MATLAB EXPO 2019

“软件定义一切”的机遇与挑战

MathWorks 宋胜凯

“软件”无处不在

The USS Makin Island.

The Alenia Aermacchi M-346.

The Bell 525 Ships 1 and 2 over the Palo Duro Canyon.

Airnamics co-founders Marko Thaler and Zoran Bjelić with the R5 MSN1 prototype after its first flight.

Airbus A380, the world’s largest commercial aircraft.

RMSV-designed Unmanned Combat Ground Vehicle.

HAWK surface-to-air missile launch.

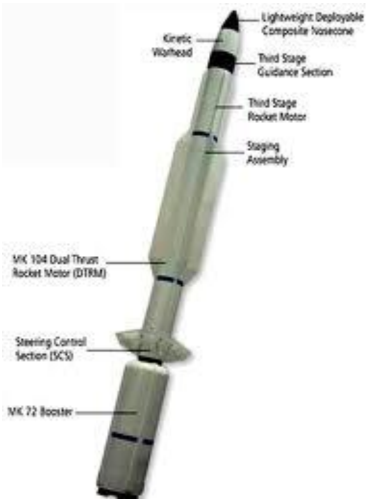
NASA’s Ares I rocket.

The IRIS observatory.

Artist's rendition of Mars rover.
Graphics courtesy of NASA/JPL/Cornell.

MBD无处不在

软件规模庞大并持续增长



Aegis missile system

SBIRS satellite

787 Dreamliner

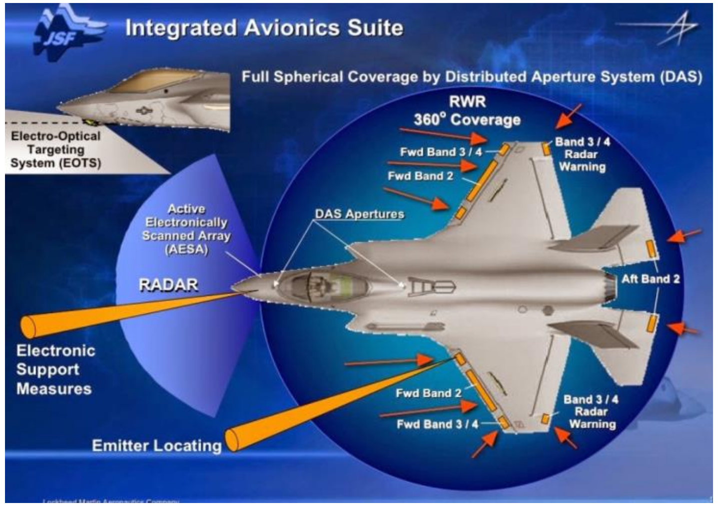
- SBIRS 卫星 ~25,000 行代码
- Aegis 导弹 ~400万行代码
- 787 梦想客机 ~ 650万行代码
- GM Volt ~ 1000万行代码

软件规模持续增长！

GM Volt

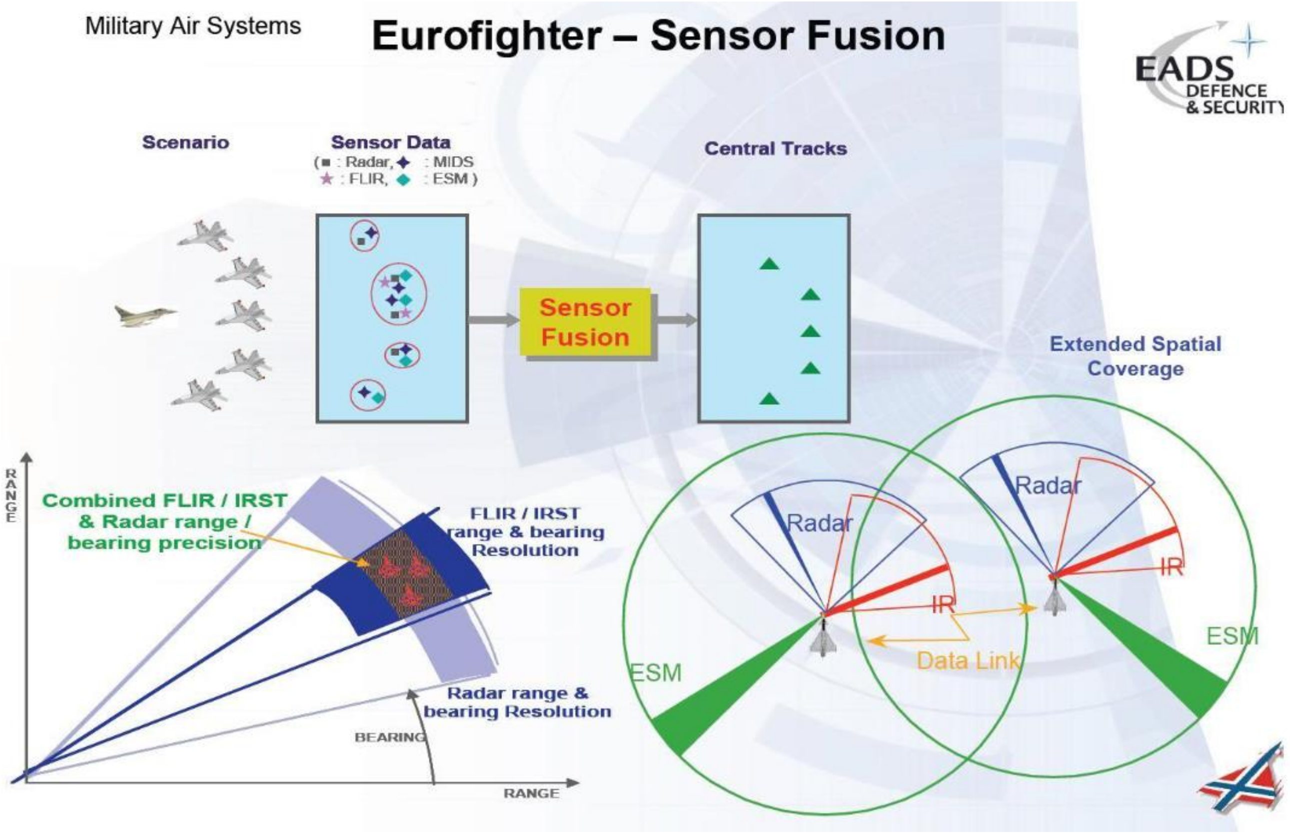
系统复杂度增加

多功能雷达系统：搜索、跟踪、监视、导引



多功能孔径：雷达、电子战、通信…

软件定义、高度综合！



多传感器系统：雷达、电子支援、光电/红外…

Reutech Radar System

- 基于模型的舰载海空搜索警戒雷达的系统开发

挑战

- 此舰载雷达的首要目标是它能够在动态变化的环境中，适应较大范围的多种海况。RRS团队必须在航海试验时收集到的数据基础上，对设计进行快速的更新和修正。
- 处理算法复杂，需要多片FPGA协同处理，算法验证将仅能在底层信息交互架构调试完成后，才能进行。

The RSR 210N multipurpose 2D radar system.

