东风商用车
DONGFENG TRUCKS

# MATLAB助力东风发动机自主电控平台研发

*周杰敏, 东风商用车技术中心*

MATLAB EXPO

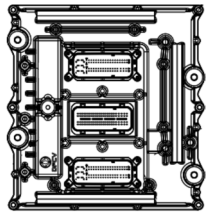
# 背景

## 开发历程

随着中国汽车产业的崛起，自主研发迫在眉睫，对于发动机而言，自主电控系统的研发是实现核心竞争力突破的关键。

东风商用车自2012年致力于发动机自主电控系统(EECU)的研发，于2017年实现工程化，2018年满足国五排放标准的自主电控系统实现小批量上市，2021年满足国六排放标准的自主电控系统实现量产。

2012 → 2017 → 2018 → 2021 → 未来



自主EECU工程化开发

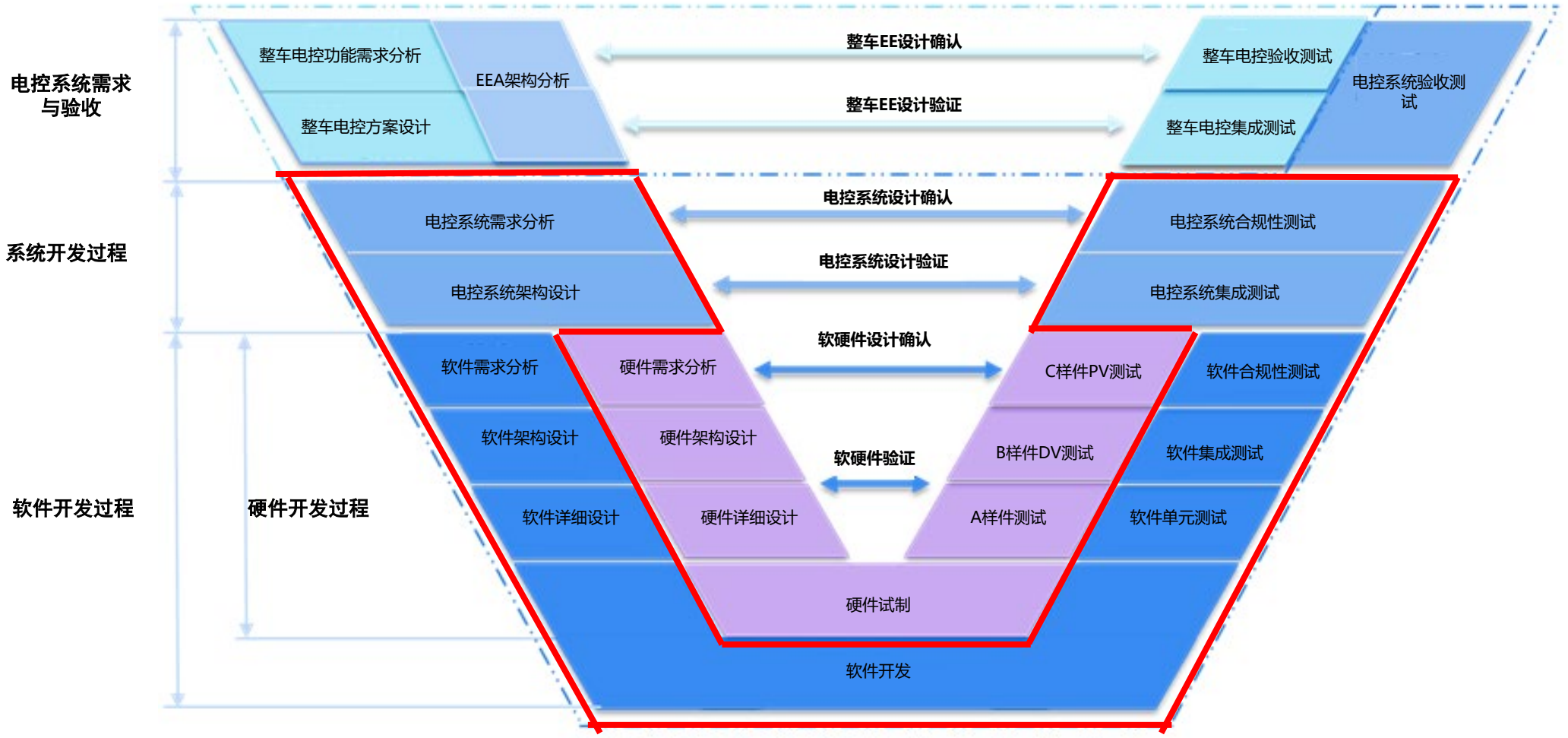
国五小批量

国六量产

## 面临的挑战

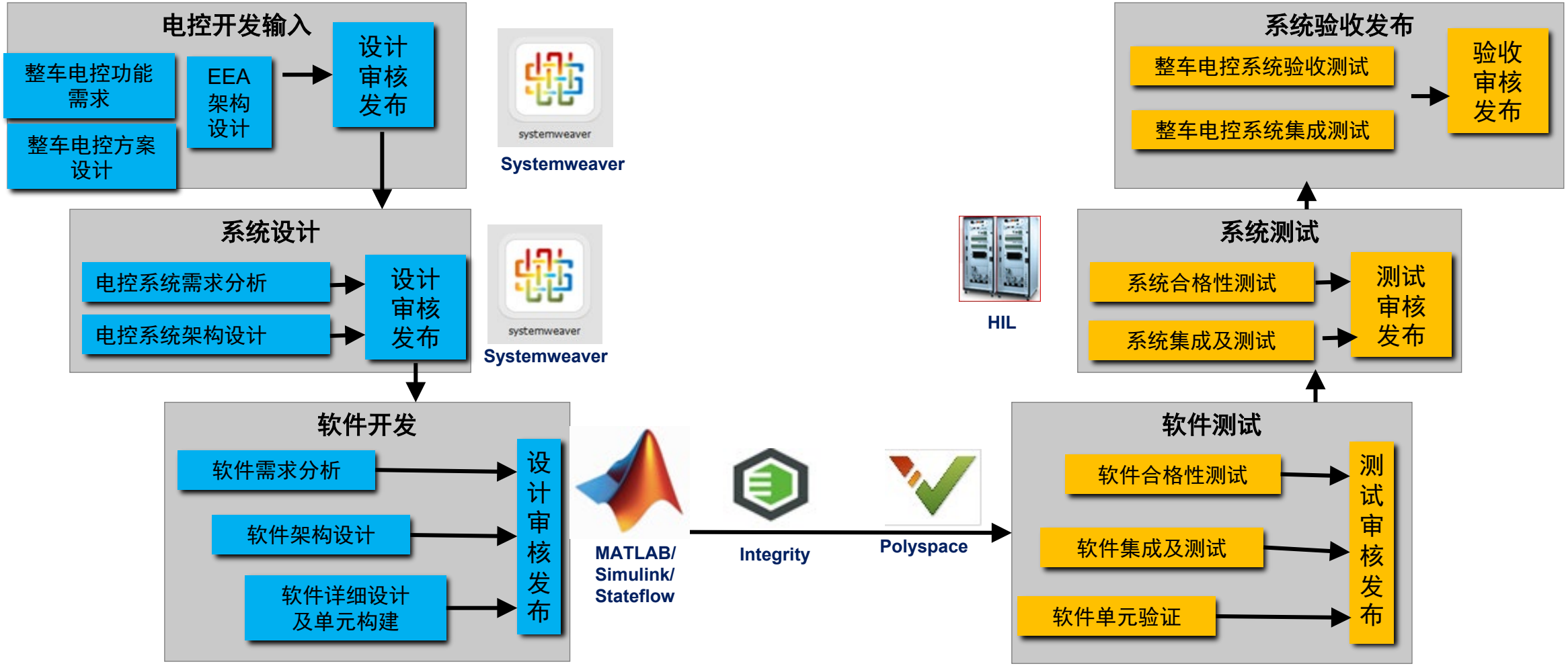
- 应对产品和市场需求，现有平台的软件快速更新迭代；
- 日益复杂的功能模块关系；
- 高度的可靠性和安全性要求。

