

2020 MathWorks 中国汽车年会

自动驾驶开发中的MATLAB语言应用
及其符合功能安全要求的最佳实践

徐天皓

MathWorks中国区高级技术咨询顾问

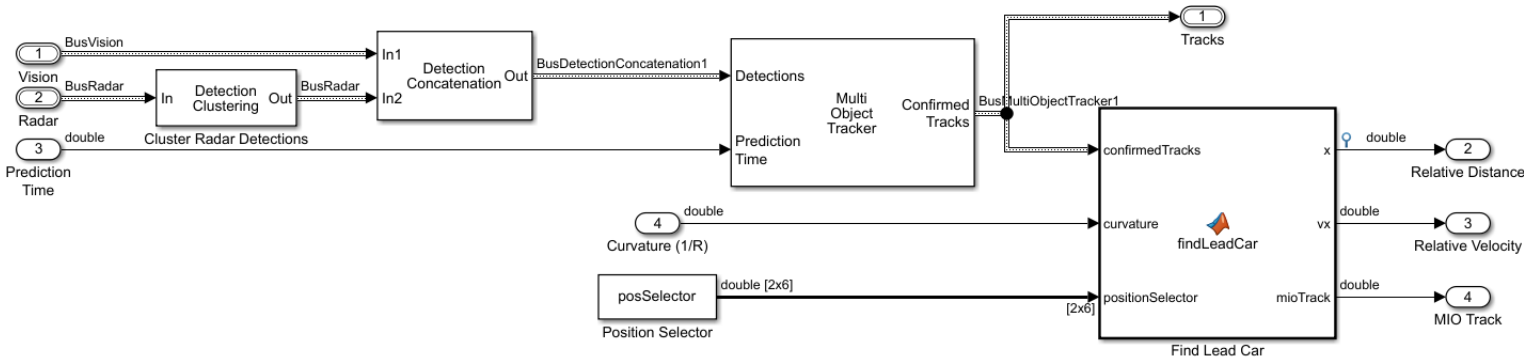


简介

- 我们已经知道Simulink和Stateflow [适合为ISO 26262 QM到ASIL-D的应用程序生成代码](#)
- MATLAB 已经逐渐被用于设计 AD/ADAS 的算法原型
 - 适用于矩阵，图像处理，深度学习的编程语言
 - [大量的专业算法函数](#)：包括传感器融合，统计和机器学习，图形处理、自动驾驶，深度学习等工具箱
 - 丰富的支持代码生成的[语言功能](#)
 - MATLAB语言也可以无缝集成到敏捷工作流工具中

MATLAB语言与自动驾驶

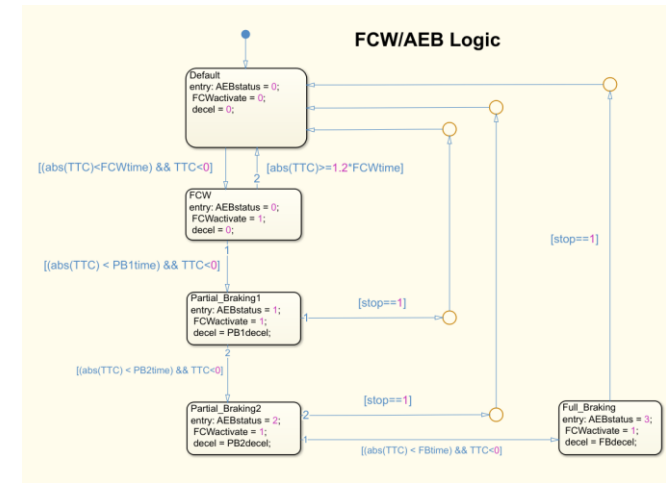
Tracking and Sensor Fusion



```

trackID = 0;
for i = 1:confirmedTracks.NumTracks
    position = positionSelector * confirmedTracks.Tracks(i).State;
    x = position(1); % Longitudinal position of the track relative to ego
    y = position(2); % Lateral position of the track relative to ego
    % No point checking otherwise
    if x < maxY && x > 0
        leftLane = polyval([curvature/2.0, halfLaneWidth],x); % Half a lane to the left
        rightLane = polyval([curvature/2.0, -halfLaneWidth],x); % Half a lane to the right
        % Find a new lead car
        if (yrightLane <= y) && (y <= yleftLane)
            maxX = x;
            trackID = i;
        end
    end
end
if trackID > 0
    mioState = confirmedTracks.Tracks(trackID).State;
else
    mioState = inf(miaStates,1,'like',confirmedTracks.Tracks(1).State);
end
% Output:
x = mioState(1); % Longitudinal position of the lead car
vx = mioState(2); % Longitudinal velocity of the lead car
mioTrack = trackID; % Index of the most important track
    
```