解决方案

- 使用MATLAB和Simulink工具，采用基于模型设计（MBD）的流程，来开发算法、模型的关键单元，继而完成系统级的仿真
- 使用Fixed-Point Designer把浮点数据模型设计自动转化为定点数据模型
- 从模型中直接生成HDL代码
- 在海试时，快速尝试、调整和定点化算法，并自动生成代码

结果

- 10分钟内自动生成**75，000行HDL代码**
- 开发时间减少4171小时（约两个工程师人*年）
- 信号处理模块可复用
- FPGA固件（firmware）高可靠性

“若不采用基于模型设计（MBD），要想按时完成本项目将会非常困难。使用HDL Coder自动生成HDL代码，以及将信号处理算法的设计与详细的硬件实现分离开来，这两项能力帮我们节省了两个工程师人*年。”

— *Kevin Williams, Reutech Radar System*

复杂系统开发

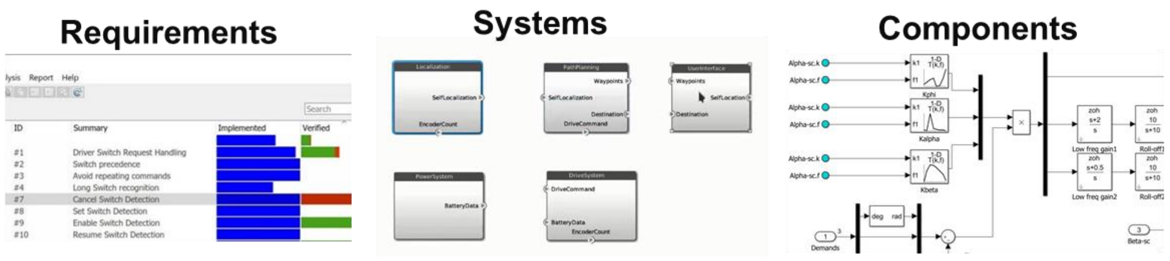
*“The current trend in system design is an increasing level of integration between aircraft functions and the systems that implement them. While there can be considerable value gained when integrating systems with other systems, **the increased complexity yields increased possibilities for errors**, particularly with functions that are performed jointly across multiple systems.”*

[ARP4754]

*“**Modeling, simulation, and prototyping used during architecture definition can significantly reduce the risk of failure** in the finished system. Systems engineers use modeling techniques and simulation on large complex systems to manage the risk of failure to meet system mission and performance requirements.”*

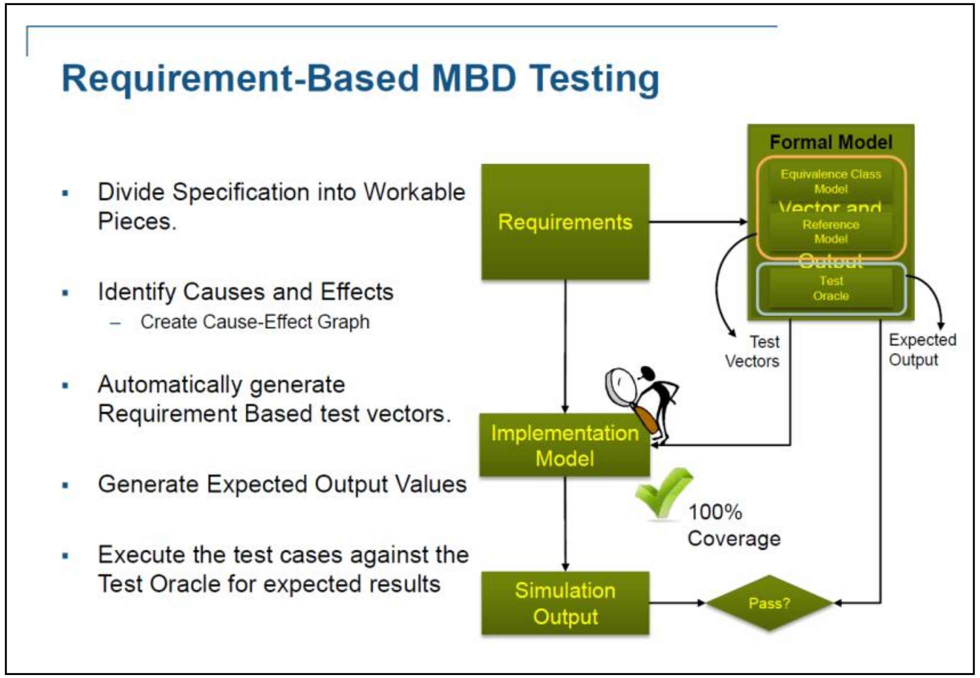
[INCOSE]

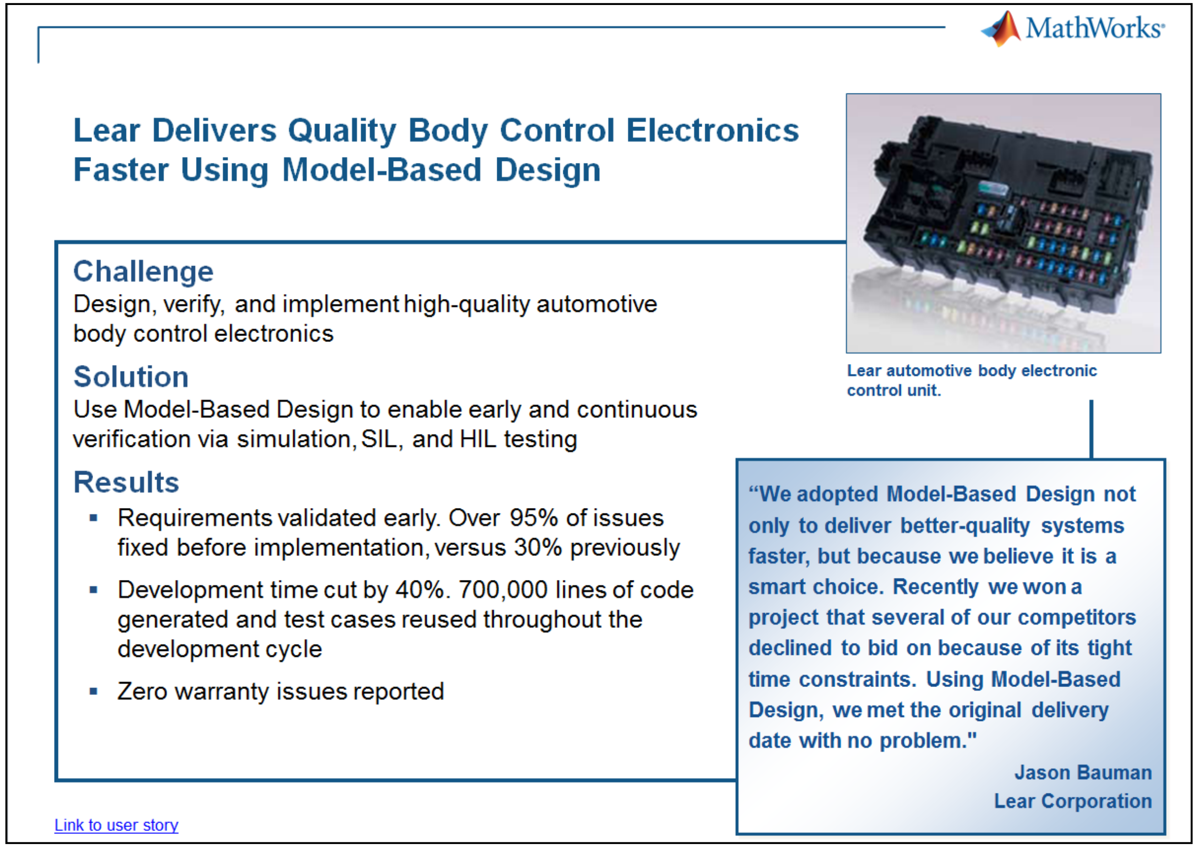
仿真(Simulation)
&
基于模型设计(MBD)