# 软件开发流程




软件开发V流程，参考ASPICE流程进行管控

# 软件开发流程

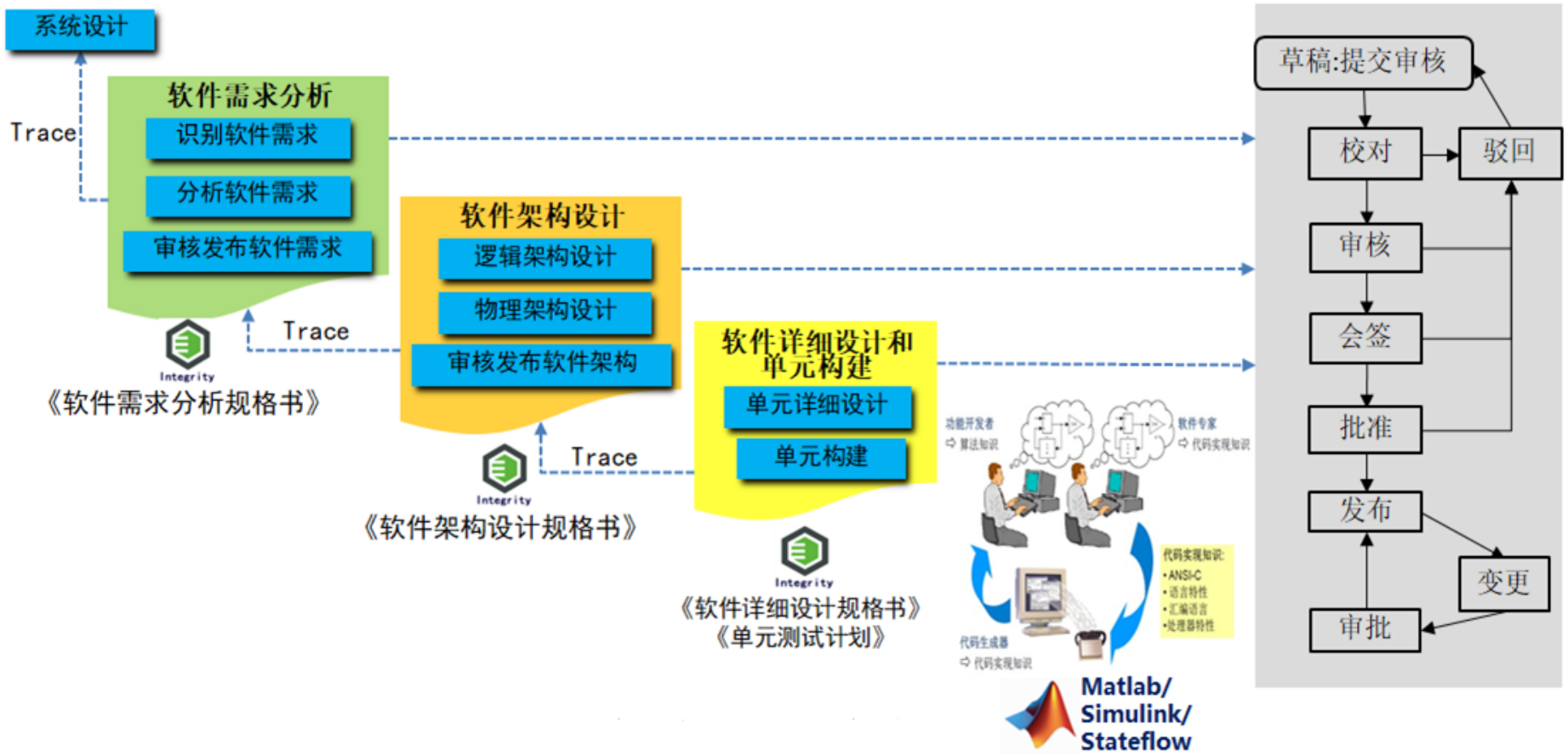



建立自动流水线式的工具链平台,保证流程的执行，提升开发效率

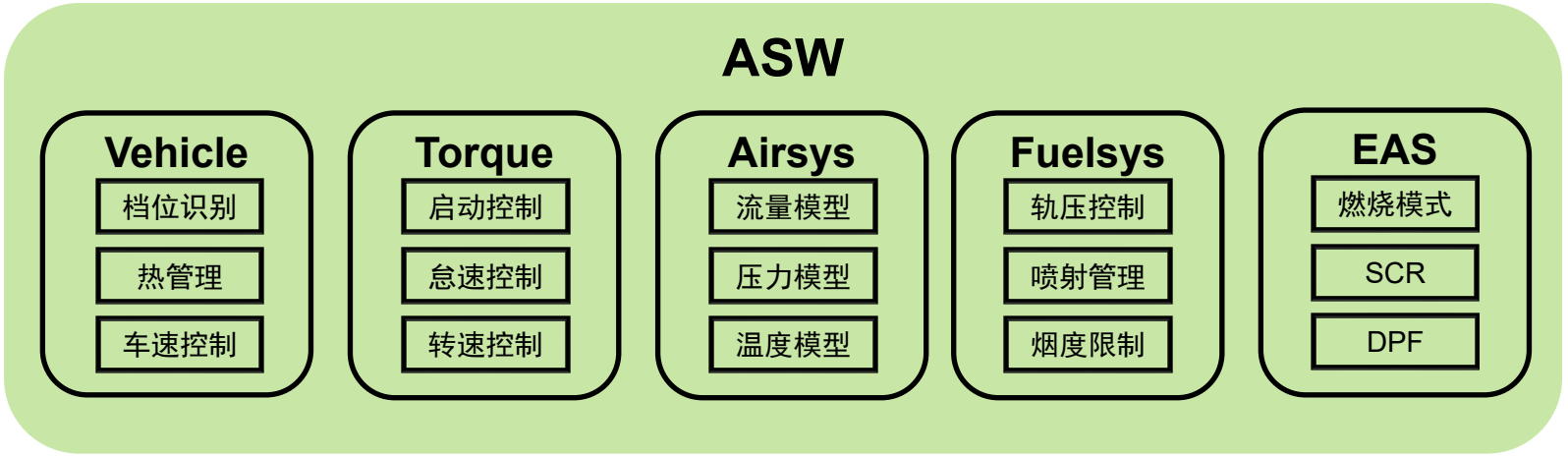
# 软件开发流程-1软件需求开发

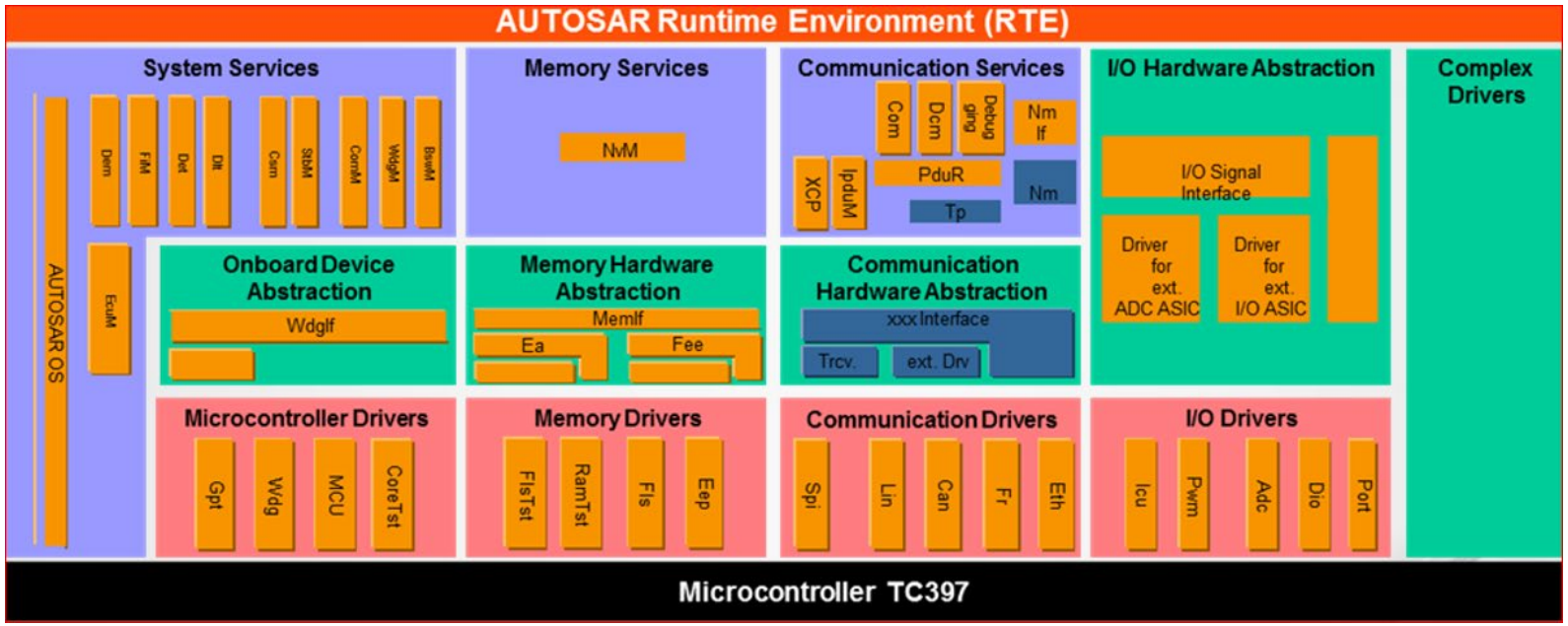
## 软件需求设计和管理

- 参考ASPICE流程进行需求设计，审批，变更，追溯及版本管理。
- V流程各环节的追溯，包括需求与架构，需求与测试报告等等。



# 软件开发流程-2软件架构设计




## 架构设计

- 基于Autosar架构，实现软件模块化，接口标准化；
- ASW按照功能进行分类，可迅速移植到任意控制系统中；
- ASW功能模块配置不同调度周期，以及不同CPU核内运行，尽可能提高运行效率。