我们可以从MATLAB生成可通过认证的代码吗？

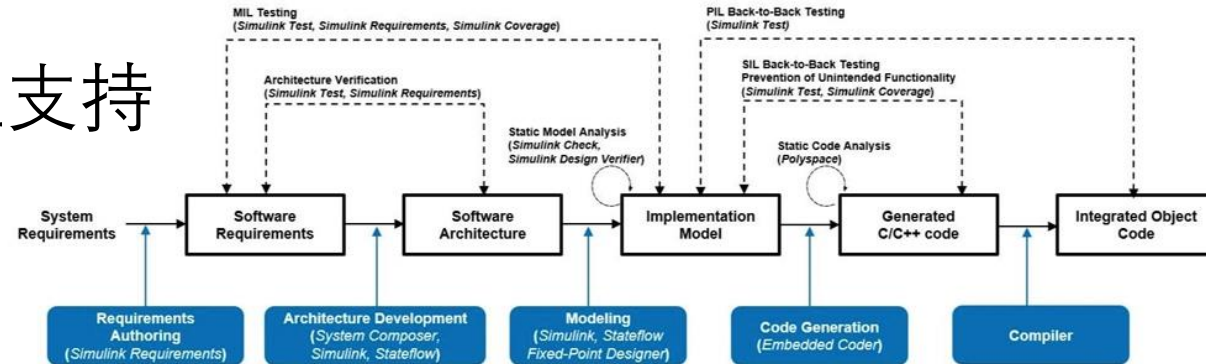
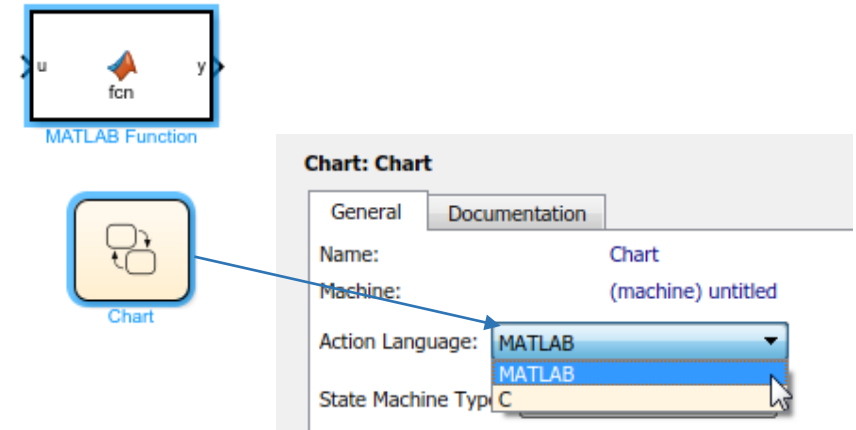


