WHITE PAPER

Diseño basado en modelos para sistemas de control espaciales

MathWorks®

Imagine que su equipo de trabajo está desarrollando el sistema de energía de un satélite. El sistema incorpora una combinación de elementos físicos (como batería y paneles solares), lógica de control y condiciones externas (como temperatura y radiación). Antes de comenzar con el diseño, es importante considerar algunos aspectos fundamentales; por ejemplo:

- Cómo dimensionar las baterías
- Qué ocurre si los requisitos cambian
- Cómo optimizar el diseño para garantizar el rendimiento deseado
- Cómo probar el diseño a fondo con el menor riesgo posible

Ya sea que esté desarrollando sistemas de control para un sistema de vuelo, un robot industrial, una turbina eólica, una máquina de producción, un vehículo autónomo, una excavadora o un servoaccionamiento eléctrico, si su equipo de trabajo desarrolla código manualmente y utiliza captura de requisitos basada en documentos, la única manera de abordar estos aspectos será mediante prueba y error, o realizando pruebas con un prototipo físico. Y si un solo requisito cambia, deberá volver a codificar y desarrollar todo el sistema, lo que supone días o incluso semanas de retraso en el proyecto.

Utilizando el diseño basado en modelos con MATLAB® y Simulink®, en lugar de código y documentos redactados manualmente, se crea un modelo del sistema, que incorpora el modelo físico, los algoritmos de control y el entorno. Puede simular el modelo en cualquier etapa para obtener una vista instantánea del comportamiento del sistema, probar varios escenarios hipotéticos y realizar análisis de tradeoffs sin riesgos ni retrasos, y sin necesidad de invertir en hardware.

Este white paper presenta una introducción al diseño basado en modelos, y ofrece sugerencias y prácticas recomendadas. Con ejemplos del mundo real, muestra cómo los equipos de trabajo de varios sectores industriales han adoptado el diseño basado en modelos para reducir el tiempo de desarrollo, minimizar los problemas de integración de componentes
y ofrecer productos de mayor calidad.

¿Qué es el diseño basado en modelos?

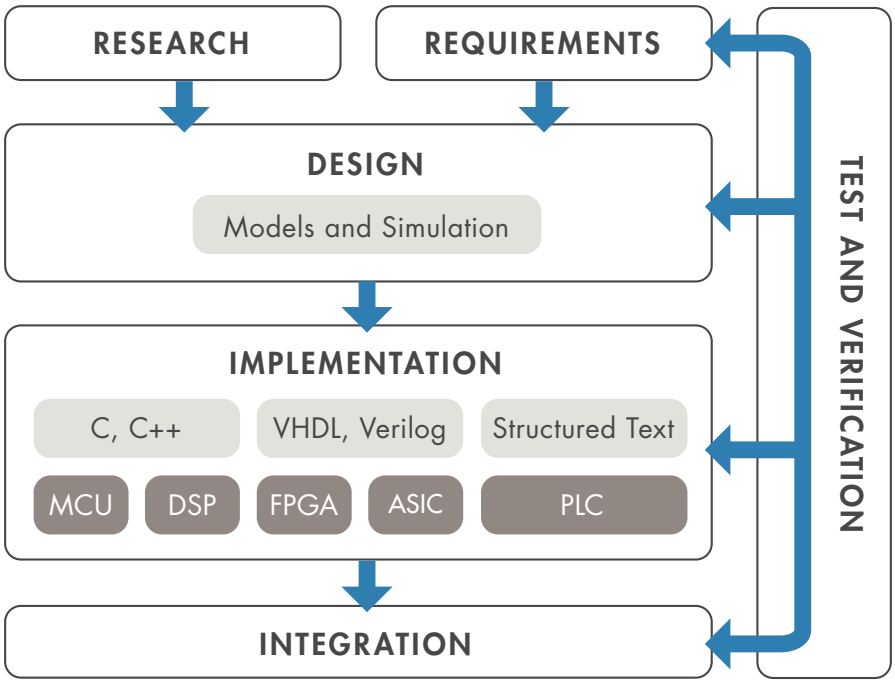
La mejor manera de comprender el diseño basado en modelos es a través de un ejemplo concreto:

> Un equipo de ingeniería aeroespacial se propone diseñar el sistema de orientación, navegación y control (GNC) de un satélite. Dado que utilizan el diseño basado en modelos, comienzan creando un modelo de la arquitectura a partir de los requisitos del sistema; en este caso, el modelo del satélite. A partir de ahí, se deriva un modelo de simulación/diseño. Este modelo de alto nivel y baja fidelidad incluye partes del software del sistema de control que se ejecutará en el satélite, además de la planta y el entorno operativo.
>
> El equipo de ingeniería realiza pruebas iniciales del sistema e integración simulando este modelo de alto nivel en varios escenarios, para verificar que el sistema se ha representado correctamente y que responde adecuadamente a las señales de entrada.
>
> Luego, incorporan detalles al modelo, probando y verificando continuamente el comportamiento en el nivel de sistema respecto de las especificaciones. Si el sistema es grande y complejo, cada componente se puede desarrollar y probar individualmente, y probarlos con frecuencia en una simulación completa del sistema.

Básicamente, se crea un modelo detallado del sistema y el entorno en el que opera. Este modelo refleja el conocimiento acumulado sobre el sistema (la propiedad intelectual). El grupo de ingeniería genera código automáticamente a partir del modelo de los algoritmos de control para realizar pruebas y verificación del software. Después de realizar pruebas de hardware-in-the-loop, descargan el código generado en hardware de producción para probarlo en el sistema final real.

