Principios de Simulación para Sistemas Ciber-Físicos y Gemelos Digitales

Instituto de Computación
Facultad de Ingeniería, UdelaR

November 8, 2022

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

El curso

Temario:		
Lunes 7	**Introducción a los Sistemas Ciber-Físicos**	
	Complejidad de los Sistemas Ciber-Físicos	1:15hs
	Ingeniería basada en modelos	
	Integración de modelos	
	Internet de las Cosas como ejemplo de Sistema Ciber-Físico	
	Coffee break.	20min
	Lab: presentar caso de uso del lab. introducción IoT, Lora, WiFi, Arquitectura	1:15hs
Martes 8	**Principios de Simulación y Co-Simulación de Sistemas Ciber-Físicos**	
	Simulación y co-simulación de tiempo continuo	1:15hs
	Simulación y co-simulación de eventos discretos	
	Co-simulación híbrida	
	coffee break	20min
	Lab: intro a NS3, Cocosim, instalar VM, probar herramientas.	1:15hs
Miercoles 9	Laboratorio Gemelos Digitales	3:00hs
Jueves 10	**Gemelos Digitales**	
	Concepto y arquitectura	
	Casos de estudio y modelos de ejemplo en diversas industrias	
Viernes 11	Laboratorio Gemelos Digitales	3:00hs

Los Docentes

- Renzo E. Navas, IMT Atlantique, Rennes, France
- Javier Roman, IIE, FING, UdelaR, Uruguay
- Matías Richart, INCO, FING, UdelaR, Uruguay
- Javier Baliosian, INCO, FING, UdelaR, Uruguay

El Grupo de Investigación MINA

In the beginning was... el marcapasos

Subject: INVITACIÓN: Software embebido. Algunas experiencias de desarrollos locales
Date: Mon, 24 Nov 2003 16:44:15 -0300
From: Eduardo Grampin <grampin@fing.edu.uy>
To: Gonzalo Tejera - INCO <gtejera@fing.edu.uy> et al

Estimados, este martes 25/11/03 a las 20 hs, el Ing. XXXX XXXX de la empresa YYY dictará la charla:

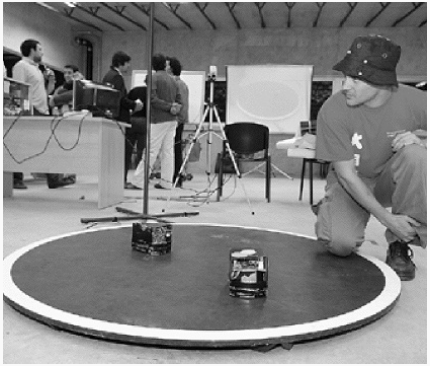
"Software embebido. Algunas experiencias de desarrollos locales"

YYY es una empresa nacional que produce marcapasos basados en un microcontrolador RISC. Esta charla describe el trabajo en programación de bajo nivel para estos sistemas embebidos. A partir de esta primera actividad conjunta que hacemos, estamos pensando proponer un taller de programación en firmware para el año que viene como electiva del área de arquitectura y afines.

La charla es en el Salón 101, y forma parte del curso de Arquitectura II. Están todos invitados.

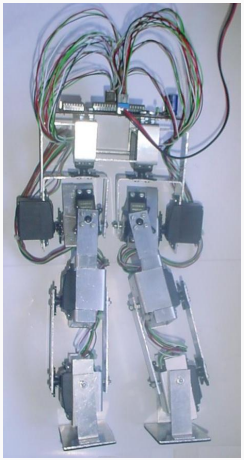
Saludos,
Eduardo

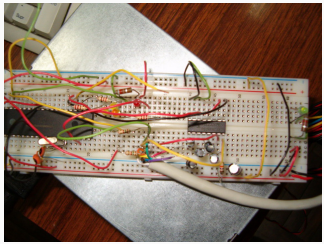
… EL *RIKISHI* DE SUMO



```
>
> -------- Original Message --------
> Newsgroups:     fing.general
> Subject:     Primer Campeonato Uruguayo de Sumo de Robots, sumo.uy 2004
> Date:     Tue, 18 May 2004 14:17:17 -0300
> Organization:     Facultad de Ingenieria.
>
> *"PRIMER CAMPEONATO URUGUAYO DE SUMO DE ROBOTS, sumo.uy 2004"*
>
>
> En el mes de agosto del presente año, organizado por el Departamento de
> Arquitectura del Instituto de Computación, se desarrollará un evento
> consistente de un campeonato abierto de sumo robótico en el que
> participarán equipos de nuestra universidad, de otras universidades,
> liceos y público en general, junto con un workshop donde los
> participantes presentarán sus equipos, e investigadores presentarán
> artículos relacionados.
>
> Por más información, en breve estará disponible el sitio web
> http://www.fing.edu.uy/inco/eventos/sumo.uy
>
>
```

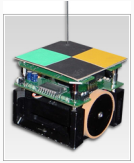
... Y EL PRIMER BÍPEDO

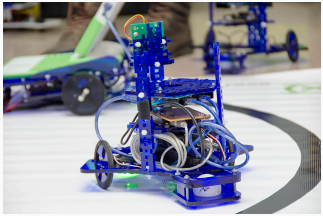




https://www.fing.edu.uy/inco/grupos/mina/pGrado/pgrobip/index.html

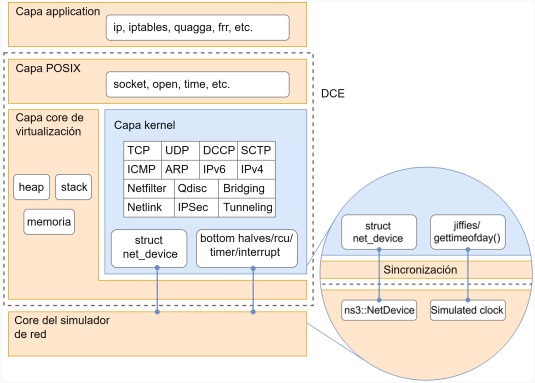
Luego llegó el fútbol de robots (con visión) y butiá