Yes! MATLAB和Simulink集成

- MATLAB语言可由MATLAB Function模块和/或Stateflow调用

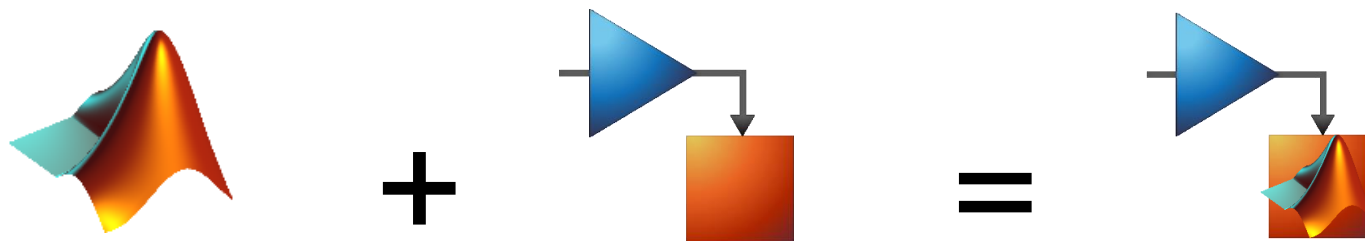
- MATLAB运算符
- MATLAB函数

- 我们的IEC认证套件和参考工作流程支持Simulink/MATLAB代码生成



最佳实践

- 我们可以将这些结合起来，并兼得两全其美
 - + MATLAB语言的丰富性
 - + Simulink验证工具系列的严格性



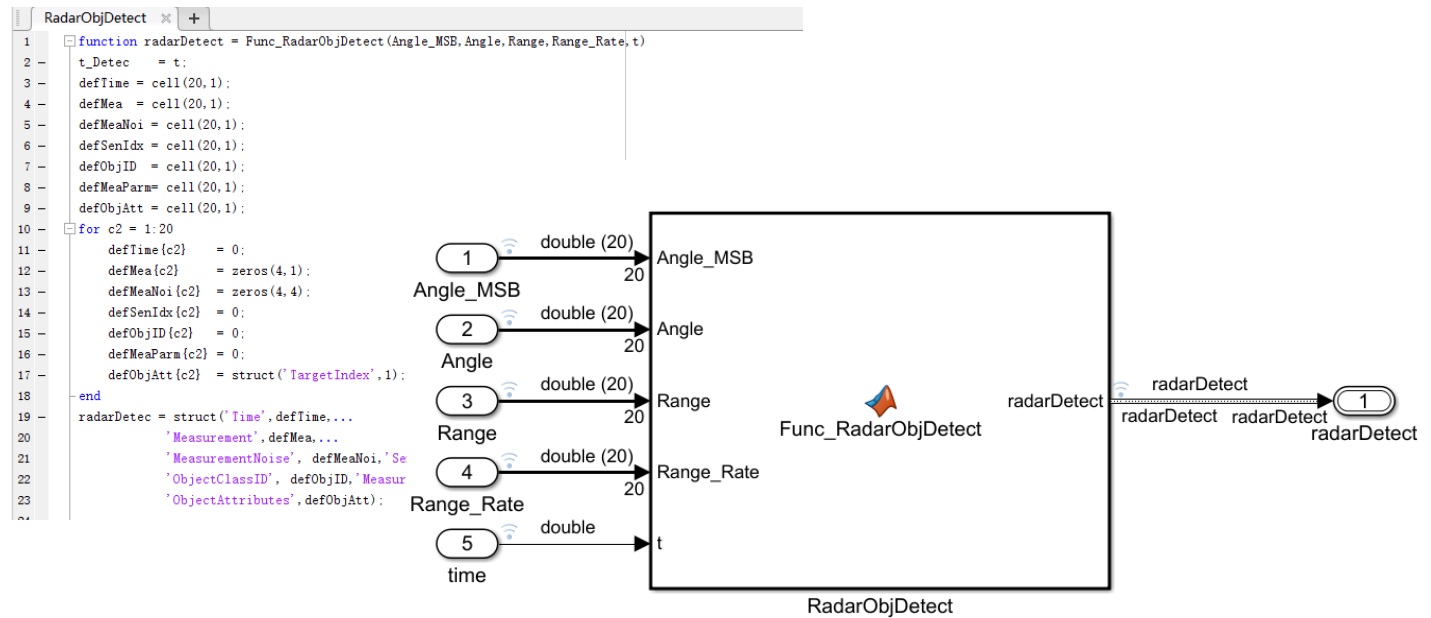
“我是MATLAB用户， Simulink对我有用吗?”

→ 如果您需要提供符合性证明

→ 定义MATLAB算法的数据流、控制流架构

MATLAB + Simulink ISO 26262 工作流程

- 我们的ISO验证流程现在支持这种组合语言
 - 需求可追溯性
 - 设计规范的符合性
 - 证明正确的功能
 - 证明没有意外功能
 - 软件的设计说明



需求可追溯性

- + Simulink Requirements 支持撰写需求，导入/导出需求、以及将需求链接到模型元素、代码及测试用例：
 - + 您可以将需求链接到MATLAB函数中的**代码行**，就像将需求链接到Simulink模块一样。
 - + 链接到MATLAB代码行中的**需求**可以生成为**注释**嵌入到自动代码中
 - + **MATLAB源代码**和**用户注释**也可以作为注释包括在自动代码内
- + 生成**需求追溯性报告**以获取证据

需求可追溯性样本

Requirements Editor

File Edit Display Analysis Report Help

View: Requirements Search

Index	ID	Summary
tracker		
1	#1	Track object path with extended Kalm...
1.1	#3	Compute Phi, Q, and R
1.2	#4	Propagate the covariance matrix
1.3	#5	Propagate the track estimate
1.4	#6	Compute results
1.4.1	#7	Observation estimate
1.4.2	#8	linearize measurement matrix
1.4.3	#9	Estimate error
1.5	#10	Compute Kalman gain
1.6	#11	Update estimate
1.7	#12	Update covariance matrix

Type: Functional

Index: 1.4.3

Custom ID: #9

Summary: Estimate error

Description Rationale

Keywords:

Revision information:

Links

Implemented by:

- residual = meas - yhat

Editor - Block: sldemo_radar_eml/MATLAB Function

EDITOR VIEW

New Open Save Find Files Compare Go To EDIT Breakpoints RUN SIMULINK

FILE NAVIGATE BREAKPOINTS

MATLAB Function

```
33  
34 % 4 a). Compute observation estimates:  
35 Rangehat = sqrt(xhat(1)^2+xhat(3)^2);  
36 Bearinghat = atan2(xhat(3),xhat(1));  
37  
38 % 4 b). Compute observation vector y and linearized measur  
39 yhat = [Rangehat;  
40         Bearinghat];  
41 M = [ cos(Bearinghat)          0 sin(Bearinghat)  
42       -sin(Bearinghat)/Rangehat 0 cos(Bearinghat)/Rangehat 0  
43  
44 % 4 c). Compute residual (Estimation Error)  
45 residual = meas - yhat;  
46  
47 % 5. Compute Kalman Gain:  
48 W = P*M'*inv(M*P*M' + R);
```

EXTKALMAN Ln 23 Col 28

Code Replacements Report

Coder Assumptions

Generated Code

[-] Main file

- ert_main.c

[-] Model files

- sldemo_radar_eml.c (3)
- sldemo_radar_eml.h

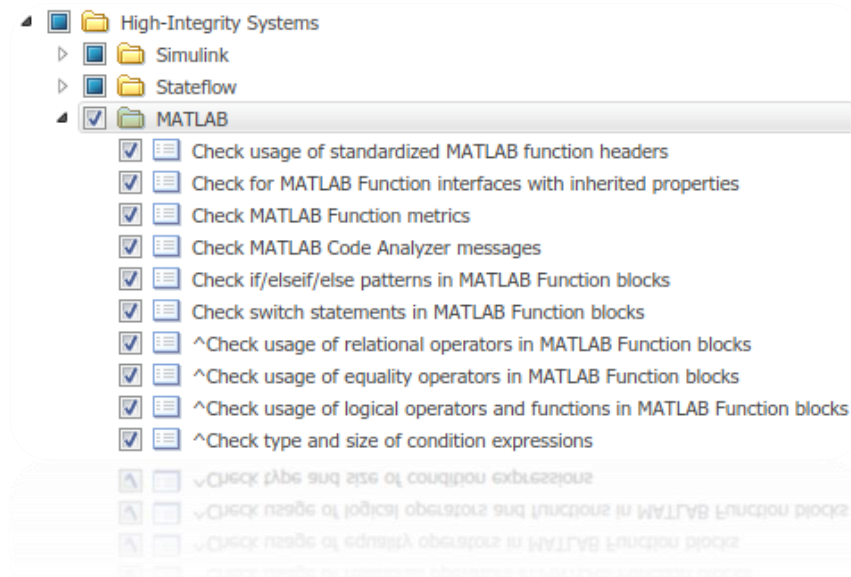
```
467 M[7] = 0.0;  
468  
469 /* 4 c). Compute residual (Estimation Error) */  
470 /* <S1>:1:45' residual = meas - yhat; */  
471 /* Requirements for MATLAB Function: '<S1>[737719.842.6' Line 45:  
472 * 1. Estimate error (tracker#9)  
473 */  
474 sldemo_radar_eml_B.residual[0] = sldemo_radar_eml_B.PolarCoords[0] -  
475     rtb_range;  
476 sldemo_radar_eml_B.residual[1] = sldemo_radar_eml_B.PolarCoords[1] -  
477     rtb_whiteNoise_idx_0;  
478  
479 /* 5. Compute Kalman Gain: */  
480 /* <S1>:1:48' W = P*M'*inv(M*P*M' + R); */
```


设计和代码规范

Simulink Check

- + Simulink Check可以检查是否具有良好的代码合规性
 - + 降低复杂度
 - + 强化注释密度
 - + MATLAB and Simulink之间的强数据类型
 - + 查找具有混合数据类型的逻辑运算符
- + 实施模型检查以搜索未链接到需求的MATLAB代码。
- + MISRA-C相关的MATLAB Coder和Embedded Coder设置
- + MATLAB 代码规范

MATLAB® PROGRAMMING
GUIDELINES
Version 1.0 (Japanese version)
Japan mbd Automotive Advisory Board
(Jmaab)
Mar 30, 2018



展示正确的功能

Simulink Requirements

Simulink Design Verifier

Simulink Test

- + 基于需求撰写测试用例, 并将测试用例与链接到需求。
- + Simulink Design Verifier 设计错误检测
- + Simulink Design Verifier 形式化验证
- + 通过Simulink Test执行测试用例
- + 从模型到代码进行软件在环 (SIL), 处理器在环 (PIL)测试。

表明没有任何意外功能

Simulink Coverage

Simulink Design Verifier

- + Simulink Coverage显示测试用例的完整性
 - + 模型覆盖度
 - + SIL/PIL代码覆盖度
- + Simulink Design Verifier可用于补足测试用例，确保背靠背测试的完整性。