Este ejemplo demuestra que el diseño basado en modelos utiliza los mismos elementos que los flujos de trabajo de desarrollo tradicionales, con dos diferencias fundamentales:

- Muchos de los pasos del flujo de trabajo que toman mucho tiempo o son proclives a errores están automatizados, tales como la generación de código.
- El desarrollo gira en torno a un modelo del sistema, desde la captura de requisitos hasta el diseño, la implementación y las pruebas.



Flujo de trabajo del diseño basado en modelos.

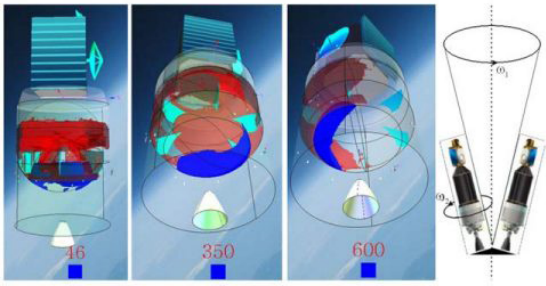
Ingeniería de sistemas, captura y gestión de requisitos

En un flujo de trabajo tradicional, donde los requisitos se capturan en documentos, la transferencia de requisitos puede provocar errores y retrasos. A menudo, el equipo de ingeniería que crea los documentos o requisitos de diseño no es el mismo que diseña el sistema, los requisitos se pasan de un grupo a otro y no suele haber comunicación clara o consecuente entre ambos.

En el diseño basado en modelos, los requisitos se crean, analizan y gestionan en un modelo de Simulink. Asimismo, se pueden crear requisitos de texto enriquecido con atributos personalizados y enlazarlos con diseños, código y pruebas. Los requisitos también se pueden importar y sincronizar desde fuentes externas, como herramientas de gestión de requisitos. Cuando un requisito vinculado al diseño cambia, se genera una notificación automática. De ese modo, se puede identificar qué parte del diseño o la prueba está afectada directamente por el cambio, y tomar las medidas adecuadas para abordarlo. El grupo de trabajo puede definir, analizar y especificar arquitecturas y composiciones para componentes de software y sistemas.

Y el equipo de ingeniería de sistemas puede utilizar MATLAB y Simulink para realizar análisis dinámicos. Con modelos ejecutables de sistemas terrestres y de naves espaciales multidominio, validan y verifican los requisitos, lo que proporciona información útil sobre el comportamiento y el rendimiento en el nivel de sistema que no se puede obtener únicamente mediante análisis estáticos. Este enfoque también permite realizar un seguimiento de los requisitos a partir de especificaciones de alto nivel, supervisar la implementación detallada de los requisitos en el diseño, y realizar el seguimiento de requisitos en el código fuente generado automáticamente. También se pueden vincular requisitos a casos de prueba y medir automáticamente la cobertura de los requisitos a medida que se ejecutan los casos de prueba.

Caso práctico: *ESA y Airbus Defence and Space*



Movimiento del propulsor en etapas superiores girando a 46, 350 y 600 segundos. La distribución después de 350 segundos se torna irregular.

"El diseño basado en modelos nos permitió crear un marco para diseñar controladores de vuelo con algoritmos de diseño de control robustos de vanguardia, crear modelos físicos multidominio, ajustar el diseño a través de optimización, y generar código para pruebas de PIL en el hardware de destino, todo en el mismo entorno".

— Hans Strauch, Airbus D&S

Cuando un lanzador de la Agencia Espacial Europea (ESA), como Ariane 5 o Vega, pone su carga útil satelital en órbita, el sistema de control de actitud (ACS) toma el control, orienta la carga útil y ordena la separación de la etapa superior del lanzador. Además de orientar el satélite, el sistema ACS debe identificar y gestionar problemas relacionados con el proceso de separación, el sistema de amortiguamiento del propulsor y una amplia variedad de posibles fallos de hardware.

ESA y Airbus necesitaban simular fallos de separación con un modelo físico para probar la capacidad del controlador de detectar fallos y tomar medidas correctivas. También necesitaban simular el sistema de amortiguamiento del propulsor, fugas en tuberías, válvulas atascadas y otro tipo de fallos. Además, deseaban ejecutar optimizaciones para identificar el rendimiento del sistema en el peor de los casos si se produjeran fallos.

Asimismo, buscaban probar los algoritmos de control en hardware del computador de vuelo en las primeras etapas de desarrollo. A medida que aumenta su complejidad, los algoritmos de control superan el límite de rendimiento del procesador y otros recursos con carga computacional. Por lo tanto, necesitaban verificar el rendimiento de los algoritmos y la utilización de recursos en un computador de vuelo representativo durante el diseño del controlador, cuando resulta más fácil corregir problemas.

Para lograrlo, ESA y Airbus utilizaron el diseño basado en modelos con MATLAB, Simulink y una combinación de herramientas de generación de código, modelado físico, diseño de lógica de control, verificación y validación para crear el Marco de diseño y control de actitud de la etapa superior (USACDF). El marco, que permite la simulación de lazo cerrado y la verificación de algoritmos de control con modelos físicos, se utiliza para desarrollar demostradores de conceptos complejos para operaciones de misiones de mantenimiento orbitales.

Diseño

En un enfoque tradicional, cada idea de diseño se debe codificar y probar en un prototipo físico. Como resultado, solo se puede explorar un número limitado de ideas y escenarios de diseño, ya que cada iteración de prueba aumenta el tiempo y el coste de desarrollo del proyecto. En el ámbito espacial, las decisiones de diseño son sumamente fundamentales a la hora de garantizar que el sistema físico utilizado en la misión sea el correcto.

Con el diseño basado en modelos, el número de ideas que se pueden explorar es prácticamente ilimitado. Un modelo captura los requisitos, los componentes del sistema, la propiedad intelectual y los escenarios de prueba. Dado que el modelo se puede simular, se pueden investigar cuestiones y problemas de diseño mucho antes de crear hardware costoso. De ese modo, se pueden evaluar rápidamente múltiples ideas de diseño, explorar tradeoffs y comprobar cómo cada cambio de diseño afecta al sistema.