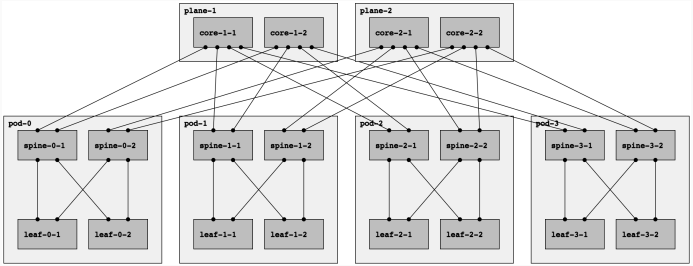
...Y LAS REDES WIFI COMUNITARIAS

Simular es una necesidad

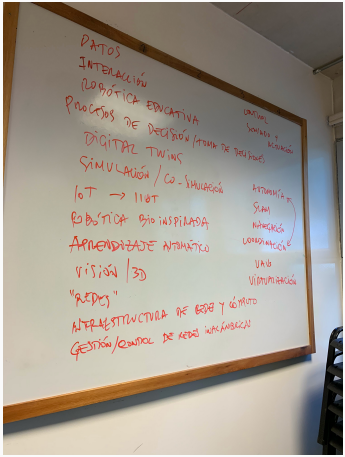


Y se necesitan recursos de cómputo y redes

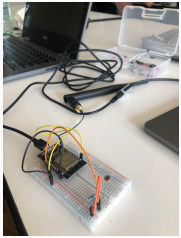
- Gestión y Control de redes ópticas (flexigird) e inalámbricas (WiFi, LoRa, 5/6G...)
- La internet, cloud, datacenters, CDNs....
- Elasticidad, escalabilidad
- Soluciones distribuidas/ centralizadas, optimización, ML...

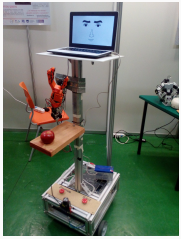


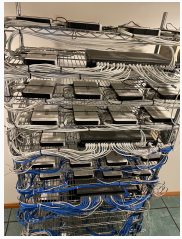
Recapitulando: Mapa Conceptual



Otras aplicaciones







Introducción a los Sistemas Ciber-Físicos

Los Sistemas Cíber Físicos (SCF) o, en inglés, *Cyber-Physical Systems*, son sistemas que integran computación, redes y procesos físicos, con bucles de realimentación donde procesos físicos impactan sobre procesos de cómputo y viceversa.

El desafío es combinar abstracciones que han evolucionado durante siglos para modelar los sistemas físicos (p.e., ecuaciones diferenciales y procesos estocásticos), con abstracciones de las ciencias de la computación (algoritmos y programas).

Estos mismos desafíos se dan a hora de simular estos sistemas, por ejemplo, para implementar la idea de gemelos digitales.

Internet de las Cosas como ejemplo de Sistema Ciber-Físico

¿Cuál es la relacion entre IoT y SCF?

Los conceptos de SCF e IoT surgieron de comunidades diferentes; los SCF surgen principalmente de una perspectiva de ingeniería de sistemas y control.

SCF

Es un sistema de elementos computacionales que colaboran para controlar entidades físicas. Típicamente, sistemas mecánicos y eléctricos ... se conectan en red utilizando componentes de software. Utilizan el conocimiento compartido y la información de los procesos para controlar de forma independiente sistemas de logística y producción[1].

[1] Minerva R, Biru A, Rotondi D (2015) Towards a definition of the Internet of Things (IoT). IEEE. https://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Issue1_14MAY15.pdf

¿Cuál es la relación entre IoT y SCF?

En cambio, el concepto de IoT surgió desde una perspectiva de redes y tecnologías de la información, que preveía la integración del ámbito digital en el mundo físico.

IoT

El término se utiliza como palabra clave para abarcar diversos aspectos relacionados con la extensión de Internet y la Web al ámbito físico, mediante el despliegue generalizado de dispositivos distribuidos espacialmente con capacidades integradas de identificación, detección y/o actuación[2].

[2] Miorandi D, Sicari S, Pellegrini F, Chlamta I (2012) Internet of Things: Vision, Applications and Research Challenges. Ad Hoc Networks 10 (7): 1497-1516. https://doi.org/10.1016/j.adhoc.2012.02.016

¿En qué se diferencian?

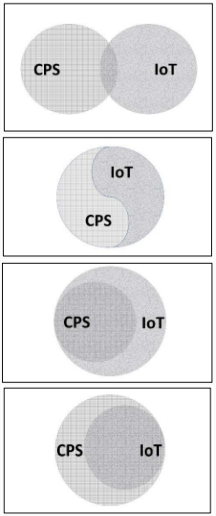
Aunque tanto IoT como CPS pretenden aumentar la conexión entre el ciberespacio y el mundo físico mediante el uso de tecnología de detección de información e interactiva, tienen diferencias evidentes[3]:

- IoT hace hincapié en la creación de redes y pretende interconectar todas las cosas del mundo físico. Por lo tanto, se trata de una plataforma e infraestructura de red abierta;
- SCF hace hincapié en el intercambio de información y la retroalimentación, donde el sistema debe retroalimentar y controlar el mundo físico además de detectar el mundo físico, formando un sistema de bucle cerrado.