The screenshot displays the Simulink Coverage tool interface. On the left, a block diagram shows a 'ZOH' block connected to an 'EXTKALMAN' MATLAB Function block. The 'EXTKALMAN' block has two outputs: 'residual' (output 2) and 'xhatOut' (output 4). The 'residual' output is connected to a 'Residuals' block, and the 'xhatOut' output is connected to an 'Est. Position [x, xdot, y, ydot]' block. On the right, the 'Coverage Details' window is open, showing a table of metrics and their coverage values:

Metric	Coverage
Cyclomatic Complexity	2
Decision	100% (3/3) decision outcomes

Below the table, the MATLAB code for the 'EXTKALMAN' function is displayed. Two lines of code are circled in red: line 1, 'function [residual, xhatOut] = EXTKALMAN(meas, deltat);', and line 16, 'if isempty(P)'. The code includes comments and initialization logic for the Kalman filter.

软件设计说明报告

- 在提供ISO 26262符合性证据中需要提供软件的设计说明报告。

8.5.1 Software unit design specification resulting from requirements [8.4.2](#) to [8.4.5](#).

NOTE In the case of model-based development, the implementation model and supporting descriptive documentation, using methods listed in [Tables 5](#) and [6](#), specifies the software units.

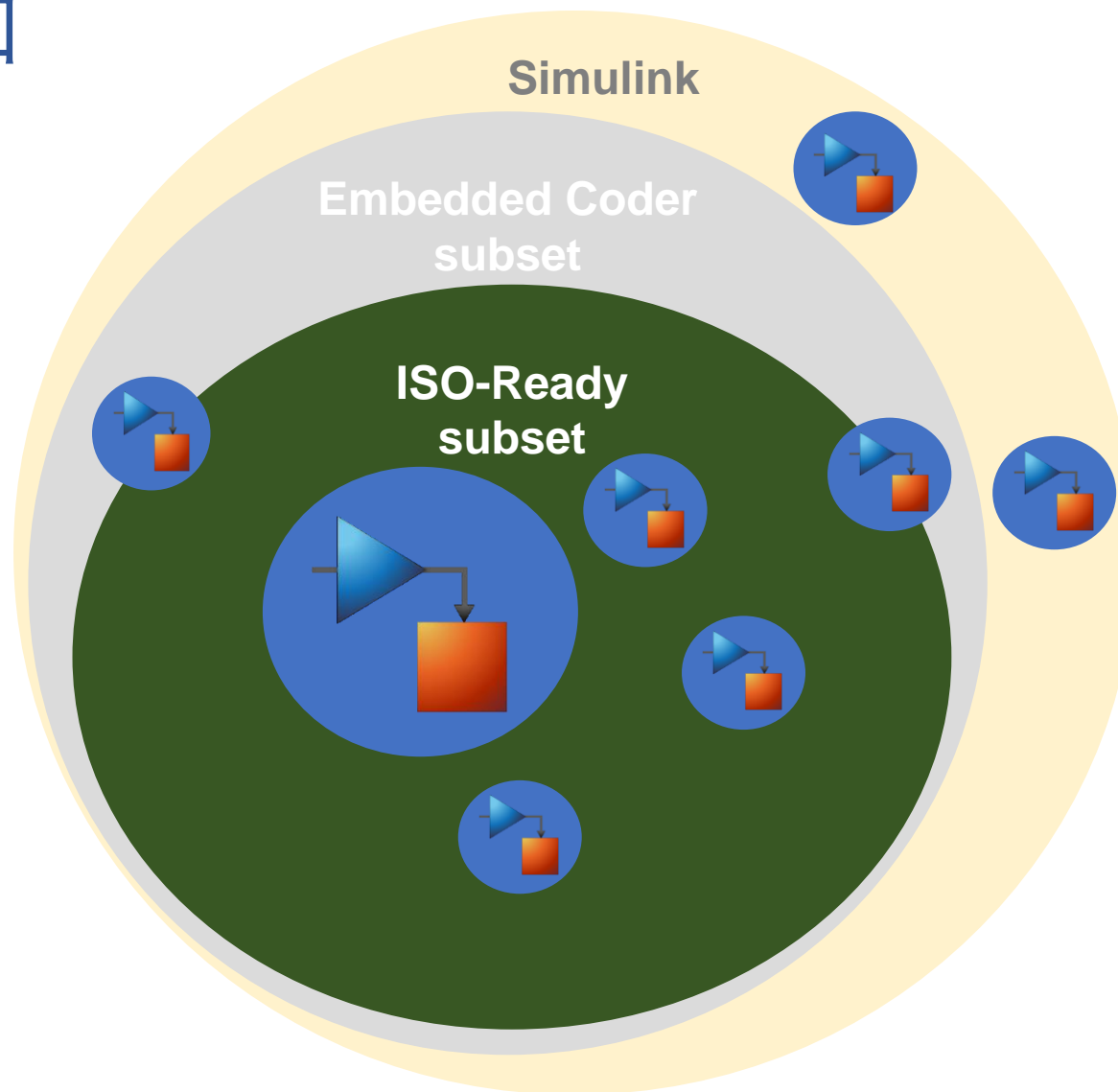
- MATLAB/Simulink Report Generator™ 包含预定义的系统设计报告模板。
 - 建议自定义模板以包含外部MATLAB代码。