需求建模和验证

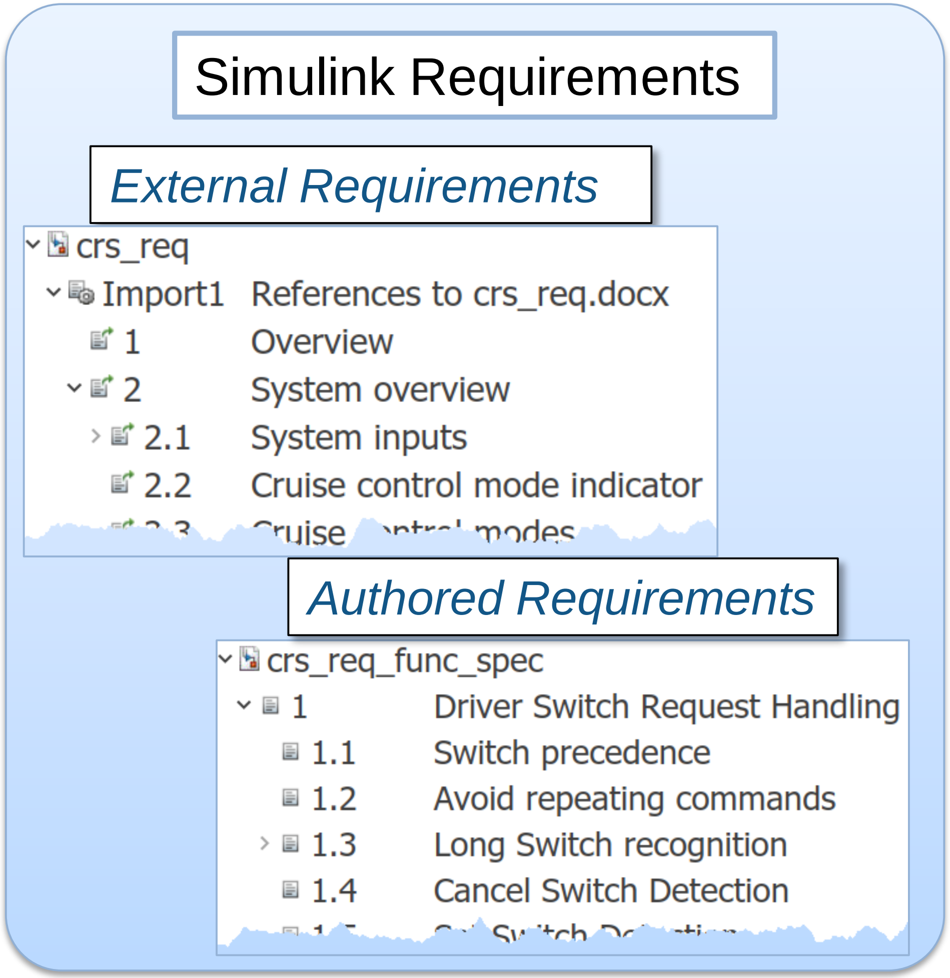
需求模型-基于需求的测试





高层抽象模型

需求的关联和追踪



- 导入需求:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- 与需求源同步更新
- 添加低层需求 R2019a
- 编辑需求
- 通过ReqIF输出
  - 与外部工具同步R2019a

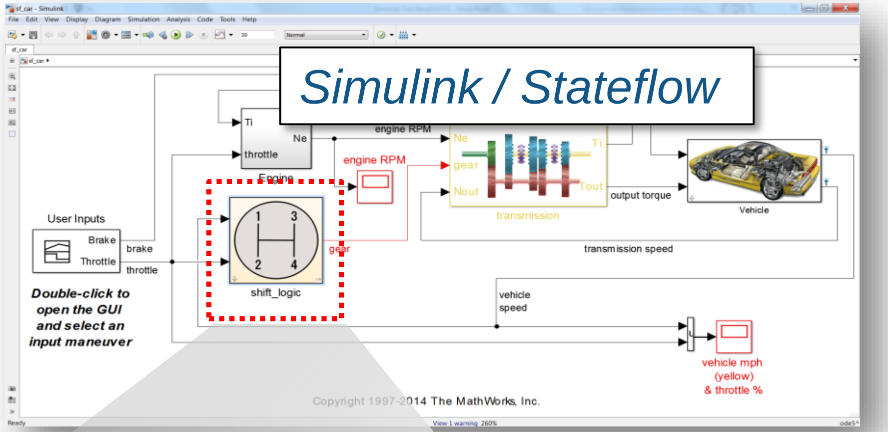
系统实现和需求验证

Requirements

TransmissionReq

1	Transmission Operating Modes
1.1	Reverse cannot be entered from drive
1.2	Engine only starts in Park

Implemented By



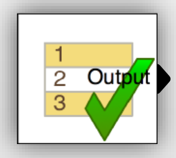
Verified By

Test Case

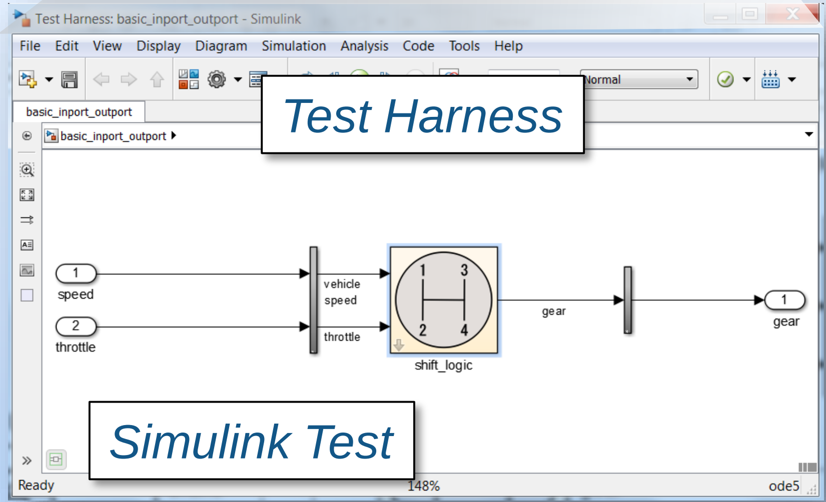
Inputs

MAT / Excel file (input)

Signal Editor

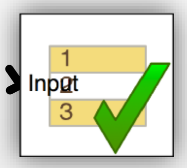


Test Sequence



Assessments

MAT / Excel File (baseline)