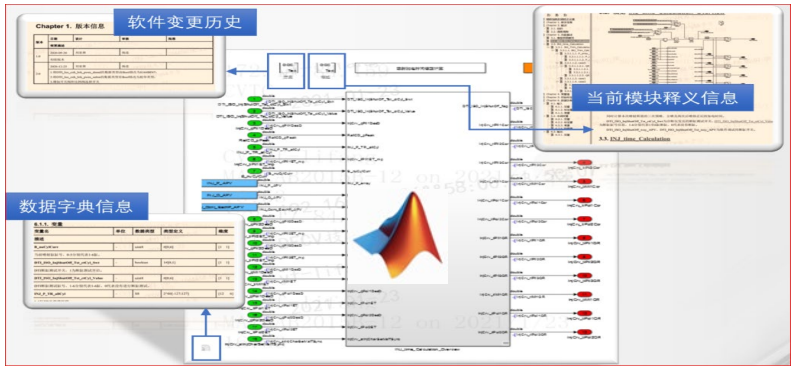
# 软件开发流程-3软件详细设计及实现

## 模型构建和代码生成

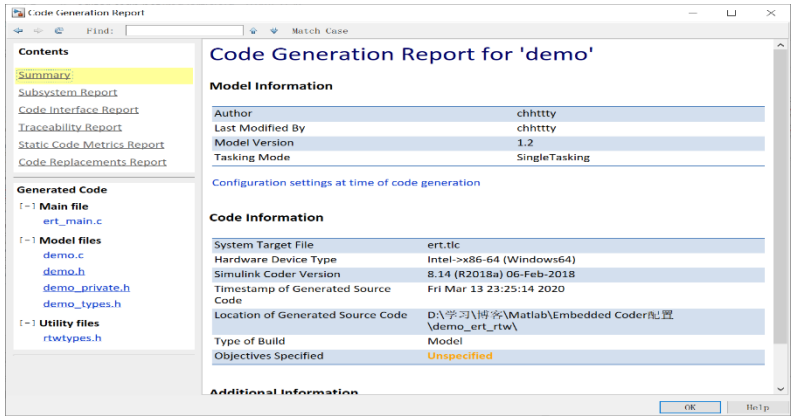
- 模型搭建、模型测试(MIL)、代码生成、单元说明文档生成均为自主工具链调用MATLAB相关工具箱实现
- 每个环节按照退出标准结束，保证软件可靠