[3] Ma H (2011) Internet of Things: Objectives and Scientific Challenges. Journal of Computer Science and Technology 26 (6): 919–924. https://doi.org/10.1007/s11390-011-1189-5

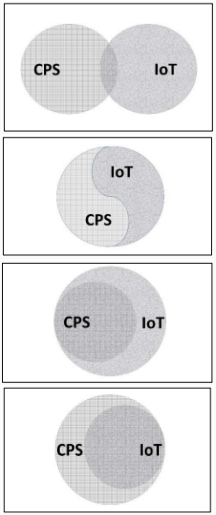
¿Cuál es la relación entre IoT y SCF?

- Las definiciones de CPS e IoT están convergiendo con un énfasis común en los sistemas híbridos de componentes digitales, analógicos, físicos y humanos que interactúan en sistemas diseñados para funcionar a través de la física y la lógica integradas.
- Reconocer esta convergencia puede unir campos y sectores actualmente aislados para avanzar en torno a objetivos y oportunidades de investigación, aplicación e innovación compartidos.
- El diseño y la construcción de los sistemas SCF/IoT requieren la consideración del contexto funcional del sistema, incluyendo cómo se utiliza el sistema y con qué propósito o resultado.



¿Cuál es la relación entre IoT y SCF?

- Una perspectiva unificada de los sistemas CPS/IoT facilita el trabajo en la composibilidad abierta y la composicionalidad fiable para la innovación en la creación de nuevos sistemas y sistemas de sistemas.
- Esto permite priorizar los objetivos de investigación, desarrollo y despliegue, incluyendo la habilitación de estrechos vínculos de estado físico y lógico y el desarrollo de métodos híbridos discretos y continuos para la conceptualización, realización y garantía de los sistemas CPS/IoT.
- La naturaleza híbrida de los sistemas CPS/IoT tiene importantes implicaciones para la ingeniería, como el verificabilidad del diseño, la seguridad ciberfísica, la gestión del ciclo de vida, la temporización y la sincronización, etc.



Complejidad de los Sistemas Ciber-Físicos

- Un sistema es un grupo de entidades organizadas que interactúan para formar un todo unificado con el propósito de lograr un objetivo general.
- Un sistema puede ser **físico**, **conceptual** o ambos
- la materia y la energía son las partes principales de un sistema físico
- la información y el conocimiento se codifican utilizando las relaciones materia-energía que muestran el comportamiento general observable.
- Un sistema conceptual abstrae las partes físicas y las relaciones de un sistema físico para exhibir su significado.
- Define las relaciones abstractas entre sus elementos pero no las interacciones reales.

- La combinación de ambos sistemas debe ser estudiada organizarse y gestionarse con un método bien definido.

"An engineered system is a system designed or adapted to interact with an anticipated operational environment to achieve one or more intended purposes while complying with applicable constraints"[4]

[4] International Council on Systems Engineering, ed. INCOSE Systems Engineering Handbook. Vol. 2.0. 2000.

- Consideramos que un sistema es complejo cuando está compuesto por componentes heterogéneos e interconectados que interactúan de forma intrincada.
- Un sistema **complicado** no es necesariamente un sistema **complejo**.
- Un sistema complicado puede descomponerse en sus partes y cada una de ellas puede estudiarse de forma aislada.
- El comportamiento y las propiedades resultantes del sistema son la composición de todos los comportamientos y propiedades de cada modelo.
- Sus interacciones no cambian su comportamiento original.

complejo

En un sistema complejo, las interacciones entre los componentes conducen a cambios en los comportamientos de cada modelo de forma de forma no ambigua o comportamientos inesperados.

- Un sistema complejo requiere ser ejecutado como un todo para observar su comportamiento global.
- En un sistema complejo, la composición de las propiedades de cada componente no se se refleja en el todo, y no pueden estudiarse como un subconjunto disjunto de las propiedades del sistema complejo.

Optimización de sistemas complejos

- Para un sistema de este tipo, la optimización del rendimiento global en su conjunto no puede lograrse optimizando el rendimiento de cada subsistema y componente.
- Por el contrario, debe lograrse teniendo en cuenta las interacciones entre los componentes, el entorno, los sistemas de control, el software y el comportamiento humano.
- Sólo si tenemos en cuenta el sistema en su conjunto, podremos optimizarlo en relación con las de optimización deseada, por ejemplo, la eficiencia, la seguridad, el consumo o el coste.

El la ingeniería basada en modelos propone abordar la complejidad del desarrollo apoyándose en abstracciones de los diferentes componentes.