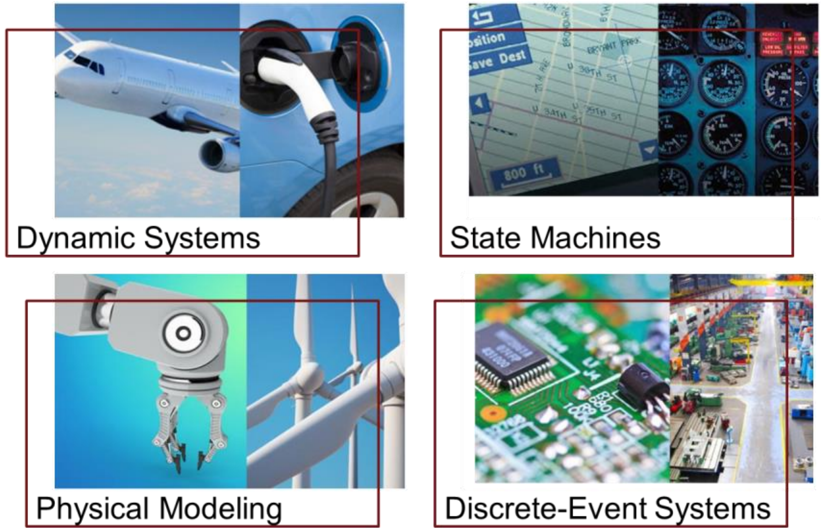
Test Assessments

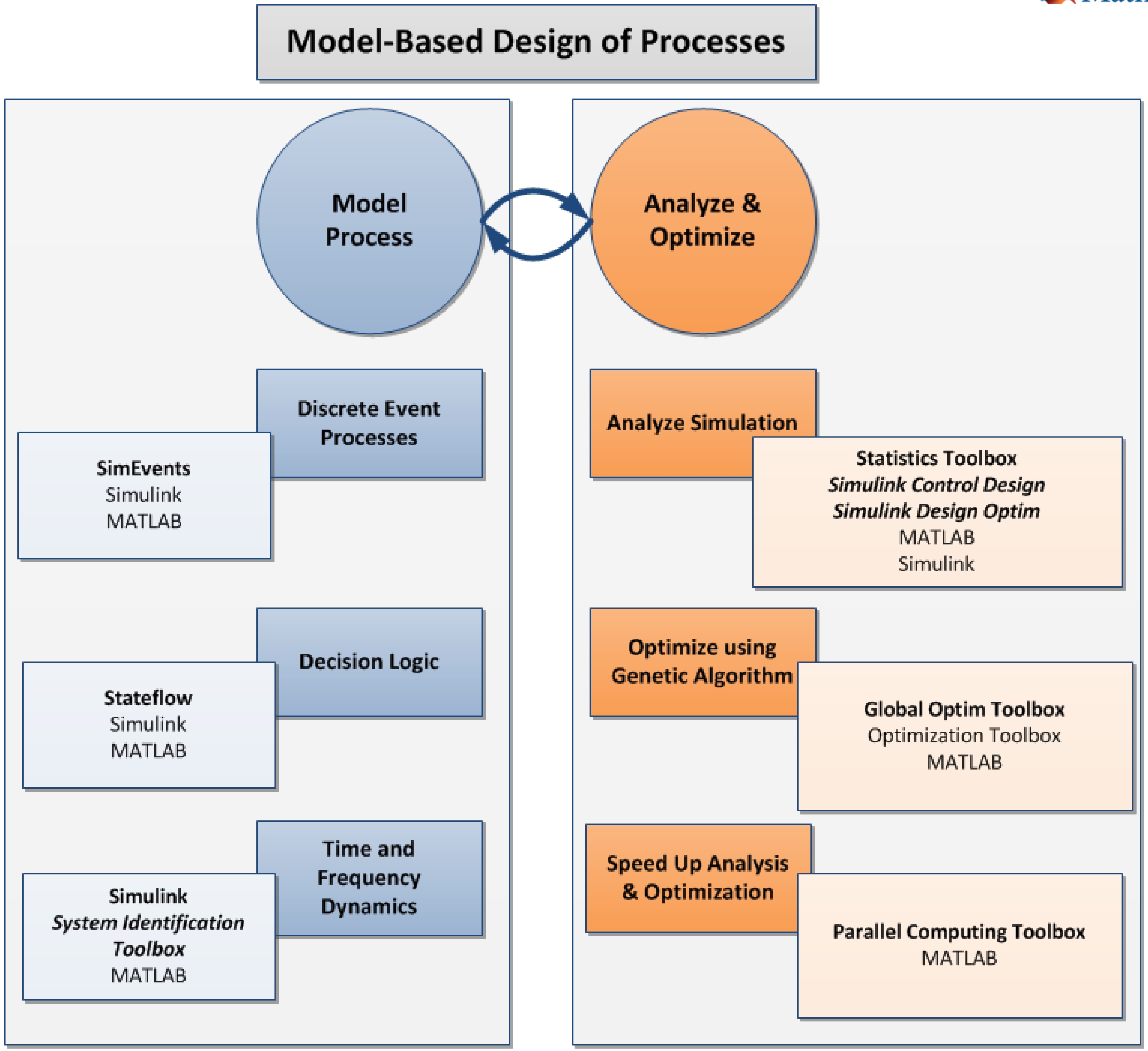
```
function customCriteria
Perform custom criteria
test.verifyThat(test.sl
```

MATLAB Unit Test

基于模型的设计过程

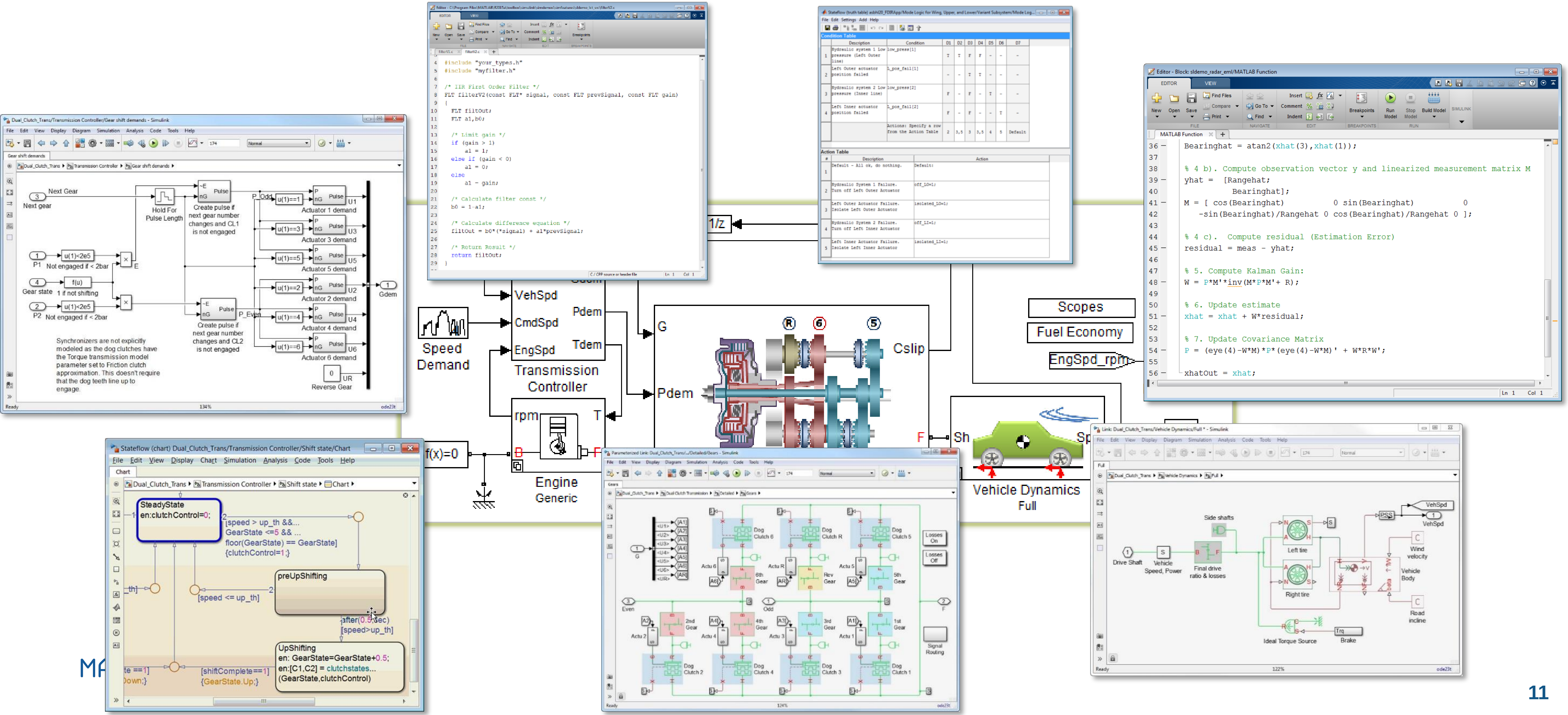
- 连续时间/离散时间系统建模
- 状态机建模
- 物理系统建模
- 离散事件系统建模