### 自主工具链


### 自动生成说明文档



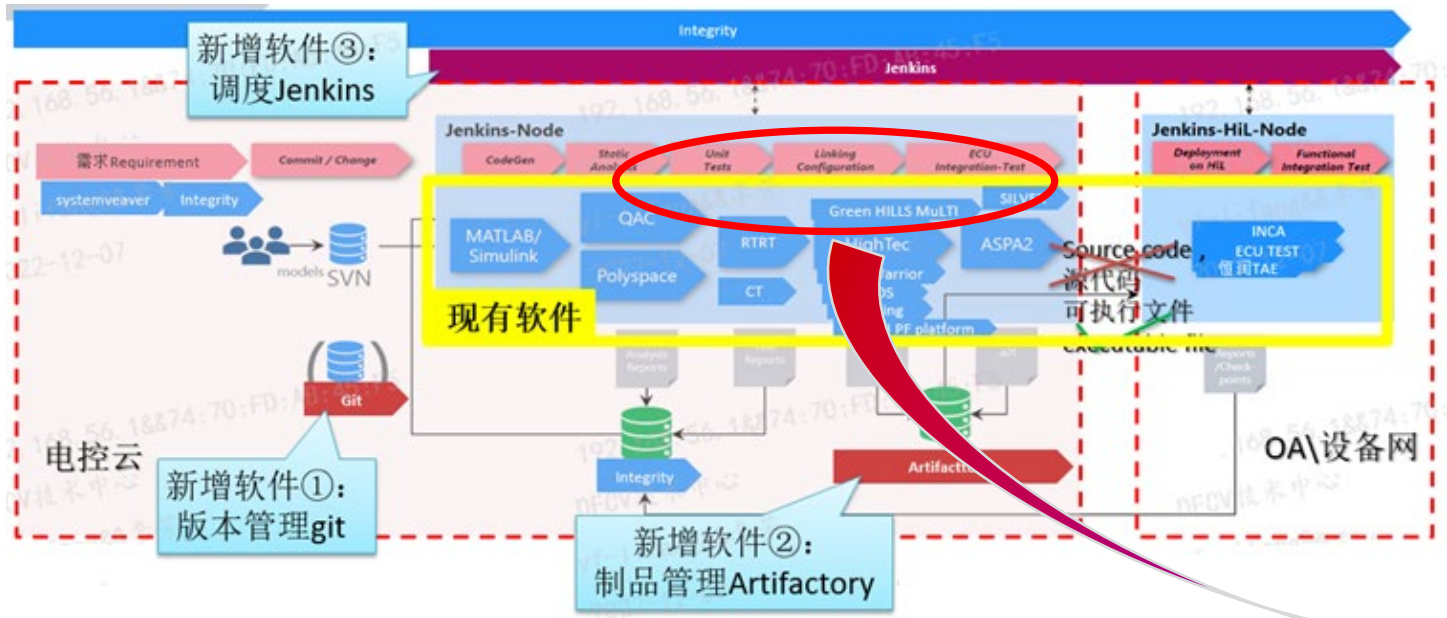

### Embedded Coder自动生成代码



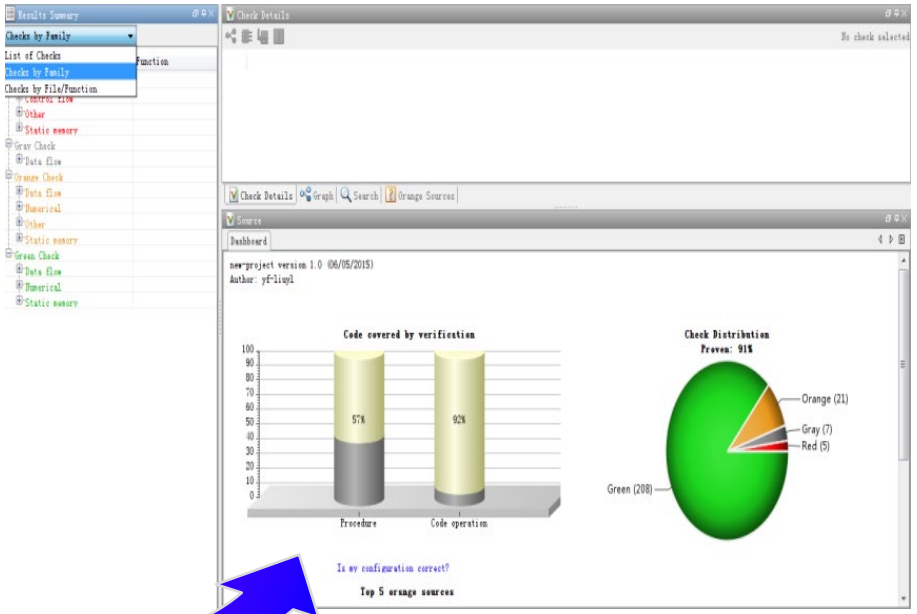
# 软件开发流程-4软件验证

## 代码验证

- 基于ASPICE的标准化验证流程
- 代码静态分析为自主工具链调用Polyspace实现，实现编码规范，运行时错误等检查，保障代码可靠性
- 验证报告追溯需求，保证软件可靠性




自主工具链




Polyspace代码静态分析

# 软件开发流程-4软件验证

## Polyspace的引入

- 在高原试验中某模块数据溢出，导致发动机转速异常波动
- 多人协作开发时，某个变量被多处写入，程序工作异常

## Polyspace用于代码验证的优势

- 检查代码是否满足编码规范（Misra-C），生成报告
- 检查代码中是否存在运行时错误，无需编写测试用例、执行代码
- 保证代码中没有严重的错误，减少动态测试工作量

### Chapter 1. Polyspace Code Verification Summary

**Table 1.1. Code Metrics Summary**

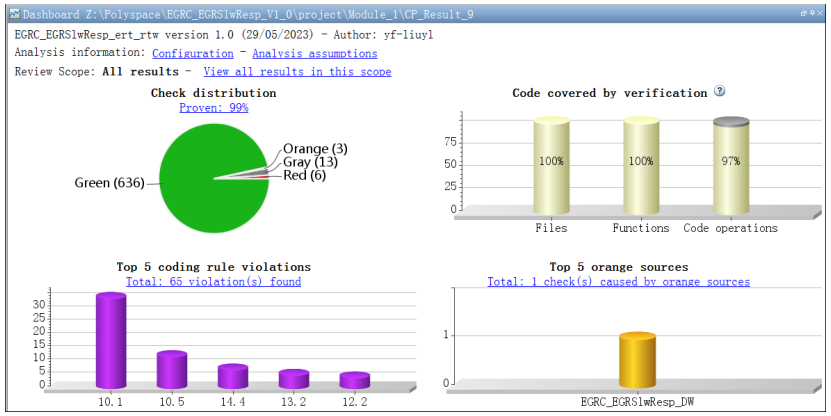
| Polyspace Code Metrics | Disabled |
|---|---|
| Pass/Fail | |

**Table 1.2. Coding Standard Summary - MISRA C:2012 Guidelines**

| MISRA C:2012 Guidelines | Enabled |
|---|---|
| Violations | 65 |
| Pass/Fail | |

**Table 1.3. Run-Time Checks Summary**

| Run-Time Checks | Enabled |
|---|---|
| Number of Red Checks | 6 |
| Number of Gray Checks | 13 |