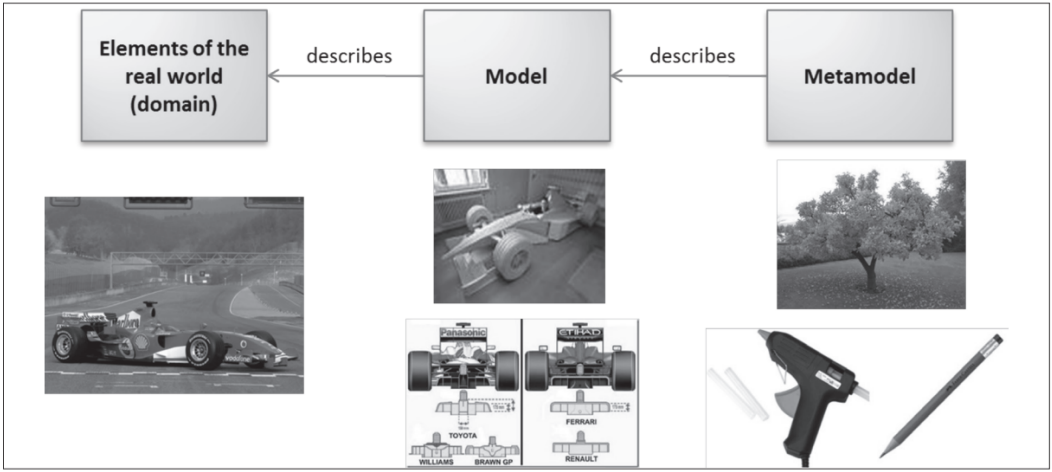
Ingeniería basada en modelos

MDE: La respuesta a la Complejidad I

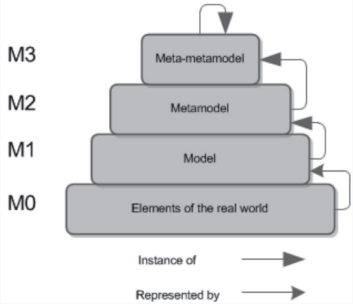
- Un enfoque para el desarrollo de software llamado Ingeniería Dirigida por Modelos (MDE),
- Eleva el nivel de abstracción de los lenguajes tradicionales mediante el uso de modelos, permitiendo el uso de conceptos más cercanos al dominio de los problemas.
- No nació como una forma de estudiar sistemas sino como una forma de desarrollarlos.
- Sin embargo, la abstracción resultante puede utilizarse como sustituto de la realidad, ya que proporciona resultados equivalentes.
- Un concepto clave para trabajar con modelos es el **metamodelo**.

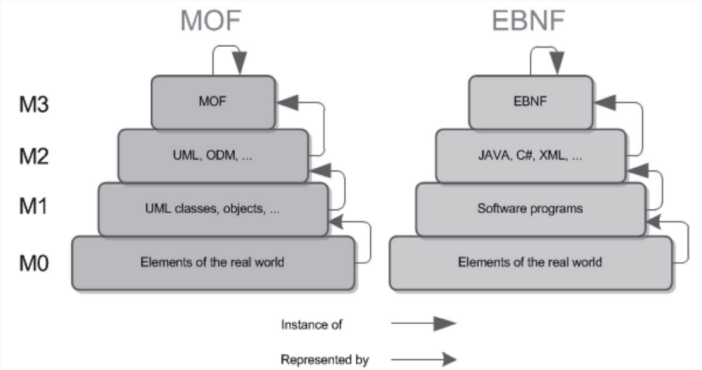
Un modelo es una abstracción de un componente, dispositivo o sistema original que refleja sólo el subconjunto de propiedades relevantes para un fin determinado y que puede utilizarse en lugar del original, evitando manejar la complejidad completa de la realidad.

Elementos de MDE



Modelos, Metamodelos, y Meta-metamodelos





¿Cómo deben ser?

- Un modelo debe ser independiente, reutilizable e interoperable.
- La independencia y la reutilización permiten a un experto utilizar los modelos existentes como ladrillos de construcción para desarrollar nuevos sistemas, junto con el grado de abstracción requerido.
- La interoperabilidad permite utilizar el modelo en diferentes sistemas sin un esfuerzo significativo por parte del usuario final.

- los modelos se utilizan como representación de sistemas, entidades o procesos que muestran su comportamiento en el tiempo.
- dos tipos de tiempo diferentes: el tiempo del reloj en la pared y el tiempo simulado.
- Podemos utilizar los modelos para probar el sistema sin necesidad de probarlo en la realidad.

- La simulación nos permite conocer el impacto de un cambio en el modelo sin necesidad de realizar una costosa prueba sobre el terreno.
- La simulación se convierte en una tecnología clave para desarrollar sistemas complejos que tienen un impacto directo en el ser humano y su vida.

Integración de modelos

Cómo integrar modelos I

- El diseño basado en modelos de un sistema ciberfísico (CPS) requiere manejar tanto la parte cibernética como la física del sistema.
- El enfoque basado en componentes se utiliza como mecanismo de base para permitir la integración y simulación de modelos.
- La co-simulación es un enfoque de integración de modelos en el que éstos se integran utilizando sus entradas y salidas expuestas como interfases.
- La co-simulación permite combinar componentes heterogéneos (es decir, modelos).
- Consiste en técnicas que permiten la simulación en conjunto de un sistema componiendo diferentes modelos y simuladores heterogéneos.

Cómo integrar modelos II

- Una co-simulación puede integrar la dinámica continua en simuladores con hardware-in-the-loop en tiempo real, bancos de pruebas físicos o software y hardware.
- Tradicionalmente, la simulación de sistemas complejos da lugar a una simulación monolítica, personalizada para un fin específico.
- La co-simulación permite interconectar modelos y subsistemas independientes (no integrados) y simular el sistema creado como un todo.
- Es necesario orquestar la ejecución de los distintos simuladores para obtener una simulación coherente y correcta.
- El algoritmo que controla la ejecución de cada simulador independiente se llama Algoritmo Maestro (MA).

Cómo integrar modelos III

- El MA controla cómo avanza el tiempo simulado en cada simulador e intercambia datos entre ellos en función de su topología.

Functional Mock-up Interface (FMI) I

- FMI es un estándar abierto para el intercambio de modelos.
- FMI define una interfaz en C que se implementa mediante un ejecutable llamado *Functional Mock-up Unit* (FMU).
- Un entorno de simulación puede utilizar FMI para crear una instancia de la FMU y simularla junto con otras FMU o modelos nativos del entorno de simulación.