基于模型的设计过程-开发平台的特征

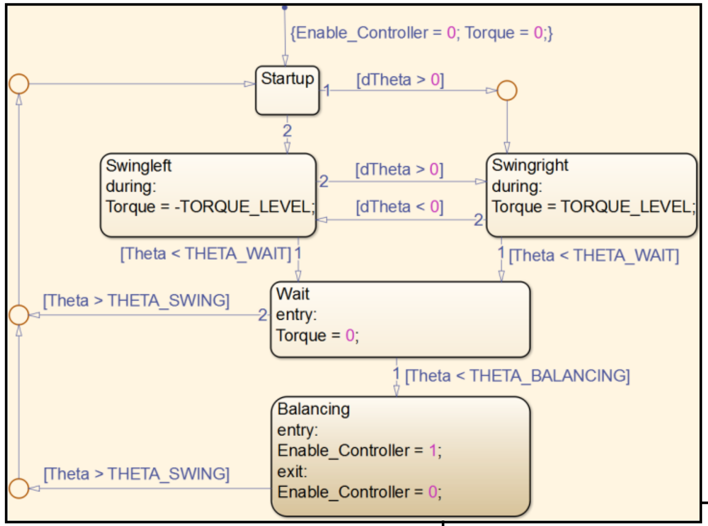
多域建模与多学科建模



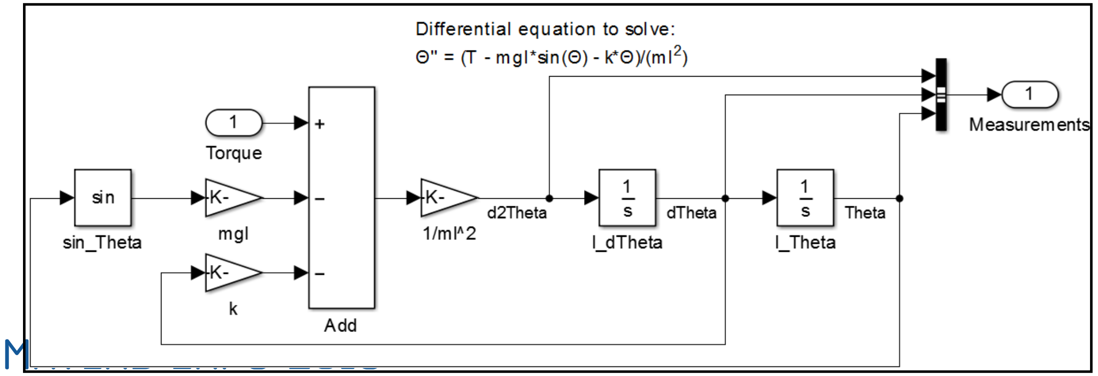
基于模型的设计过程——多种描述方式

```
function [symbols, weights] = gainctrl(rxsig, train)
% 1-tap adaptive equalizer using LMS or RLS algorithm

% Equalizer settings
lambda = 0.99;
Delta = 0.1+0i;
weights = 0+0i;

for n = 1:length(rxsig)
    u = rxsig(n);  % received sample
    y = conj(weights) * u;
    if n<=length(train)
        d = train(n);
    else
        d = detect(real(y)) + 1j*detect(imag(y));
    end
    % Single-tap RLS
    Delta = 1/(lambda/Delta + u*conj(u));
    G = Delta * u;
    e = d - y;  % symbol estimation error
    weights = weights + G*conj(e);
    symbols(n) = y;
end
```

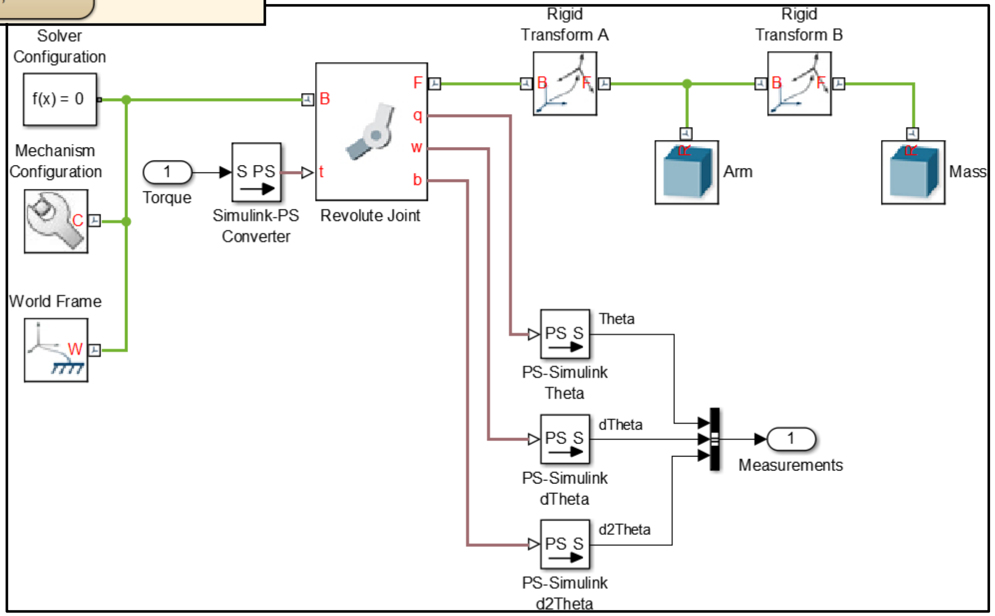
MATLAB



Stateflow



Simulink

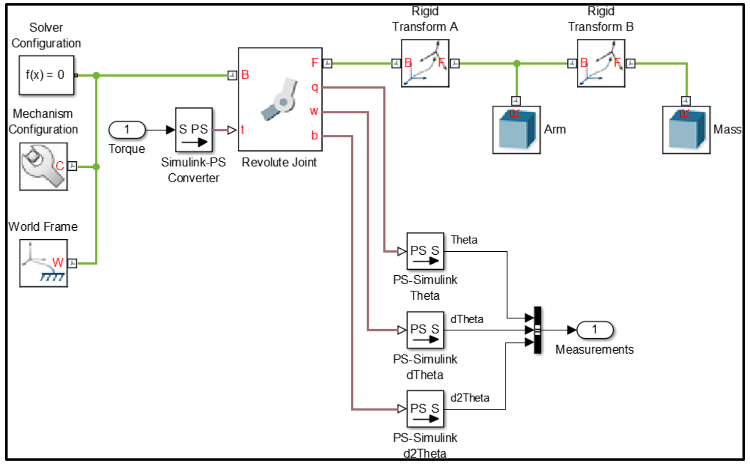


Simscape

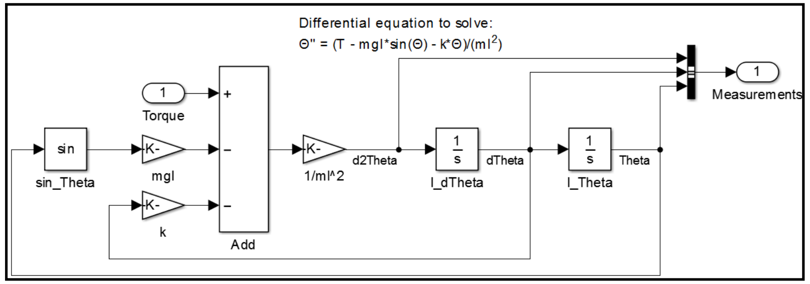
基于模型的设计过程——统一的自动代码生成环境

```
function [symbols, weights] = gainctrl(rxsig, train)
% 1-tap adaptive equalizer using LMS or RLS algorithm

% Equalizer settings
lambda = 0.99;
Delta = 0.1+0i;
weights = 0+0i;

for n = 1:length(rxsig)
    u = rxsig(n);  % received sample
    y = conj(weights) * u;
    if n<=length(train)
        d = train(n);
    else
        d = detect(real(y)) + 1j*detect(imag(y));
    end
    % Single-tap RLS
    Delta = 1/(lambda/Delta + u*conj(u));
    G = Delta * u;
    e = d - y;  % symbol estimation error
    weights = weights + G*conj(e);
    symbols(n) = y;
end
```