| Number of Orange Checks | 3 |
| Number of Green Checks | 636 |
| Proven | 99.5% |
| Pass/Fail | |




验证报告总览

### Unproven Run-Time Errors

**Table 4.3. Z:\Polyspace\EGRC_EGRSlwResp_V1_0\sources\EGRC_EGRSlwResp.c**

| ID | Check | Function | Line | Col | Detail | Jus | Severity | Status | Comment |
|---|---|---|---|---|---|---|---|---|---|
| 1394 | Overflow | EGRC_EGRSlwResp_step() | 1379 | 37 | Warning: operation [+] on scalar may overflow (on MIN or MAX bounds of INT32) | Yes | Unset | Not a defect | 经过分析，加法操作数数值较小，累加不会溢出INT32 |
| 1453 | Overflow | EGRC_EGRSlwResp_step() | 1633 | 37 | Warning: operation [+] on scalar may overflow (on MIN or MAX bounds of INT32) | Yes | Unset | Not a defect | 经过分析，加法操作数数值较小，累加不会溢出INT32 |
| 1516 | Overflow | EGRC_EGRSlwResp_step() | 1924 | 14 | Warning: operation [+] on scalar may overflow (result strictly greater than MAX UINT32) | Yes | Unset | Not a defect | 当时间累积溢出时，将被限制在MAX_uint32，不影响功能。 |