La *Functional Mock-up Unit*

Una FMU es el ejecutable que implementa la interfaz. Durante la exportación de una FMU se genera un archivo FMU a partir de un modelo del sistema, mientras que durante la importación de una FMU se genera un modelo del sistema a partir de un archivo FMU.

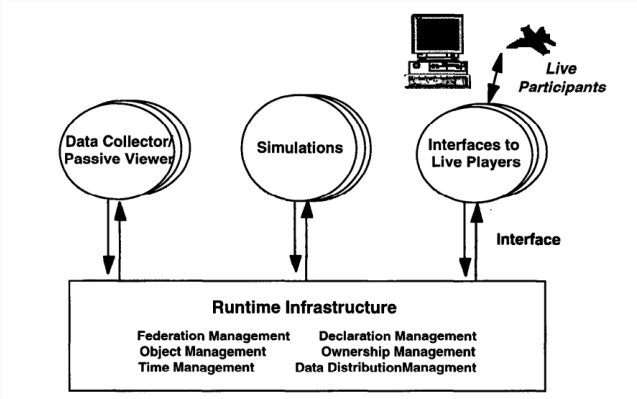
Una FMU contiene:

- **Un archivo XML que describe el modelo:** contiene información sobre el modelo, por ejemplo, definiciones de variables, e información del modelo, como el nombre, la herramienta de generación y la versión de FMI.
- **Ecuaciones del modelo:** Un modelo puede describirse mediante ecuaciones diferenciales ordinarias, relaciones algebraicas y ecuaciones discretas. Estas ecuaciones pueden representarse a su vez mediante funciones C.
- **Archivos de recursos opcionales:** archivos opcionales que pueden incluirse en la FMU, documentación (HTML), icono del modelo (archivo de mapa de bits), mapas y tablas y otras bibliotecas o DLL que se utilizen en el modelo.

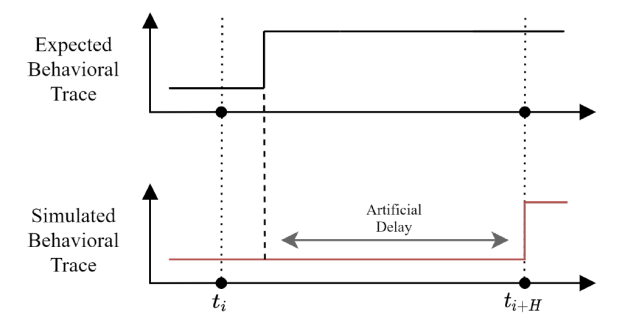
Intercambio de modelos y co-simulación:

Las normas FMI especifican actualmente dos tipos de protocolos: FMI para el intercambio de modelos (importación y exportación), y FMI para la co-simulación (master/slave). La principal diferencia entre estos dos protocolos es que en el intercambio de modelos la FMU se simula utilizando el solver de la herramienta de importación, mientras que en la co-simulación la UMF se envía con su propio solver.

Originalmente desarrollada por la *Defense Modeling & Simulation Office* (DMSO) en los años 90.



- **Avances de tiempo independientes**. Los avances de tiempo de cada simulador ocurren en sincronía con la hora del reloj de pared (o la hora del reloj de pared escalada, derivada como una función lineal de la hora del reloj de pared). El simulador avanza autónomamente su propio tiempo sin coordinar dichos avances con el RTI.
- **Avances de tiempo coordinados y pautados**. Cada simulador ocurre en sincronía con el tiempo del reloj de pared (escalado), pero los avances de tiempo están coordinados con el RTI para asegurar que las relaciones *antes y después* en el sistema físico se reproduzcan correctamente.
- **Avances de tiempo coordinados y sin ritmo definido.** Los avances de tiempo de cada simulador no están sincronizados con el tiempo del reloj de pared, y se coordinan los avances de tiempo para garantizar que las relaciones antes y después se modelen correctamente.



- El uso de la comunicación de activación temporal pura en una co-simulación de un sistema híbrido tiene efectos secundarios.

Sincronización II

- Hay que tener en cuenta dos aspectos principales: el retraso artificial introducido por el muestreo y el retraso en la propagación de datos entre simuladores.
- Una mayor precisión reduce el rendimiento, aumentando los puntos de comunicación necesarios.
- Una simulación rápida implica reducir el número de puntos de comunicación, lo que puede producir resultados de simulación no correctos.
- Una solución ideal requiere coordinar un componente de manera que produzca y consuma datos sólo cuando su semántica interna lo requiera.

Sincronización III

- En una co-simulación en red, cada punto de comunicación reduce el rendimiento global: los datos deben trasladarse de una máquina a otra a través de una red.
- Algunos enfoques proponen minimizar las transferencias de datos que requieren una red agrupando en la misma máquina las SU que tienen una fuerte conexión (es decir, fuertemente acopladas).