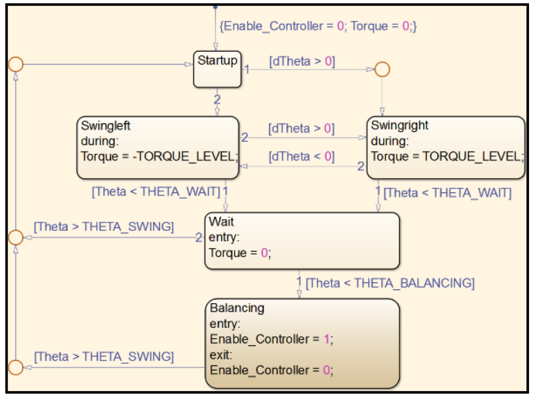
MATLAB



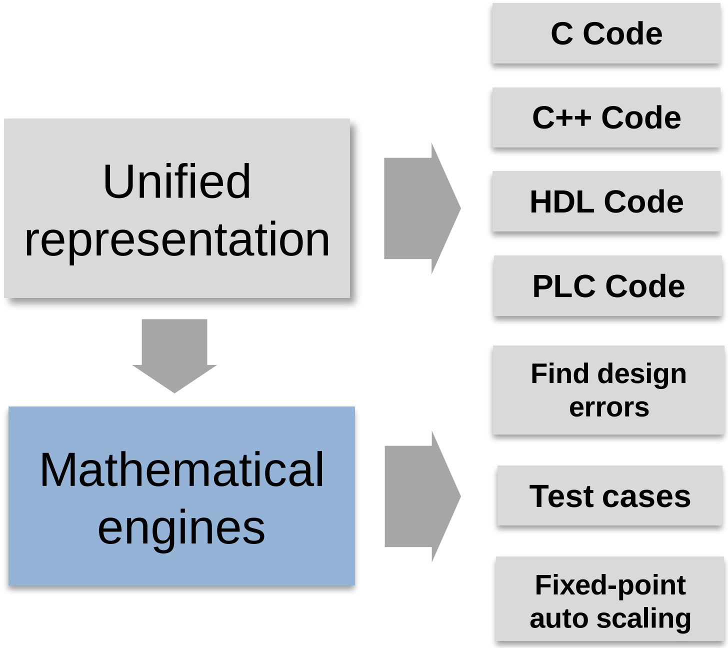
Simscape



Simulink

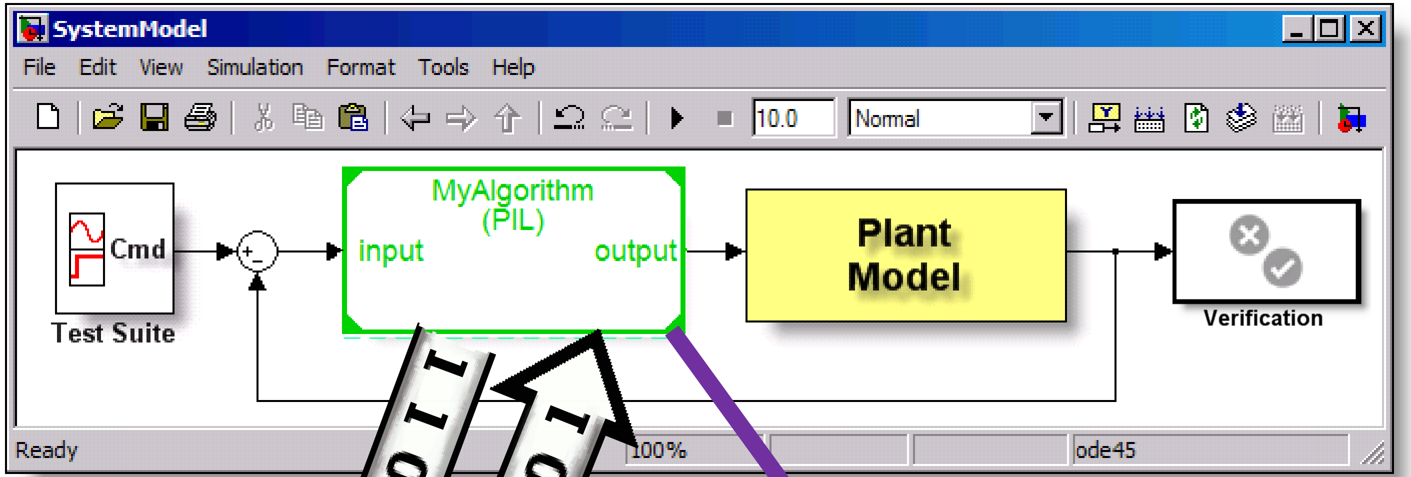


Stateflow

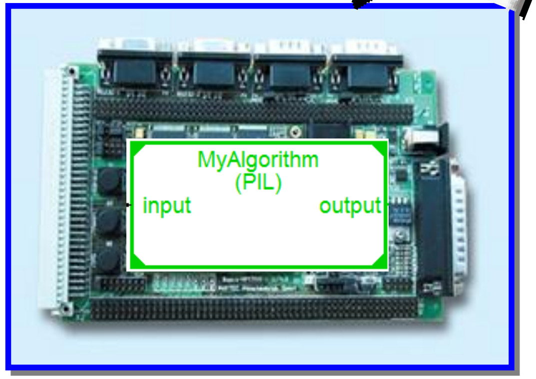


在环验证 In-Loop Verification

SIL and PIL



Non-Real-Time Synchronization with Host at Each Time Step

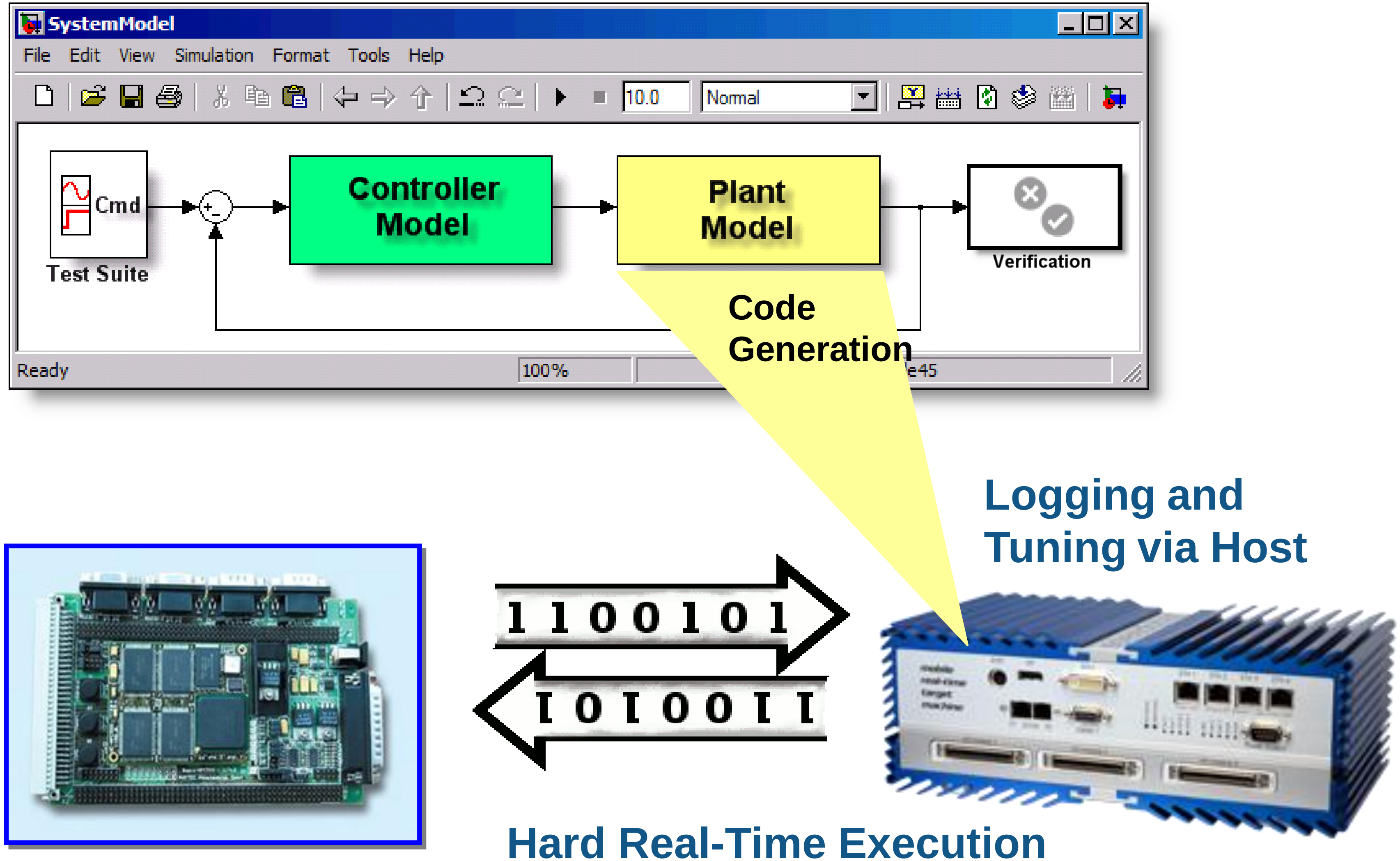


Execution History

- **Logged signal results comparison**
- **Code coverage**
- **Execution timing**