Caso práctico: *Tessella*

Representación artística de la nave Solar Orbiter.

"Ya habíamos comprobado las ventajas del diseño basado en modelos en varios proyectos anteriores. En este proyecto, MATLAB y Simulink nos permitieron crear una especificación detallada para disminuir la diferencia entre los algoritmos de prototipo que desarrollamos, ajustamos y probamos en Simulink, y la implementación final del software".

— Andrew Pollard, Tessella

La misión Solar Orbiter del programa Cosmic Vision de la Agencia Espacial Europea consiste en responder cuestiones fundamentales sobre el sistema solar y el origen del universo. El subsistema de control de actitud y órbita (AOCS) se encargará de mantener la nave espacial y su escudo solar orientados hacia el sol, y de conservar una actitud concreta para optimizar la precisión de los instrumentos.

El subsistema AOCS debe ajustar continuamente la actitud de la nave espacial Solar Orbiter para que el escudo solar proporcione la máxima protección mientras la nave pasa cerca del Sol. Por razones de seguridad, AOCS no puede permitir que la nave se desvíe más de 6,5 grados respecto del Sol en ningún momento, incluso después de un fallo. Durante las observaciones científicas, la estabilidad de apuntamiento debe estar comprendida dentro de unas pocas décimas de segundo de arco.

Además de cumplir con estos requisitos, AOCS debe tener en cuenta los par motores de perturbación de la presión de la radiación solar, el gradiente de gravedad y las fuerzas aerodinámicas.

La estructura física de la nave agregó complicaciones al diseño del AOCS. El escudo solar crea una distribución de masa inusual que dificultaba la estabilidad. Además, múltiples apéndices flexibles, como los arrays de paneles solares, hacen que toda la estructura sea susceptible a resonancia.

El equipo de ingeniería de Tessella demostró su idoneidad en codificación y verificación formales utilizando el diseño basado en modelos para diseñar, modelar, simular y realizar ajustes preliminares de los algoritmos. Con Simulink, el equipo modeló los sistemas de accionamiento de la nave, incluyendo sus cuatro ruedas de reacción y propulsores químicos. Para lograr un control preciso de los propulsores, el equipo de trabajo desarrolló y modeló un algoritmo de comando del actuador utilizando modulación por ancho de pulsos.

Sistema de energía

Profesionales de ingeniería de sistemas de energía utilizan MATLAB y Simulink para ejecutar simulaciones con el fin de analizar el perfil de energía de las misiones, predecir los efectos del deterioro de las baterías en el sistema y realizar un diseño detallado de componentes eléctricos tales como convertidores CC-CC.

Pueden modelar rápidamente componentes y sistemas eléctricos, tales como arrays de paneles solares y reguladores de tensión, utilizando los bloques proporcionados, o bien crear bloques personalizados cuando el diseño lo requiera. Luego, pueden simular el modelo para solucionar los complejos sistemas de ecuaciones subyacentes sin necesidad de escribir código de bajo nivel, y visualizar los resultados inmediatamente. También pueden incluir efectos térmicos y de actitud en los modelos para realizar una simulación multidominio en un único entorno.

Caso práctico: *Lockheed Martin*

Nave espacial Orion de la NASA.

"Con Simscape Electrical creamos un modelo de sistema de potencia integrado que conecta el dominio eléctrico y el térmico, lo que nos permite tener una perspectiva integral durante las simulaciones de misiones. Si necesitamos modelar los motores que giran los paneles solares, también podemos integrar esos componentes mecánicos".

— Hector Hernandez, Lockheed Martin

El grupo de ingeniería de Lockheed Martin desarrolló un modelo de sistema de energía para el programa Constellation de la NASA, utilizando Microsoft® Excel® y Visual Basic® para Aplicaciones (VBA). Sin embargo, a medida que se examinaban más casos de prueba, el modelo se ralentizaba y, a veces, se bloqueaba. Dado que se trataba de un modelo de energía de un solo nodo, no se lo podía utilizar para evaluar las caídas de tensión en todo el sistema o los requisitos de calidad de la energía.

El equipo de ingeniería de Lockheed Martin utilizó Simulink y Simscape Electrical para modelar y simular el sistema de energía de la nave espacial Orion. Para calcular con precisión una curva de corriente-voltaje (IV) para el array de paneles solares, el modelo tiene en cuenta el número de celdas por ala, los ángulos de apuntamiento del array, las sombras y los efectos térmicos, tales como energía solar, infrarrojo planetario y albedo. También se consideraron factores de degradación del rendimiento, tales como radiación, contaminación y colisiones con micrometeoroides y desperdicio espacial.

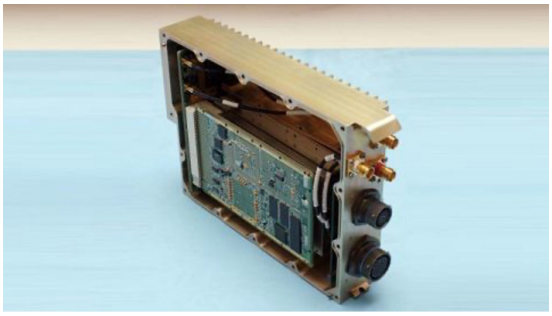
El modelo de la batería de iones de litio tiene en cuenta los efectos de histéresis, temperatura y estado de carga de la batería. El grupo de ingeniería también creó un bloque de carga de energía personalizado para simular las cargas variables generadas por la propulsión, la orientación y otros sistemas de la nave espacial mientras se activan y desactivan durante una misión. Utilizando el modelo de sistema de energía integrado, el equipo de ingeniería ejecutó simulaciones de diversos perfiles de misión. Supervisaron los cambios de misión y carga para evaluar impactos relacionados con la energía durante el desarrollo, escenarios de fallos simulados, incluyendo fallos de baterías, atascos de interruptores, y fallos del array de paneles solares en las fases de ascenso y órbita. Los resultados se utilizaron como guía para el desarrollo de oportunidades de misiones y protocolos para casos de fallos.