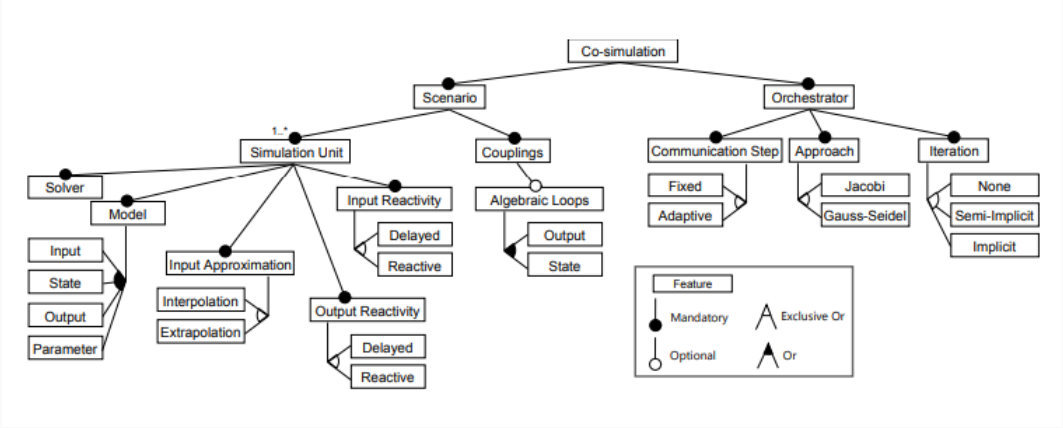
Principios de Simulación y Co-Simulación de Sistemas Ciber-Físicos

- La ingeniería de sistemas complejos requiere la integración de varios formalismos, herramientas y normas diferentes.
- La complejidad intrínseca del mundo real lleva a utilizar abstracciones y lenguajes dedicados a diferentes dominios que muestran explícitamente comportamientos interesantes.
- En la comunidad de la simulación, la simulación colaborativa se centra en la orquestación entre diferentes unidades de simulación que representan diferentes partes del mismo sistema,
- con el fin de comprender mejor el comportamiento emergente del sistema.

- la unidad de simulación es una entidad ejecutable, normalmente una caja negra, que puede, por ejemplo, encapsular un modelo y su *solver*, un proceso binario ejecutable o un proxy de un dispositivo de hardware.
- La orquestación es necesaria para definir los momentos en los que una unidad de simulación intercambia datos con otras unidades de simulación.
- Existen varios modelos de coordinación que siguen diferentes semánticas.
- Por ejemplo, los modelos de coordinación más populares se basan en la semántica de temporización por paso regular y por evento.
- Un modelo de coordinación puede estar distribuido en diferentes núcleos, CPUs, dispositivos o infraestructura de red. Todas estas propiedades y la heterogeneidad de la semántica repercuten en la precisión y el rendimiento general de la co-simulación.

Simulación y co-simulación de tiempo continuo

Conceptos de co-simulación I



El **escenario de co-simulación** señala una o más unidades de simulación, describe cómo las entradas y salidas de sus modelos, e incluye la configuración de parámetros relevantes.

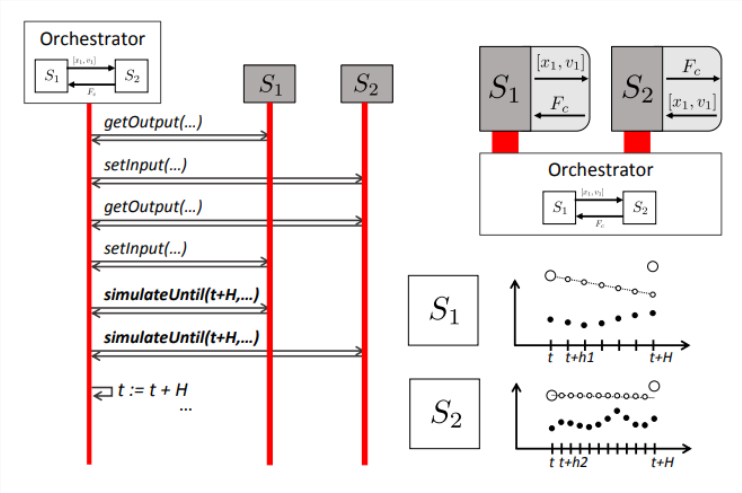
Cada **unidad de simulación** representa una caja negra capaz de producir un comportamiento. Para producir comportamiento, la unidad de simulación necesita tener una noción de:

- un modelo, creado por el modelador a partir de su conocimiento del sistema en estudio;
- un *solver*, que forma parte de la herramienta de modelado utilizada por el modelador, que aproxima el comportamiento del modelo; y

- una aproximación de la entrada, que aproxima las entradas del modelo a lo largo del tiempo, que será utilizada por el *solver*;
- reactividad de entrada y reactividad de salida, que determinan qué entradas recibe la unidad de simulación desde el orquestador.

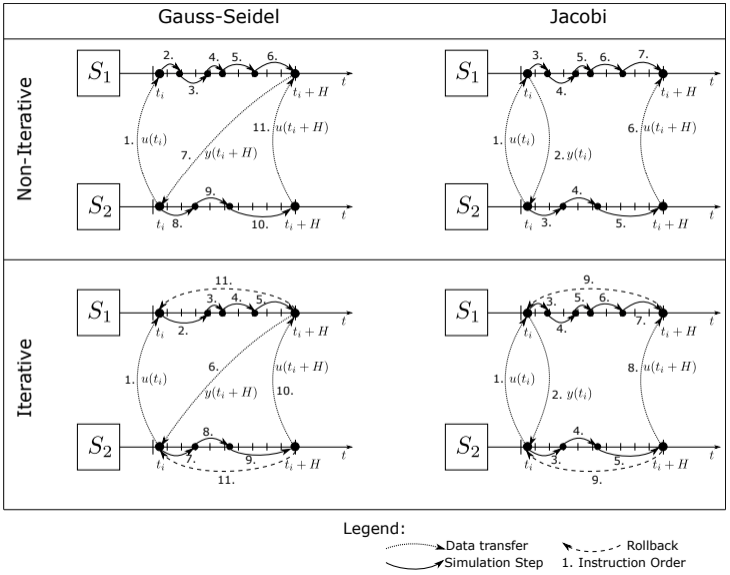
El **orquestador** es el responsable de ejecutar la co-simulación. Inicializa todas las unidades de simulación con los valores adecuados, establece/obtiene sus entradas/salidas y coordina su progresión a lo largo del tiempo simulado.