在环验证 In-Loop Verification

HIL, Rapid Prototyping

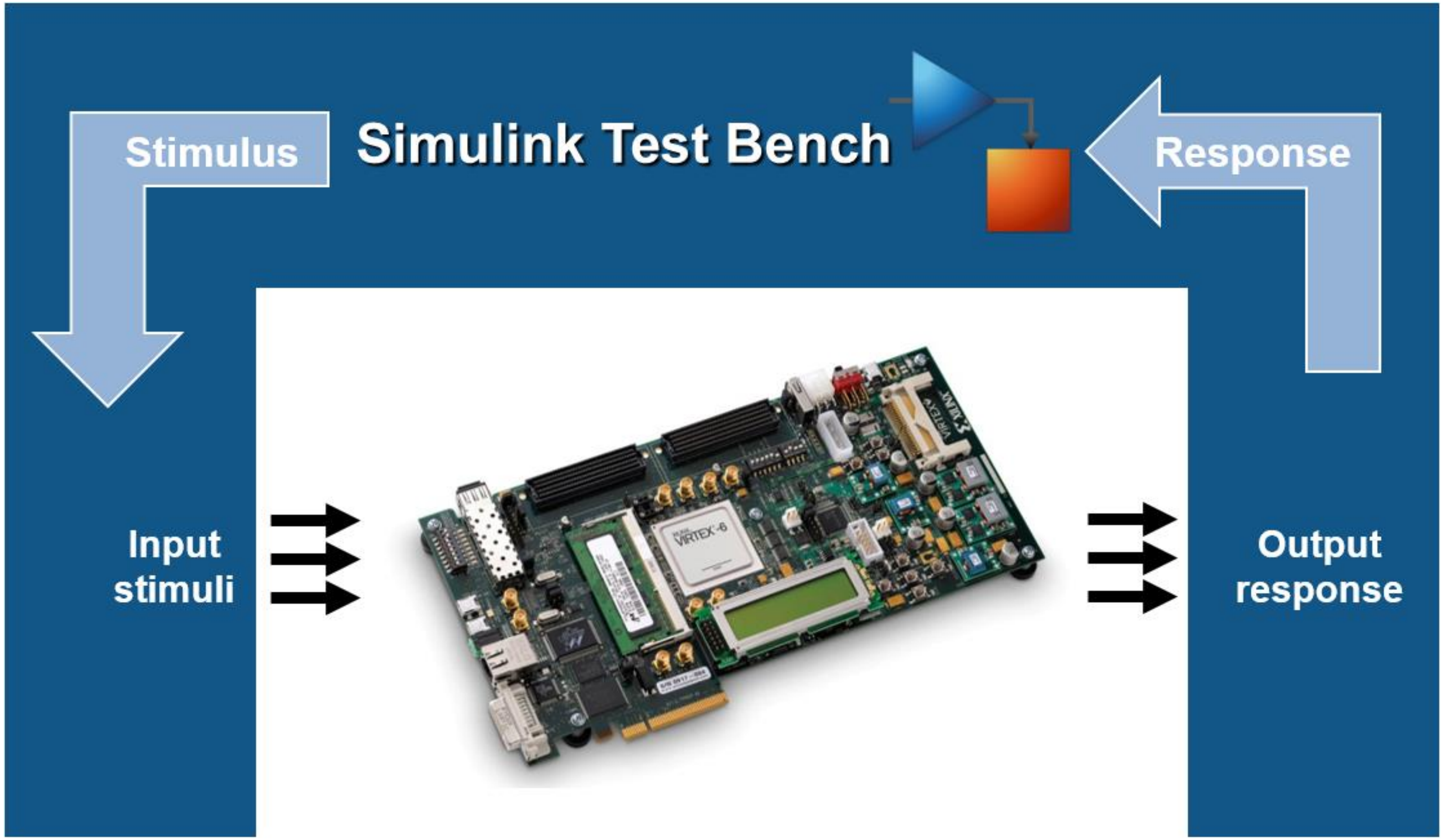


Logging and Tuning via Host

Hard Real-Time Execution

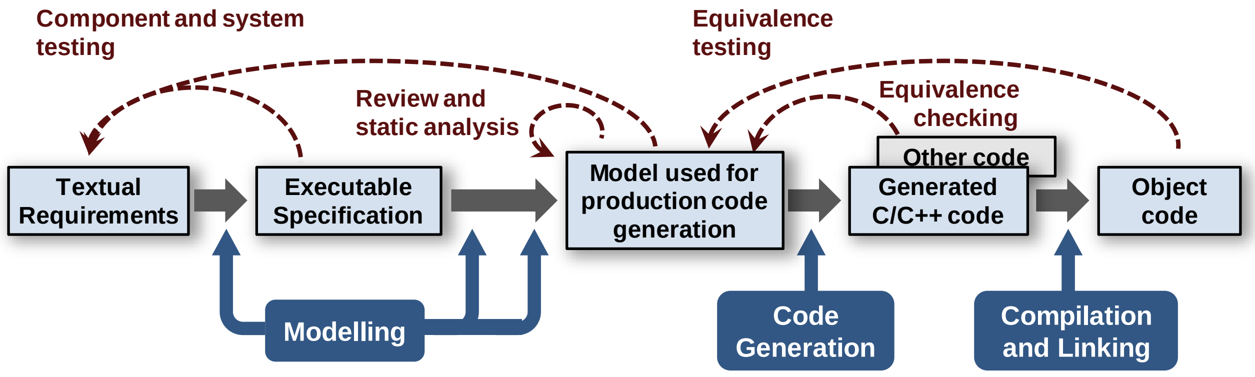
在环验证 In-Loop Verification

FIL, Test Bench Simulation