Sistema de comunicaciones

Profesionales de ingeniería de sistemas de comunicaciones utilizan MATLAB y Simulink como entorno de diseño común para desarrollar, analizar e implementar sistemas de comunicaciones de naves espaciales. Con MATLAB y Simulink, pueden realizar prototipado de elementos de la cadena de señales, incluidos elementos digitales, de RF y de antenas. Luego, pueden combinar la labor de varios equipos de trabajo en un modelo ejecutable en el nivel de sistema.

También pueden explorar rápidamente las imperfecciones en el nivel de sistema y examinar escenarios hipotéticos que resultan difíciles de reproducir en laboratorio. A medida que el diseño madura, pueden generar automáticamente código C para procesadores integrados, o código HDL para FPGA.

Caso práctico: *BAE Systems*



Placa personalizada utilizada en el flujo de trabajo de diseño tradicional.

"Un ingeniero con varios años de experiencia en código VHDL necesitaba 645 horas para codificar manualmente una forma de onda SDR totalmente funcional. Otro ingeniero con menos experiencia completó el mismo proyecto con Simulink y System Generator de Xilinx en menos de 46 horas".

— Dr. David Haessig, BAE Systems

BAE Systems es una empresa de vanguardia en tecnología SDR. Su flujo de diseño tradicional se basa en codificación manual de FPGA en VHDL®. Recientemente, surgió la posibilidad de comparar su enfoque tradicional con el diseño basado en modelos utilizando herramientas de MathWorks y Xilinx®. Ejecutaron dos proyectos de desarrollo de formas de onda SDR en paralelo y descubrieron que Simulink® y System Generator de Xilinx reducían drásticamente el tiempo de desarrollo.

Cuando la empresa se contrató para desarrollar una forma de onda de comunicación satelital estándar militar (MIL-STD-188-165A) para su implementación en una radio de comando, control, comunicaciones, computadores, inteligencia, vigilancia y reconocimiento (C4ISR), aprovecharon la oportunidad para evaluar un nuevo flujo de diseño para reducir el tiempo de desarrollo.

Utilizando Xilinx, BAE Systems aplicó el diseño basado en modelos con Simulink y System Generator de Xilinx para diseñar y desplegar una forma de onda SDR MIL-STD-188 10 veces más rápido que con el enfoque tradicional de codificación manual. Tras el éxito de este proyecto inicial, BAE Systems colaboró con MathWorks, Virginia Tech, Xilinx y Zeligsoft para mejorar la portabilidad de las formas de onda. Este grupo está desarrollando una interfaz para incorporar el código generado por Simulink Coder™ o System Generator de Xilinx directamente en radios de Software Communications Architecture (SCA).

Orientación, navegación y control

Con MATLAB y Simulink, ingenieros de sistemas de control pueden probar los algoritmos de control con modelos de planta antes de la implementación, de modo que pueden realizar diseños complejos sin necesidad de emplear prototipos costosos. Además, pueden crear diseños para múltiples configuraciones físicas, como la arquitectura de bus común del diseño de un satélite. En un entorno único, es posible:

- Crear y compartir modelos de GNC
- Integrar y simular los efectos en el nivel de sistema de cambios en diseños mecánicos y de sistemas de control
- Reutilizar código de vuelo generado automáticamente y casos de prueba
- Integrar nuevos diseños con herramientas y diseños existentes

Caso práctico: *Lockheed Martin Space Systems*

El observatorio IRIS.

El observatorio Interface Region Imaging Spectrograph (IRIS) se encuentra actualmente en órbita terrestre, capturando espectros ultravioleta e imágenes de alta resolución del Sol. Las imágenes capturadas ayudarán a profesionales científicos a comprender mejor el flujo de energía y plasma en los niveles más bajos de la atmósfera solar.

> "Un equipo de ingeniería de cuatro integrantes diseñó, integró y probó el sistema GNC en solo 23 meses. Nuestra eficiencia aumentó con el uso de las mismas herramientas tanto para el análisis como el desarrollo de código, y logramos generar 20.000 líneas de código sin errores. Esto nos convenció de usar el diseño basado en modelos".
>
> — *Vincentz Knagenhjelm, ingeniero de GNC, Lockheed Martin Space Systems*

En proyectos similares anteriores, el equipo de ingeniería de Lockheed Martin solía crear extensos documentos de diseño de algoritmos, algunos con más de 1000 páginas, y los desarrolladores escribían código manualmente basándose en su interpretación de estos documentos. El proceso entero era lento, y a veces se introducían errores durante la codificación manual. Con un plazo de tan solo 23 meses para diseñar, integrar y realizar pruebas de software, fue necesario acelerar sustancialmente el proceso de entrega de software.

Con el diseño basado en modelos, el grupo de ingeniería de Lockheed Martin aceleró el desarrollo del software de vuelo GNC de IRIS. Utilizando MATLAB y Simulink, desarrollaron un modelo básico del sistema de control para analizar el rendimiento del apuntamiento y de la precisión con la que reorientar la nave espacial.

Verificaron el diseño inicial de GNC ejecutando simulaciones de lazo cerrado con el modelo de planta y realizando análisis de cobertura del modelo en las simulaciones con Simulink Coverage™. Utilizaron Embedded Coder para generar código C para cada componente, agregando una pequeña cantidad de código manual como "puente" para un microprocesador Moog Broad Reach Engineering endurecido con radiación y su software ejecutivo. Con una interfaz de usuario personalizada de MATLAB, el equipo de trabajo ejecutó diversos casos de prueba de Simulink para cada unidad de software de vuelo GNC.