Conceptos de co-simulación IV



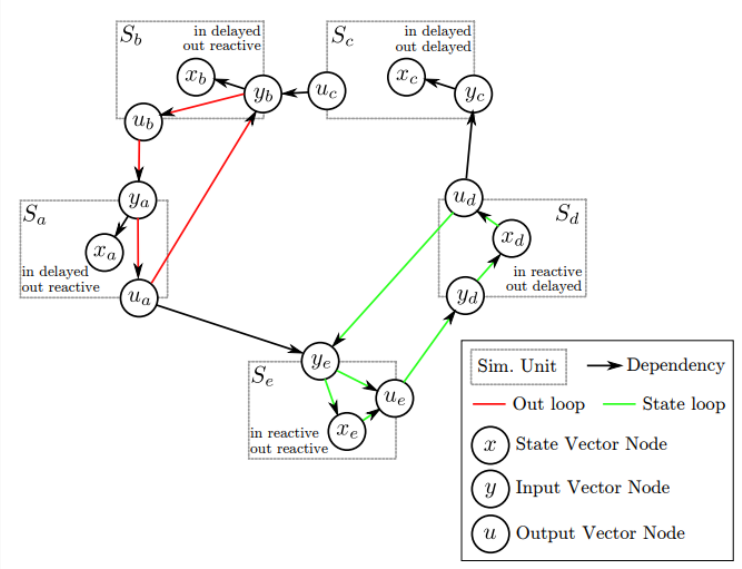
Ejemplo de coordinación de co-simulación (izquierda), escenario de co-simulación (arriba a la derecha) y comportamiento interno de las unidades de simulación (abajo a la derecha).

Conceptos de co-simulación V



Algoritmos de orquestación.

Conceptos de co-simulación VI



Ejemplo abstracto de escenario de co-simulación con el gráfico de dependencia. Las variables de entrada, estado y salida son vectores. Hay múltiples bucles algebraicos.

- En los sistemas ciberfísicos, los componentes físicos suelen diseñarse y modelarse utilizando ecuaciones diferenciales o un lenguaje que las utiliza.
- Estas ecuaciones pueden definirse utilizando el lenguaje *Modelica*.
- En estos componentes, las señales de entrada y salida son en su mayoría señales de tiempo continuo.

```
model FirstOrder
  parameter Real c=1 "Time constant";
  Real x (start=10) "An unknown";
equation
  der(x) = -c*x "A first order differential equation";
end FirstOrder;
```

Figure 1: El código en modélica para el sistema de primer orden ($\dot{x} = -c \cdot x, x(0) = 10$)

Simulación y co-simulación de eventos discretos

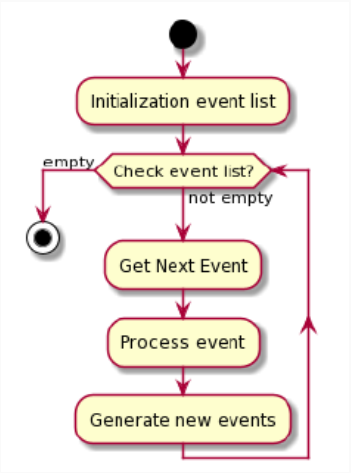
- Los sistemas como las redes, el software, las CPUs y el diseño de hardware presentan algunos fenómenos que no pueden expresarse mediante un sistema basado en TC.
- Problemas como sincronización, exclusión mutua, paralelismo, no pueden definirse ni estudiarse mediante ecuaciones diferenciales.
- El tiempo del modelo puede avanzar más rápido o más lento que el tiempo del reloj de pared.
- En una simulación de eventos discretos, el tiempo avanza no por la emulación del reloj de pared (es decir, representado como un continuo) sino por los cambios de los estados del sistema en instantes discretos en el tiempo.

DES

Una simulación de eventos discretos (DES) describe el comportamiento de un sistema definiendo cómo evoluciona el estado interno según una secuencia discreta de eventos.

- Un evento es un acontecimiento instantáneo que cambia el estado del sistema en un punto específico del tiempo, llamado marca de tiempo.
- La marca de tiempo tiene un alcance limitado a la unidad de simulación. Un evento puede causar otros eventos con la misma marca de tiempo o con una marca de tiempo mayor.
- A diferencia de una simulación en tiempo continuo, una unidad de simulación de eventos discretos **no puede cambiar su estado entre dos eventos sucesivos**.

- La simulación se lleva a cabo aumentando el tiempo y actualizando el estado del modelo en consecuencia (o cuando sea necesario).
- Cuando la lista de eventos está vacía o se alcanza el final del tiempo de simulación, la simulación se termina.
- Si un evento con una marca de tiempo mayor se ejecuta antes que otro con una menor se produciría un error de causalidad.
- En un sistema donde se ejecutan múltiples simulaciones ED paralelas, un reloj local no garantiza una marca de tiempo consistente para todos los eventos.



- La principal preocupación es la sincronización temporal entre los componentes del sistema: cuando varias partes de un sistema de ED se ejecutan en paralelo, pueden desencadenar eventos de forma esporádica o a un ritmo diferente.
- En un DES paralelo, una posible solución es utilizar un reloj de simulación global compartido entre los ordenadores.
- Sin embargo, la ejecución global puede sufrir un bajo rendimiento porque dos eventos raramente tienen la misma marca de tiempo exacta.
- Una posible solución para acelerar la simulación es permitir la ejecución concurrente de eventos en un tiempo simulado diferente, pero deben sincronizarse.

