciadas en la base de datos. Por su parte, los arcos en líneas punteadas refieren a relaciones inferidas a partir de axiomas (reglas).

El "tape" está compuesto por tres propiedades: cinta del tape (elemento "tape_i_SP"), posición de la cabeza lectora (elemento "pos_SP"), y próximo movimiento que debe realizar la cabeza sobre la cinta (elemento "mov_SP"). El comportamiento del "tape" se define mediante el comportamiento aceptación de un símbolo (definido como "accept_symbol_B") y el comportamiento asociado al movimiento de la cabeza (definido como "move_head_B"). Ambos comportamientos se especifican a partir de los eventos identificados en el fenómeno modelado.

En particular, se tiene que "accept_symbol_B" es especificado por el evento de entrada "inputSymbol_InpE", el cual permite actuar sobre las propiedades del "tape": escribe el símbolo en la posición "pos_SP" actual de la cinta "tape_SP" y, además, fija el siguiente movimiento de la cabeza "mov_SP". Este evento de entrada desencadena los eventos que especifican a "move_head_B". Estos son: *i)* el evento interno "mov_IntE" el cual, a partir del movimiento configurado "mov_SP", actualiza la posición de la cabeza "pos_SP"; *ii)* el evento de salida "outputSymbol_OE", el cual influencia la "tm_control".

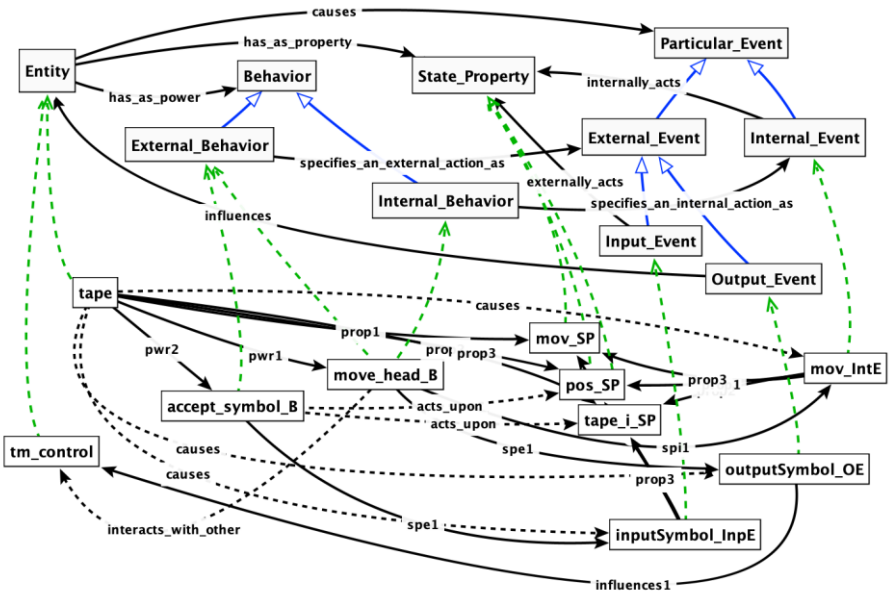

Fig. 4. Vista en ConceptBase de PEventCO en donde se instancia el "Tape System" propuesto en [6].

Luego, la ontología *PEventCO* ha dado soporte a la dinámica de comportamiento basada en eventos discretos requerida para este ejemplo. La propuesta ha permitido representar los eventos que suceden en el fenómeno bajo estudio y los elementos estructurales que son afectados, instanciándose en este caso los eventos vinculados a la

evolución del componente Tape de la máquina de Turing. Los axiomas propuestos establecen cierto nivel de integridad en *PEventCO*, así como entre el nivel CO y el nivel FO. Asimismo, esta prueba de concepto ha ilustrado que es posible la satisfacibilidad del modelo.

5 Conclusiones y Trabajos Futuros

En este trabajo se ha presentado una ontología denominada *PEventCO* que facilita el modelado de entidades cuyos comportamientos dinámicos causan distintos tipos de eventos. Esta ontología de nivel "core" se basa en la ontología fundacional *ThingFO*, por lo que sus términos, relaciones y propiedades reutilizan los elementos del nivel superior a fin de generar una compatibilidad dentro de la arquitectura FCD-OntoArch. Además, al situarse en dicho nivel, facilita la redefinición de sus conceptos en niveles de dominio como parte del trabajo futuro.

La principal diferencia que tiene esta propuesta con la ontología UFO-B [4], es que los eventos son conceptualizados como *Assertions*. En este caso, la semántica de dicho término definido a nivel fundacional enriquece la definición de los eventos permitiendo modelar mundos en los cuales la existencia de eventos se refleje a causa de la existencia de cosas particulares (en nuestro caso, entidades).