Caso práctico: *El alunizaje de Apolo 11: diseño de naves espaciales, pasado y presente*

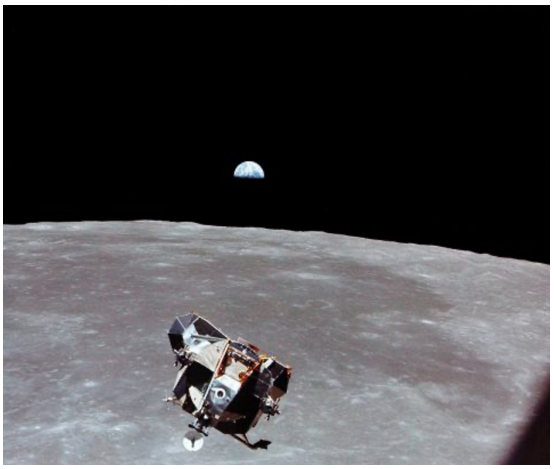


Imagen cortesía de la NASA.

Los astronautas Neil Armstrong y Buzz Aldrin a bordo del Apolo 11 alunizaron hace más de 50 años. Para conmemorar ese evento trascendental y celebrar los programas dedicados a futuros alunizajes, repasamos el relato de primera mano sobre el diseño del piloto automático digital del módulo lunar que Richard J. Gran publicó en 1999. En ese artículo, Richard describía el enfoque que aplicó en la década de los 60 y lo comparaba con la forma en que podría emplearse el diseño basado en modelos con MATLAB y Simulink para diseñar sistemas GNC en 1999. En los 20 años desde que Richard escribió su artículo, el diseño basado en modelos ha evolucionado sustancialmente. Para resaltar esta evolución, describimos cómo se pueden aplicar MATLAB y Simulink al diseño de sistemas GNC en la actualidad.

Lea el artículo para obtener más información, y descubra el modelo de sistema de Simulink que Richard Gran desarrolló al replantearse el piloto automático digital del módulo lunar.

Procesamiento de datos e imágenes, y visión artificial

Los sistemas de detección basados en visión artificial permiten un mayor grado de autonomía y precisión de la navegación en misiones espaciales. Con la navegación relativa y óptica de alta precisión, los sistemas de detección basados en visión artificial desempeñan un papel fundamental en:

- Operaciones de proximidad y reunión en órbita (RPO)
- Entrada, descenso y aterrizaje (EDL)
- Exploración robótica del sistema solar

Los requisitos cada vez más ambiciosos de las misiones de hoy en día, junto con un sector privado dinámico e innovador, generan un mayor caudal de investigaciones en técnicas de detección y percepción basadas en visión artificial que utilizan IA con Machine Learning.

Lea este white paper para obtener más información sobre:

- Detección basada en visión artificial para facilitar la autonomía de naves espaciales
- Cómo las tendencias de Machine Learning en el ámbito espacial están a punto de afectar a la inteligencia artificial (IA) de naves espaciales
- Cómo utilizar las rutinas de MATLAB y Simulink para centrarse en el diseño de nivel superior

Caso práctico: *Cornell University*

Dispositivo de análisis acústico utilizado por el programa Bioacoustics Research Program (BRP) para recopilar datos de grandes ballenas barbadas y otros mamíferos marinos. Foto cortesía de Dimitri Ponirakis.

"El cálculo de alto rendimiento con MATLAB nos permite procesar datos de big data no analizados previamente. Interpretamos los datos para comprender cómo la especie humana afecta a los ecosistemas y así informar a los responsables de toma de decisiones sobre las actividades de los seres humanos en el océano y en la tierra".

— Dr. Christopher Clark, Cornell University

Durante más de 30 años, personal científico ha estudiado las poblaciones de fauna local grabando sonidos de animales en océanos, selvas, bosques y otros entornos naturales. Utilizan los resultados para evaluar el efecto del ruido causado por los seres humanos en entornos naturales, monitorizar poblaciones de animales en peligro de extinción e investigar la comunicación animal. Los sistemas de monitorización acústica pasiva graban sonidos de forma continua, lo que genera terabytes de datos. A veces no es posible procesar ni siquiera el 1% de estos datos, por falta de algoritmos avanzados y la capacidad de procesamiento necesarios.

La variabilidad de sonidos en animales de la misma especie es otra complicación adicional. Los datos con ruido y la variabilidad aumentan el número de falsos positivos y negativos, lo que reduce la precisión de los algoritmos de detección. Procesar los cientos de terabytes de datos recopilados supone otro desafío. Un proyecto normal implica procesar años de datos acústicos no procesados, hasta 10 TB, grabados en varios canales. Cada canal puede capturar cientos de millones de eventos, sonidos que se destacan cuando los datos se visualizan en forma de espectrograma. Los algoritmos probados en muestras pequeñas y de alta calidad a menudo son considerablemente menos precisos cuando se aplican a conjuntos de datos más grandes y con más ruido.

Los analistas de datos del programa BRP utilizaron MATLAB para desarrollar software de cálculo de alto rendimiento (HPC) para procesar datos acústicos automáticamente. Un proyecto de detección-clasificación comienza recopilando clips de audio del animal que se desea detectar, clips de ruido de fondo del entorno y archivos MAT de datos acústicos archivados. Utilizando MATLAB, se desarrollan algoritmos nuevos o perfeccionan los existentes para detectar secuencias de audio en los datos archivados similares a las del catálogo de clips. El equipo de trabajo de BRP desarrolló una interfaz de MATLAB que permite especificar los algoritmos, los conjuntos de datos y el número de procesadores. Además de los algoritmos de detección y clasificación, BRP utiliza MATLAB para análisis de ruido y modelado acústico, donde se capturan y simulan los efectos de dispersión del tiempo y la frecuencia de los entornos marinos o terrestres.