到目前为止的总结

- 客户今天已成功在符合ISO 26262的产品中使用MATLAB
- 我们的验证工作流程和工具支持Simulink调用的MATLAB
- 但是...依然存在挑战
 - 符合代码标准：从MATLAB生成的代码的MISRA-C兼容性的潜在问题
 - MATLAB vs C代码覆盖度

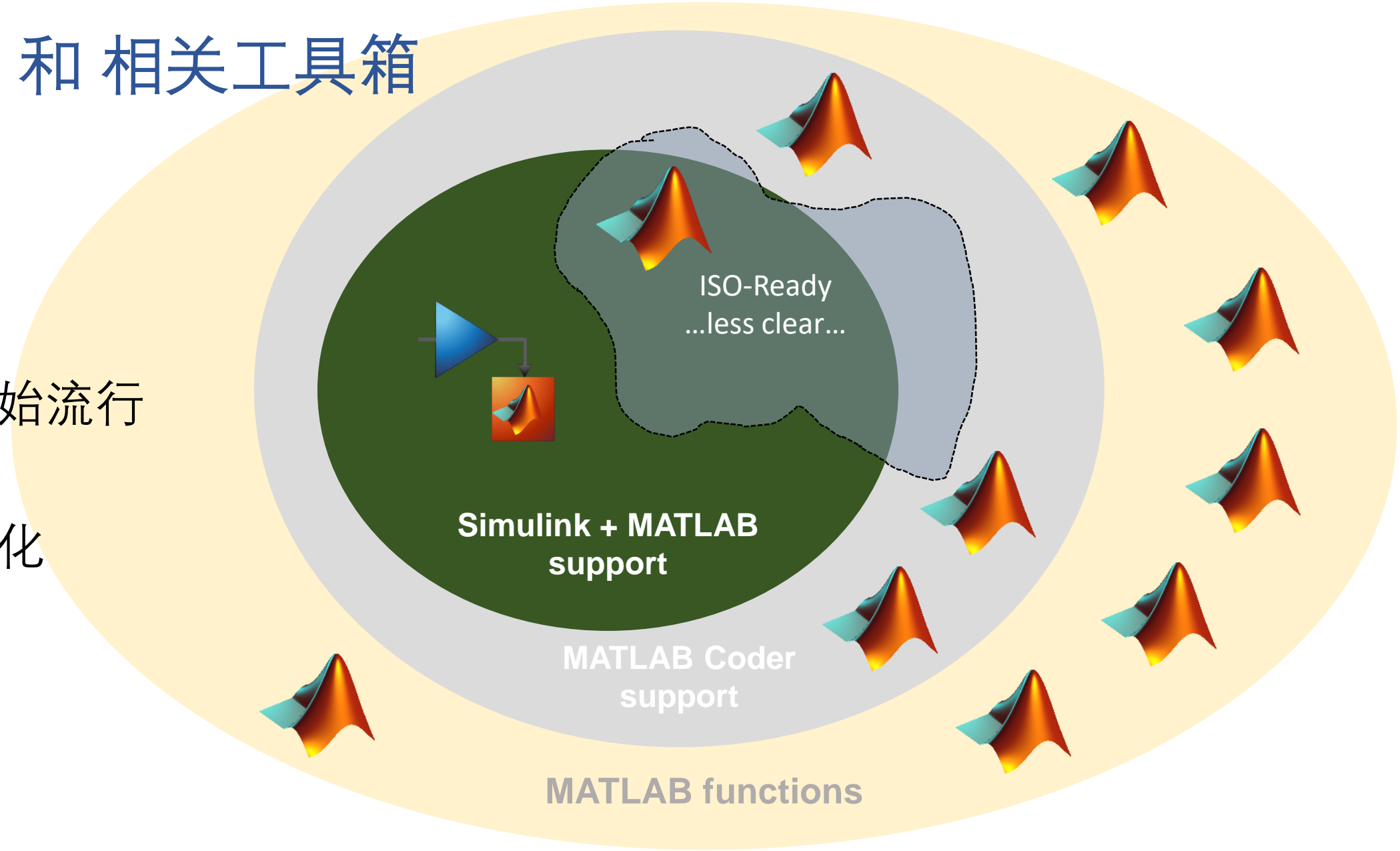
Simulink核心模块和工具箱

- 广泛接受
- 丰富的工具支持
- 经过充分验证



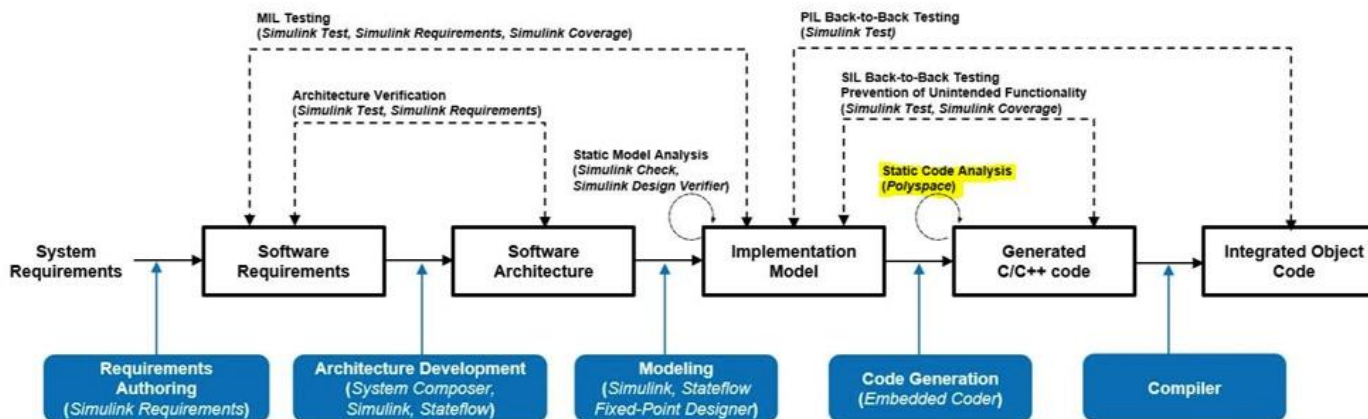