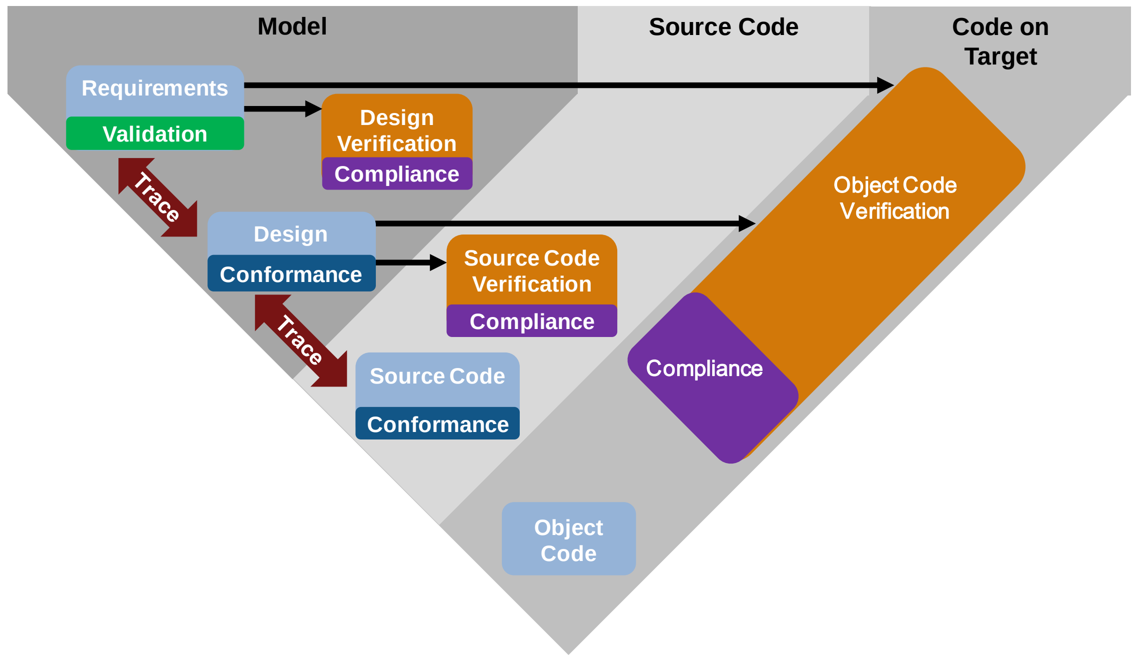


符合认证标准的工作流程

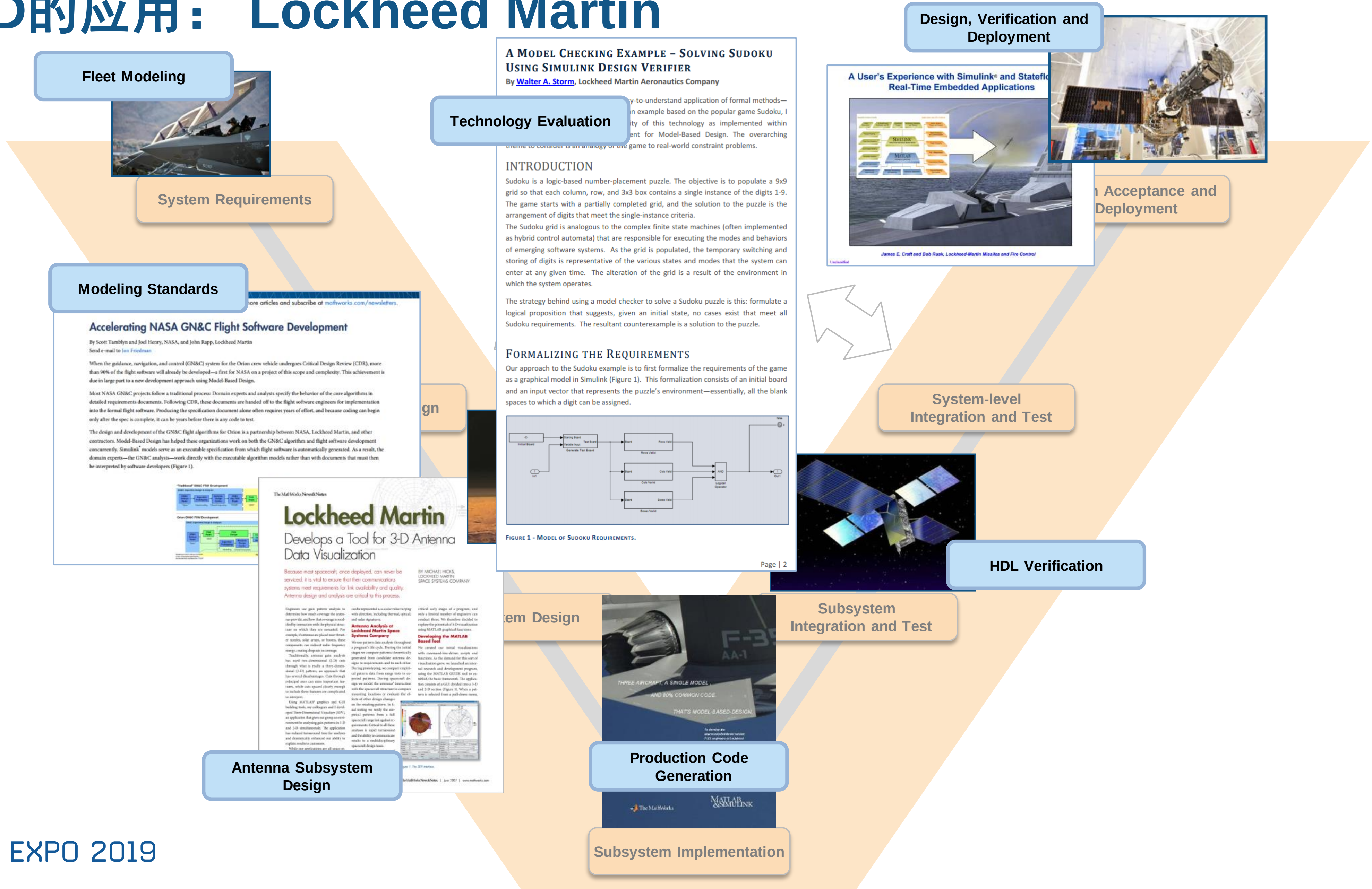
IEC 61508
ISO 26262
IEC 62405
EN 50128



DO-178C
DO-254
DO-331
DO-333



MBD的应用： Lockheed Martin



Simulate

early & often

MathWorks®

Accelerating the pace of engineering and science