Generación de código

En un flujo de trabajo tradicional, el código embebido se desarrolla manualmente a partir de modelos de sistema o desde cero. Profesionales de ingeniería de software desarrollan algoritmos de control basándose en las especificaciones redactadas por ingenieros de sistemas de control. Cada paso de este proceso, que incluye redactar la especificación, codificar manualmente los algoritmos y depurar el código escrito a mano, puede requerir mucho tiempo y ser proclive a errores.

En lugar de desarrollar miles de líneas de código manualmente, con el diseño basado en modelos, el código se genera directamente a partir del modelo. Así, el modelo actúa como puente entre la ingeniería de software y la ingeniería de sistemas de control. El código generado se puede utilizar para prototipado rápido o producción.

El prototipado rápido ofrece una forma económica y rápida de probar algoritmos en hardware en tiempo real y realizar iteraciones del diseño en minutos, en lugar de semanas. Se puede utilizar prototipo de hardware o una unidad ECU de producción. Con los mismos modelos de diseño y hardware de prototipado rápido, se pueden realizar pruebas de hardware-in-the-loop y otras actividades de prueba y verificación para validar diseños de hardware y software antes de implementar en producción.

La generación de código de producción convierte el modelo en el código real que se implementará en el sistema de producción integrado. El código generado se puede optimizar para arquitecturas de procesadores específicos e integrarse con código desarrollado manualmente ya existente.

Caso práctico: *Swedish Space Corporation*

Representación artística de SMART-1, camino a la Luna.

"Desarrollamos el sistema AOCS de SMART-1 con éxito, en un plazo de tiempo muy corto y con un presupuesto muy bajo. Las herramientas de MathWorks para la simulación y la generación de código de vuelo desempeñaron un papel clave, y servirán como base para futuros programas de satélites, como Prisma".

— Per Bodin, Swedish Space Corporation

Swedish Space Corporation (SSC) desarrolló el sistema de control de actitud y órbita (AOCS) de Small Missions for Advanced Research and Technology (SMART-1) utilizando código de vuelo generado automáticamente. El sistema AOCS orienta la nave espacial para la propulsión vectorial, el apuntamiento de instrumentos científicos, y los arrays de paneles solares para que estén siempre iluminados por el sol. También controla la alineación del vector de propulsión eléctrica dentro del cuerpo de la nave espacial durante las fases de transferencia y descenso lunar, al tiempo que proporciona un sistema avanzado de detección de fallos, aislamiento y recuperación (FDIR).

SSC necesitaba desarrollar un sistema AOCS en una misión con un perfil de bajo coste, estándares de desarrollo de software estrictos y un ciclo de desarrollo de software de menos de dos años. Dado que AOCS debía funcionar en un entorno espacial inclemente, con radiación intensa, gravedad mínima y otros efectos que no se podían probar en un laboratorio o en tierra, SSC requería rigurosas pruebas en cadena para garantizar que el sistema funcionara correctamente durante el vuelo.

SSC implementó un nuevo proceso de desarrollo basado en herramientas de MathWorks para el diseño basado en modelos con objeto de modelar, simular, generar código automáticamente y probar el software AOCS incorporado. El equipo de ingeniería desarrolló modelos de simulación precisos para predecir el comportamiento del sistema y crear casos exhaustivos de prueba del sistema y del software, conformes con el estándar de desarrollo de software ESA PSS-05. Realizaron pruebas del sistema de software en un entorno de simulación en tiempo real físico y analizó los resultados con MATLAB. También verificaron el AOCS en el nivel de sistema utilizando la nave espacial integrada. Estas pruebas incluyeron pruebas de lazo abierto y cerrado en el Centro Europeo de Investigación y Tecnología Espacial (ESTEC) de los Países Bajos.

Pruebas y verificación

En un flujo de trabajo de desarrollo tradicional, las pruebas y la verificación suelen realizarse en una fase avanzada del proceso, lo que dificulta la identificación y corrección de errores introducidos durante las fases de diseño y codificación.

En el diseño basado en modelos, las pruebas y la verificación se realizan durante todo el ciclo de desarrollo, comenzando con los requisitos y especificaciones de modelado, continuando por el diseño, la generación de código y la integración. Los requisitos se pueden crear en un modelo y realizar su seguimiento al diseño, las pruebas y el código. Los métodos formales ayudan a demostrar que el diseño cumple con los requisitos. Se pueden generar informes y artefactos, y certificar el software según estándares de seguridad funcional tales como ***NPR 7150.2*** (NASA Software Engineering Requirements) y ***ECSS-E-40*** (European Cooperation for Space Standardization, Space Engineering Software).

Caso práctico: *OHB*

Simulación de estrategias de GNC para vuelo en formación autónomo avanzado.

"Esos modelos evolucionaron en modelos de vuelo completo, que verificamos en simulaciones de lazo cerrado con un modelo de planta de Simulink. A partir de ahí, bastaba un clic para generar el código de vuelo".

— Ron Noteborn, OHB

A menudo, las misiones espaciales planificadas dependen de vuelos en formación autónomos, en los que una nave espacial se aproxima o vuela junto a otra. El proyecto Prisma, dirigido por OHB AG (OHB) en colaboración con las agencias espaciales francesa y alemana y la Universidad Técnica de Dinamarca, prueba y valida estrategias GNC para vuelo en formación autónomo avanzado.

El equipo de ingeniería de OHB utilizó el diseño basado en modelos para desarrollar algoritmos GNC, ejecutar simulaciones de lazo cerrado en tiempo real en el nivel de sistema y generar código de vuelo para los dos satélites de Prisma: Mango y Tango.