MATLAB 和相关工具箱

- 逐渐开始流行
- 不断优化



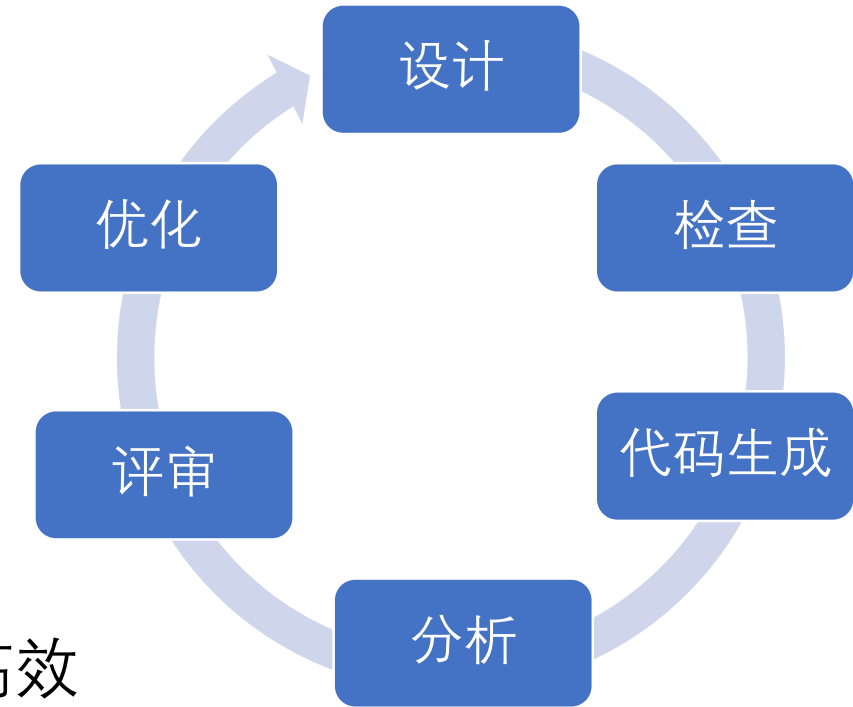
应对挑战：符合代码标准

- 要评估从MATLAB生成的代码对MISRA®C的兼容性，请使用静态代码分析器工具（例如Polyspace Bug Finder™）来检测不符合MISRA C的代码，并调查结果。如果不兼容的代码来自内置的MATLAB函数，则有两个处理方法：
 - 为MISRA C的警告写理由
 - 用重写的函数替换内置的MATLAB函数，并对非推荐代码/函数实施Model Advisor检查。