### Chapter 3. MISRA C:2012 Guidelines

#### MISRA C:2012 Guidelines Summary - Violations by File

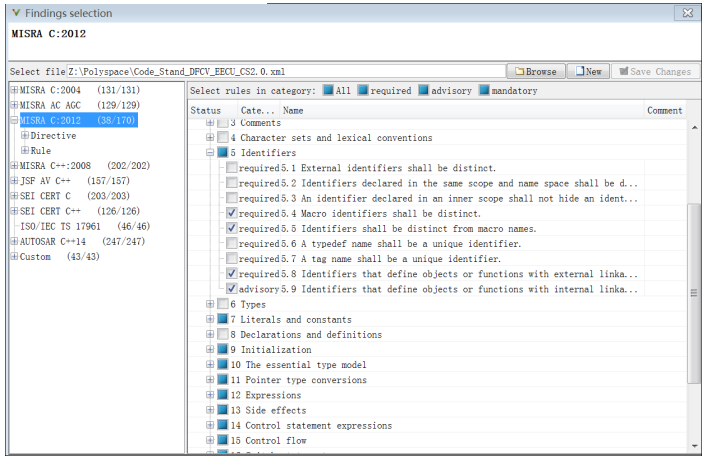
| File | Total |
|---|---|
| Z:\Polyspace\EGRC_EGRSlwResp_V1_0\sources\EGRC_EGRSlwResp.c | 65 |
| **Total** | **65** |

#### MISRA C:2012 Guidelines Violations

**Table 3.1. Z:\Polyspace\EGRC_EGRSlwResp_V1_0\sources\EGRC_EGRSlwResp.c**

| ID | Guideline | Message | Function | Line | Col | Jus | Severity | Status | Comment |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 10.5 | The value of an expression should not be cast to an inappropriate essential type. | EGRC_EGRSlwResp_RisingEdge() | 228 | 33 | No | Unset | Unreviewed | |
| 64 | 14.4 | The controlling expression of an if statement and the controlling expression of an iteration-statement shall have essentially Boolean type. | EGRC_EGRSlwResp_step() | 253 | 2 | No | Unset | Unreviewed | |
| 3 | 14.4 | The controlling expression of an if statement and the controlling expression of an iteration-statement shall have essentially Boolean type. | EGRC_EGRSlwResp_step() | 291 | 4 | No | Unset | Unreviewed | |
| 30 | 14.4 | The controlling expression of an if statement and the controlling expression of an iteration-statement shall have essentially Boolean type. | EGRC_EGRSlwResp_step() | 326 | 4 | No | Unset | Unreviewed | |
| 9 | 12.2 | The right hand operand of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand. | EGRC_EGRSlwResp_step() | 327 | 39 | No | Unset | Unreviewed | |
| 34 | 10.1 | Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category signed. | EGRC_EGRSlwResp_step() | 327 | 39 | No | Unset | Unreviewed | |