OHB dividió el diseño de GNC en operaciones de vuelo en formación, encuentro y proximidad. Probaron y analizaron las ideas de algoritmos en MATLAB antes de modelarlas para verificar los algoritmos en una simulación de lazo cerrado.

Generaron código a partir de los modelos GNC y del modelo de planta. Luego desplegaron el código de planta en Simulink Real-Time™ y compilaron el código GNC para el procesador LEON2 de destino integrado. Después, ejecutaron pruebas de hardware-in-the-loop (HIL) del sistema de Simulink Real-Time junto con el controlador LEON2 para verificar el funcionamiento en tiempo real de los algoritmos.

Introducción al diseño basado en modelos

Aun cuando los equipos de ingeniería tienen conciencia de las ventajas del diseño basado en modelos, no dejan de considerar los posibles riesgos y desafíos organizativos, logísticos o técnicos. En esta sección se abordan las preguntas frecuentes de los equipos de ingeniería al plantearse la adopción del diseño basado en modelos. Se proporcionan sugerencias y prácticas recomendadas que han ayudado a otros equipos a gestionar esta transición.

P. ¿Cómo se ven afectadas las tareas de ingeniería por la introducción del diseño basado en modelos?

R. El diseño basado en modelos no sustituye los conocimientos técnicos en materia de diseño de sistemas de control y arquitectura de software. Con el diseño basado en modelos, las tareas de ingeniería de sistemas de control abarcan desde proporcionar requisitos en papel hasta facilitar requisitos ejecutables en forma de modelos y código. Por otro lado, los equipos de ingeniería de software dedican menos tiempo a codificar software de aplicaciones y más a modelar arquitecturas, codificar SO, controladores de dispositivos y otro software de plataforma, y realizar integración del sistema. Tanto los equipos de sistemas de control como los de software influyen en el diseño en el nivel de sistema desde las primeras etapas del proceso de desarrollo.

P. ¿Qué sucede con el código existente?

R. El código existente puede formar parte del diseño; el modelo del sistema puede contener componentes modelados intrínsecos y existentes. Eso significa que se pueden introducir componentes existentes gradualmente y continuar realizando simulación del sistema, verificación y generación de código.

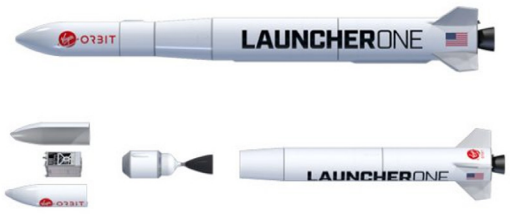
P. ¿Existe un procedimiento recomendado para adoptar el diseño basado en modelos?

R. Probar nuevos enfoques y herramientas de diseño siempre conlleva cierto riesgo. Los equipos de trabajo que han tenido éxito han mitigado este riesgo introduciendo el diseño basado en modelos gradualmente, dando pasos focalizados que ayudan a un proyecto sin ralentizarlo. Todas las organizaciones, independientemente de su tamaño, comienzan a adoptar el diseño basado en modelos en el nivel de grupo pequeño. Por lo general, comienzan con un solo proyecto que genere ganancia rápida y sirva de base para otros proyectos exitosos. Después de adquirir experiencia, implementan el diseño basado en modelos en el nivel de departamento, para que los modelos se conviertan en el centro de todo el desarrollo de sistemas integrados del grupo.

Estas cuatro prácticas recomendadas han funcionado bien en muchos equipos de trabajo:

- **Experimente con una pequeña parte del proyecto.** Una buena manera de comenzar es seleccionar una nueva área del sistema integrado, crear un modelo del comportamiento del software y generar código a partir del modelo. Un integrante del equipo puede hacer este pequeño cambio con una inversión mínima de tiempo para aprender nuevas herramientas y técnicas. Luego, puede utilizar los resultados para demostrar algunos beneficios fundamentales del diseño basado en modelos:
  - Se puede generar código de alta calidad automáticamente.
  - El código coincide con el comportamiento del modelo.
  - Simulando un modelo, se pueden corregir errores en algoritmos y obtener información más fácilmente que si se prueba el código C en un equipo de escritorio.
- **Aproveche modelos anteriores exitosos e incorpore simulación en el nivel de sistema.** Como se ha explicado en secciones anteriores, puede utilizar la simulación del sistema para validar requisitos, investigar cuestiones de diseño y realizar pruebas y verificaciones desde las primeras etapas de desarrollo. El modelo del sistema no tiene que ser de alta fidelidad; basta con que tenga suficientes detalles para garantizar que las señales de conexión tengan las unidades correctas y estén conectadas a los canales correctos, y que se capture el comportamiento dinámico del sistema. Los resultados de simulación proporcionan una visión preliminar del comportamiento de la planta y el controlador.
- **Utilice modelos para resolver problemas de diseño específicos.** Los modelos de la planta, el entorno y el algoritmo no tienen que ser a gran escala para obtener ciertos beneficios. Supongamos que necesita seleccionar parámetros para un solenoide utilizado en un accionamiento. Puede desarrollar un modelo sencillo que represente un "volumen de control" conceptual alrededor del solenoide, incluyendo qué lo impulsa y sobre qué actúa. Puede probar diversas condiciones de funcionamiento extremas y deducir los parámetros básicos sin necesidad de formular las ecuaciones. Luego, puede almacenar el modelo y emplearlo en un problema de diseño diferente, o en un proyecto futuro.
- **Comience con los elementos básicos del diseño basado en modelos.** Entre las ventajas inmediatas del diseño basado en modelos se incluyen la capacidad de crear modelos de componentes y sistemas, utilizar simulaciones para probar y validar diseños, y generar código C automáticamente para realizar prototipado y pruebas. Más adelante, puede utilizar herramientas y prácticas avanzadas, e introducir directrices de modelado, comprobación de conformidad automatizada, trazabilidad de requisitos y automatización de compilación de software.