Co-simulación basada en eventos discretos II

- La sincronización se refiere a la correcta ejecución del componente de ED en un sistema distribuido, asegurando que los experimentos repetidos de la ejecución con el mismo conjunto de entradas producen los mismos resultados.
- Los algoritmos de gestión del tiempo pueden dividirse en dos categorías principales: sincronización conservadora y optimista.
- El enfoque conservador previene estrictamente la posibilidad de cualquier error de causalidad al determinar si es seguro procesar un evento.
- El enfoque optimista utiliza una estrategia de detección y recuperación: los errores de causalidad se permite que se produzcan errores de causalidad pero, cuando se detectan, debe ejecutarse un mecanismo de recuperación (es decir es decir, el rollback) debe ser ejecutado.

Co-simulación basada en eventos discretos III

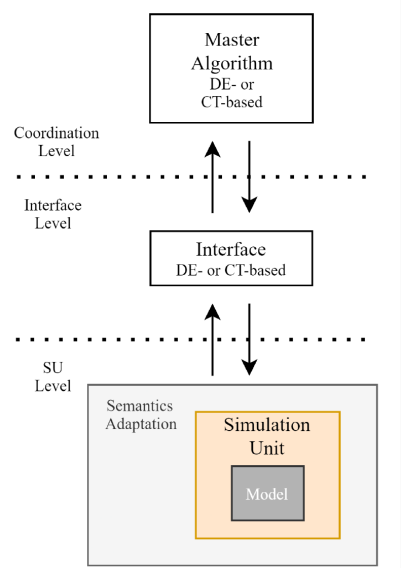
- En el contexto de la co-simulación CPS, las unidades de simulación no siempre pueden proporcionar un mecanismo de reversión y esto afecta a la usabilidad de este enfoque en los escenarios en los que hay esas unidades de simulación.
- Por el contrario, el enfoque conservador no requiere un mecanismo de reversión debido al mecanismo lookahead que impide que una SU se adelante en el tiempo si depende de otras SUs.
- Debido al mecanismo de rollback en caso de violación de la restricción de causalidad, puede dar lugar a una cascada de rollbacks que afecta al rendimiento global.
- El principal problema de este enfoque es que las operaciones de E/S no siempre pueden deshacerse con una reversión.

Co-simulación híbrida

Co-simulación Híbrida I

- Los sistemas que combinan sistemas de eventos discretos y de tiempo continuo se denominan sistemas híbridos.
- Esta noción se utiliza tradicionalmente en la comunidad de control para indicar dinámicas discretas y continuas. En concreto, un sistema híbrido tiene una evolución de tiempo continuo y cambios drásticos. Los cambios corresponden al cambio de estado de un autómata en respuesta a eventos externos.
- Los sistemas híbridos suelen describirse mediante formalismos y lenguajes específicos que mezclan comportamientos de tiempo discreto y continuo, describiendo los sistemas híbridos mediante un único lenguaje,
- para expresar los modelos físicos como ecuaciones diferenciales ordinarias (EDO) y los modelos cibernéticos como formalismo de flujo de datos.

Co-simulación Híbrida II



Principales conceptos de co-simulación: el algoritmo maestro, la interfaz y la unidad de simulación.

Adaptación semántica y composición de modelos. Cada simulador involucrado en una co-simulación define sus propios modelos de datos, interface y semánticas. A los efectos de hacerlos interoperar, si bien se pueden utilizar envolturas (wrapers) genéricas que los provean de interfaces y modelos comunes, en los hechos, la abstracción necesaria para lograr esto depende del escenario concreto. Sencillamente, no existe una sola opción de adaptación que sea mejor en todos los casos. Incluso a nivel técnico, la manera en que los eventos o señales se envían a (u obtienen de) cada simulador puede necesitar ser adaptada.

Tamaño predictivo de los pasos y localización de eventos. En los enfoques el avance del tiempo tiene que ser definido explícitamente. Sin embargo, un tamaño de paso temporal inadecuado puede causar pérdidas de eventos de uno u otro simulador. Como una restricción adicional a las habituales en la co-simulación híbrida, se agrega el requerimiento de elegir pasos de un tamaño que permitan realizar simulaciones a una velocidad compatible con el concepto de *digital twin*.

Identificación y manejo de discontinuidades. En una co-simulación, un simulador de tiempo continuo puede generar, a través de su envoltorio (*wraper*) una secuencia de eventos de salida correspondiente al muestreo de una determinada señal. Sin embargo, desde el punto de vista de otro de los simuladores involucrados, no es posible discernir si se trata de un muestreo de una señal continua, o de discontinuidades propias de una secuencia de eventos discretos. Este tipo de ambigüedades debe ser resuelto agregando funcionalidades y extendiendo los modelos de información de cada simulador. Una vez identificadas, estas discontinuidades deben ser manejadas correctamente. Los simuladores de tiempo continuo son, en realidad, un *mock-up* de un sistema continuo y las discontinuidades en las entradas deben ser manejadas de forma que la simulación resultante tenga sentido en el mundo

físico que simula, por ejemplo, la energía total debe conservarse, y un robot no debe cruzar una superficie sólida.

Estabilidad. En el caso de una co-simulación híbrida, es posible que una secuencia en particular de eventos cause que el sistema se vuelva inestable, incluso si todos los elementos continuos de la simulación son estables. Se hace necesario identificar, en el caso de los robots cooperativos, si los simuladores de tiempo continuo involucrados pueden inducir trayectorias inestables como resultado de entradas con ruido, el formato, los retardos o frecuencia de la mensajería.

Gracias.