应对挑战：符合代码标准

- **最佳实践是**
 - 运行模型检查 **Simulink Check**
 - 生成代码
 - 分析合规性 **Polyspace Bug Finder**
- 如果发现问题
 - 记录并继续
 - 修改算法或使用一个兼容功能
- 定制一个功能子集(语言子集)
- 随着时间的流逝，这个过程将变得越来越高效



应对挑战：代码覆盖度

- MATLAB函数用C / C ++实现可能很复杂

```
5. Compute Kalman Gain:  
W = P*M'*inv(M*P*M'+ R);
```

- 一条测试用例就能覆盖的MATLAB代码，实际上还需要更多的测试用例才能在生成的C中不显示任何意外功能

- 策略包括

- 开发功能/特性的单元测试
- 使用一个更简单的C代码实现
- 通过链接到测试用例的coverage justification filter跟踪缺失覆盖度的外部MATLAB代码

```
480 /* 5. Compute Kalman Gain: */  
481 /* '<S1>:1:48' W = P*M'*inv(M*P*M'+ R); */  
482 for (i = 0; i < 2; i++) {  
483     for (iU = 0; iU < 4; iU++) {  
484         Phi_tmp_tmp = (int32_T)((int32_T)(iU << 1) + i);  
485         x_tmp[(int32_T)(iU + (int32_T)(i << 2))] = M[Phi_tmp_tmp];  
486         M_0[Phi_tmp_tmp] = 0.0;  
487         Phi_tmp = (int32_T)(iU << 2);  
488         M_0[Phi_tmp_tmp] += sldemo_radar_eml_DWork.P[Phi_tmp] * M[i];  
489         M_0[Phi_tmp_tmp] += sldemo_radar_eml_DWork.P[(int32_T)(Phi_tmp + 1)] *  
490             0.0;  
491         M_0[Phi_tmp_tmp] += sldemo_radar_eml_DWork.P[(int32_T)(Phi_tmp + 2)] *  
492             M[(int32_T)(i + 4)];  
493         M_0[Phi_tmp_tmp] += sldemo_radar_eml_DWork.P[(int32_T)(Phi_tmp + 3)] *  
494             0.0;  
495     }  
496 }  
497  
498 for (i = 0; i < 2; i++) {  
499     for (iU = 0; iU < 2; iU++) {  
500         Phi_tmp_tmp = (int32_T)(i << 2);  
501         Phi_tmp = (int32_T)((int32_T)(i << 1) + iU);  
502         Phi_1[Phi_tmp] = ((x_tmp[(int32_T)(Phi_tmp_tmp + 1)] * M_0[(int32_T)(iU  
503             + 2)] + x_tmp[Phi_tmp_tmp] * M_0[iU] + x_tmp[(int32_T)(Phi_tmp_tmp +  
504             2)] * M_0[(int32_T)(iU + 4)]) + x_tmp[(int32_T)(Phi_tmp_tmp + 3)] *  
505             M_0[(int32_T)(iU + 6)] + R[Phi_tmp];  
506     }  
507 }  
508  
509 if (fabs(Phi_1[1]) > fabs(Phi_1[0])) {  
510     rtb_range = Phi_1[0] / Phi_1[1];  
511     rtb_WhiteNoise_idx_0 = 1.0 / (rtb_range * Phi_1[3] - Phi_1[2]);  
512     M_tmp = Phi_1[3] / Phi_1[1] * rtb_WhiteNoise_idx_0;  
513     M_tmp_0 = -rtb_WhiteNoise_idx_0;  
514     y_idx_2 = -Phi_1[2] / Phi_1[1] * rtb_WhiteNoise_idx_0;  
515     rtb_WhiteNoise_idx_0 *= rtb_range;  
516 } else {  
517     rtb_range = Phi_1[1] / Phi_1[0];  
518     rtb_WhiteNoise_idx_0 = 1.0 / (Phi_1[3] - rtb_range * Phi_1[2]);  
519     M_tmp = Phi_1[3] / Phi_1[0] * rtb_WhiteNoise_idx_0;  
520     M_tmp_0 = -rtb_range * rtb_WhiteNoise_idx_0;  
521     y_idx_2 = -Phi_1[2] / Phi_1[0] * rtb_WhiteNoise_idx_0;  
522 }  
523 }
```

总结

- 客户已成功在兼容ISO 26262的产品中使用Simulink 与 MATLAB
- 完善整个流程
 - 解决从MATLAB生成的代码的MISRA-C兼容性的潜在问题
 - 实现MATLAB到C代码的代码覆盖度
 - 定制化单元设计报告或其他定制化工具开发
- 延伸资料: [Best practices for Simulink and MATLAB for ISO-26262](#)
- 联系方式: <alex Xu@mathworks.com>