Caso práctico: *Virgin Orbit*



Vehículo LauncherOne de Virgin Orbit totalmente ensamblado (arriba). Carenado, carga útil, y primera y segunda etapa (abajo).

"MATLAB y Simulink nos han ahorrado casi un 90% en costes, comparado con otras alternativas. Estos productos nos proporcionan flexibilidad de codificación para desarrollar nuestros propios módulos y comprender totalmente los supuestos establecidos, algo fundamental a la hora de informar resultados a otros equipos de trabajo".

— Patrick Harvey, Virgin Orbit

LauncherOne es el vehículo de lanzamiento de dos etapas de Virgin Orbit para transportar satélites pequeños a la órbita terrestre baja. Para reducir costes y aumentar la flexibilidad del lugar de lanzamiento, LauncherOne se diseñó para ser lanzado desde un avión de transporte 747-400 en vuelo. Cada misión implica varios eventos de separación cruciales: la separación del LauncherOne del avión de transporte, la primera etapa de la segunda etapa, el carenado de la segunda etapa, y la carga útil satelital de la segunda etapa.

Cuando el diseño estructural de LauncherOne aún estaba en desarrollo, el equipo de trabajo tuvo que considerar una serie de incógnitas en el análisis de los eventos de separación, tales como las propiedades de masa de cada componente, así como las fuerzas y características temporales de los impulsores neumáticos y de resorte utilizados para iniciar las separaciones. El grupo de trabajo tuvo que ejecutar miles de simulaciones Montecarlo mientras modificaba los valores desconocidos de los parámetros para comprobar si una determinada combinación de parámetros podría provocar una colisión.

El equipo de ingeniería de Virgin Orbit modeló y simuló los eventos de separación de las etapas y la carga útil con Simulink y Simscape Multibody, y utilizaron Parallel Computing Toolbox™ para ejecutar simulaciones en paralelo en procesadores multinúcleo. Utilizando Simulink con Simscape Multibody, construyeron un modelo preliminar que constaba de formas en 3D básicas, como esferas, conos y cilindros. En la actualidad, el equipo está trabajando en la simulación del evento de separación de lanzamiento en el aire, que incorporará un modelo de fuerzas y efectos aerodinámicos. También están perfeccionando el modelo a partir de los resultados de las pruebas en tierra del hardware de vuelo como preparación para el lanzamiento inaugural de la nave espacial.

Resumen

El desarrollo gira en torno a un modelo del sistema, desde la captura de requisitos hasta el diseño, la implementación y las pruebas. Esa es la base del diseño basado en modelos. Con ese modelo del sistema puede:

- Enlazar los diseños directamente con los requisitos
- Colaborar en un entorno de diseño compartido
- Simular múltiples escenarios hipotéticos
- Tomar decisiones de diseño correctas, basadas en simulación
- Optimizar el rendimiento en el nivel de sistema
- Generar automáticamente código de software integrado, informes y documentación
- Realizar pruebas y detectar errores desde las primeras etapas de desarrollo

Herramientas para diseño basado en modelos

Productos esenciales

MATLAB

Analice datos, desarrolle algoritmos y cree modelos matemáticos

Simulink

Modele y simule sistemas integrados

Captura y gestión de requisitos

Simulink Requirements

Cree, gestione y enlace requisitos con modelos, código generado y casos de prueba

System Composer

Diseñe y analice arquitecturas de sistemas y de software

Diseño

Simulink Control Design

Linealice modelos y diseñe sistemas de control

Stateflow

Modele y simule lógica de decisión con máquinas de estados y diagramas de flujo

Simscape

Modele y simule sistemas físicos multidominio

Satellite Communications Toolbox

Simule, analice y pruebe enlaces y sistemas de comunicación satelital

Generación de código

Simulink Coder

Generación de código C y C++ a partir de modelos de Simulink y Stateflow

Embedded Coder

Generación de código C y C++ optimizado para sistemas integrados

HDL Coder

Genere código VHDL y Verilog para diseños de FPGA y ASIC

Pruebas y verificación

Simulink Test

Desarrolle, gestione y ejecute pruebas basadas en simulaciones

Simulink Check

Verifique la conformidad con directrices de estilo y estándares de modelado

Simulink Coverage

Mida la cobertura de pruebas en modelos y código generado

Simulink Real-Time

Cree, ejecute y pruebe aplicaciones en tiempo real

Productos de Polyspace

Compruebe la ausencia de errores críticos en tiempo de ejecución

Más información

Nuestro sitio web mathworks.com dispone de una serie de recursos para progresar rápidamente con el diseño basado en modelos. Recomendamos que comience con estos:

Visión general

MATLAB y Simulink para sistemas espaciales

Tutoriales interactivos

MATLAB Onramp

Signal Processing Onramp

Image Processing Onramp

Machine Learning Onramp

Deep Learning Onramp

Simulink Onramp

Stateflow Onramp

Simscape Onramp

Diseño de sistemas de control Onramp

Reinforcement Learning Onramp

Webinars

Simulink para nuevos usuarios (36:05)

Diseño basado en modelos de sistemas de control (54:59)

Diseño de sistemas de control avanzado y de mayor alcance (51:03)

Inteligencia artificial y transformación digital en la ingeniería de sistemas espaciales (1:23:50)

Cursos de formación presenciales o a su ritmo

Fundamentos de MATLAB

Fundamentos de Simulink

Diseño de sistemas de control con MATLAB y Simulink

Recursos adicionales

MathWorks Consulting Services