El principal beneficio de enmarcar la ontología de eventos dentro de la arquitectura FCD-OntoArch es que el contexto que rodea a las entidades que causan eventos y/o reaccionan a los mismos, puede ser definido en otros modelos ontológicos. Tales modelos pueden estar ubicados al mismo nivel que *PEventCO* o, en su defecto, a nivel inferior. Al mismo nivel, a priori se considera que la ontología *SituationCO* (ontología de nivel CO para situaciones particulares y genéricas [16]) jugará un papel importante en relación con *PEventCO*. Cuando las entidades se contextualizan en una situación, es posible modelar una situación particular de un mundo particular. Sin embargo, una forma de enriquecer la descripción de dicha situación es por medio de la incorporación de los comportamientos asociados a las entidades involucradas. Luego, es factible pensar que sería posible definir situaciones donde las entidades participantes exhiban (o no) este tipo de comportamiento según el interés del modelador. Esto ayudará a enriquecer la semántica del mundo real para el caso de situaciones que son percibidas en base a comportamientos discretos.

Referencias

1. Wainer, G. A.: Discrete-Event Modeling and Simulation: A Practitioner's Approach. CRC press (2017).
2. Blas, M., Gonnet, S, Zeigler, B. P.: Towards a Universal Representation of DEVS: A Metamodel-based Definition of DEVS Formal Specification. Accepted in the Theory and Foundations for Modeling and Simulation, 2021 Annual Modeling and. Simulation Conference (2021).

3. Olivé, A., Raventós, R.: Modeling Events as Entities in Object-Oriented Conceptual Modeling Languages. Data & Knowledge Engineering, 58(3), 243-262 (2006). https://doi.org/10.1016/j.datak.2005.07.002
4. Guizzardi, G., Wagner, G., De Almeida Falbo, R., Guizzardi, R. S., Almeida, J. P. A.: Towards Ontological Foundations for the Conceptual Modeling of Events. Proceedings of the 2013 International Conference on Conceptual Modeling, 327-341 (2013). https://doi.org/10.1007/978-3-642-41924-9_27
5. Olsina L.: Analyzing the Usefulness of ThingFO as a Foundational Ontology for Sciences. In Proceedings of the 2020 Argentine Symposium on Software Engineering, ASSE'20, 49 JAIIO, 172-191 (2020).
6. Zeigler, B. P., Muzy, A., Kofman, E.: Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations. Academic press (2018).
7. Guizzardi, G., Wagner, G.: Towards an Ontological Foundation of Discrete Event Simulation. In Proceedings of the 2010 Winter Simulation Conference, 652-664 (2010). https://doi.org/10.1109/WSC.2010.5679121
8. Gruber, T. A.: A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2), 199–220 (1993). https://doi.org/10.1006/knac.1993.1008
9. Wagner, G.: An Abstract State Machine Semantics for Discrete Event Simulation. In Proceedings of the 2017 Winter Simulation Conference, 762-773 (2017). https://doi.org/10.1109/WSC.2017.8247830
10. Pegden, C.D.: Advanced Tutorial: Overview of Simulation World Views. In Proceedings of the 2010 Winter Simulation Conference, 643−651 (2010). http://doi.org/10.1109/WSC.2010.5679161
11. Casati, R., Varzi, A.: The Stanford Encyclopedia of Philosophy. Available at http://plato.stanford.edu/archives/win2015/entries/events/ (2015).
12. Olsina L.: Applicability of a Foundational Ontology to Semantically Enrich the Core and Domain Ontologies. In proceedings of IC3K, KEOD'21, 13th International Conference on Knowledge Engineering and Ontology Development, held virtually in Oct. 2021, Portugal, pp. 1-9 (2021).
13. Becker P., Papa F., Olsina L.: Process Ontology Specification for Enhancing the Process Compliance of a Measurement and Evaluation Strategy, CLEI eJnal., 18:(1), 1–26 (2015).
14. Fleetwood S.: The Ontology of Things, Properties and Powers. Journal of Critical Realism, 8:(3), 343-366 (2009).
15. Koubarakis, M., Borgida, A., Constantopoulos, P. A retrospective on Telos as a meta-modeling language for requirements engineering. Requirements Eng 26, 1–23 (2021). https://doi.org/10.1007/s00766-020-00329-x
16. Olsina L., Tebes G., Becker P.: SituationCO v1.2's Terms, Properties and Relationships - A Core Ontology for Particular and Generic Situations. Preprint in Research Gate, Available at https://doi.org/10.13140/RG.2.2.23646.56644 (2021).