精确定位缺陷位置

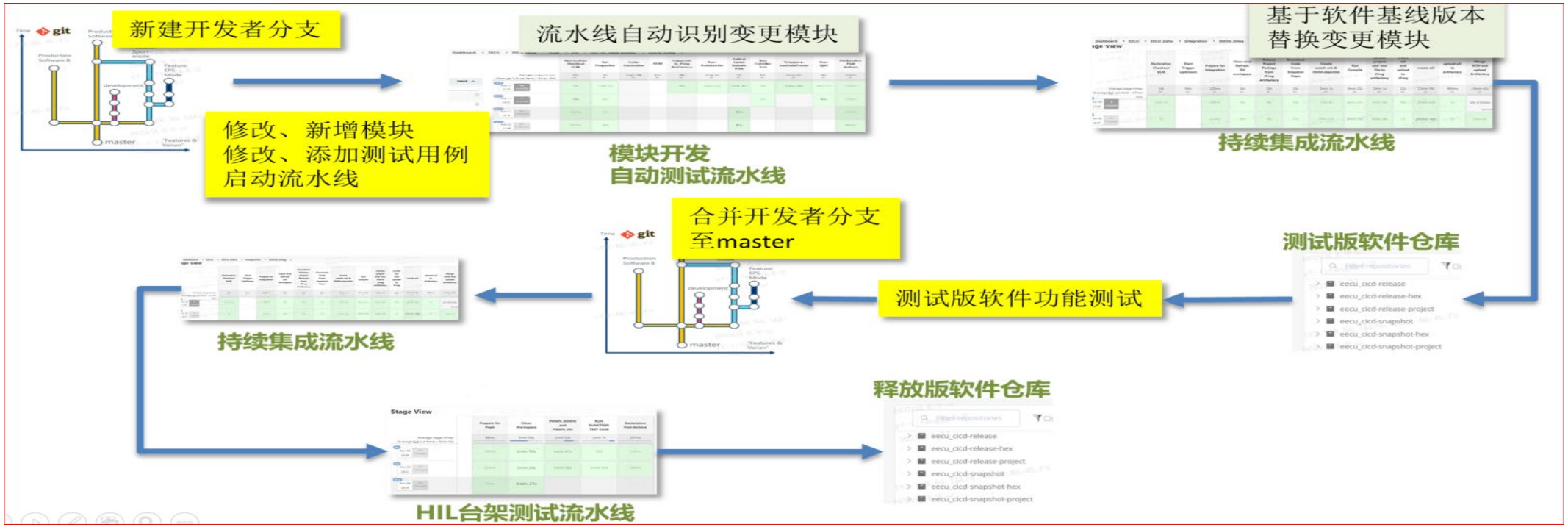



灵活配置编码规范

# 软件开发流程-5软件持续集成

源代码版本管理


持续集成，平台移植



# 自动化工具链

工具链当前状态：

- 集成当前所有电控软件开发和测试工具
- 实现自动流水线式的工具链平台，自动完成如模型检查、代码生成、代码静态和动态测试、软件集成、SIL/HIL测试等各个环节，输出中间各环节报告
- 实现手写代码、模型（用于代码生成）、测试用例、测试报告、最终制品的版本和权限管理。

持续集成自动化工具链的优点：保证软件高质量的快速迭代

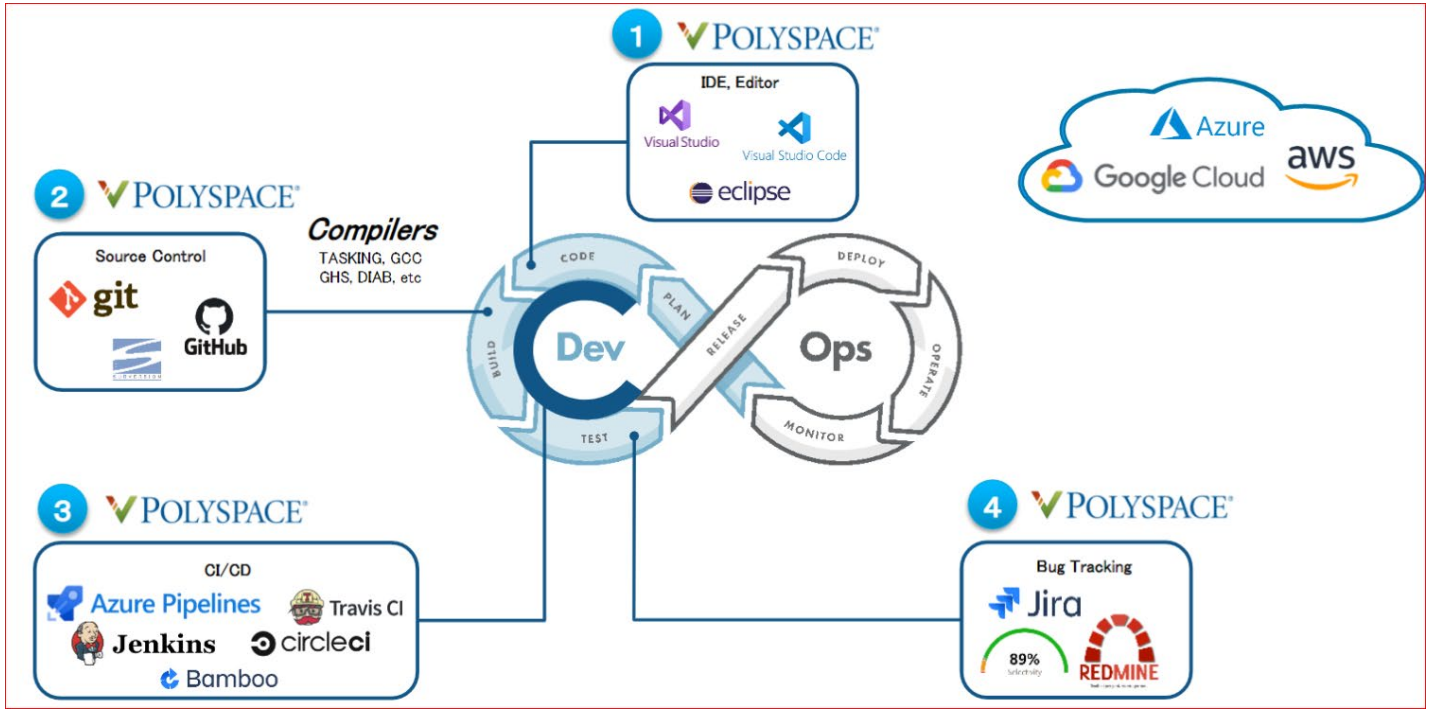
- 降低了开发风险
- 自动化的集成和测试，减少了开发时间，提高了开发效率

优化方向：形成完整的自动化工具平台

- 集成需求管理工具
- 集成Autosar的配置工具

# MATLAB助力自主电控系统开发

- MATLAB/Simulink丰富的工具箱和外部软件接口，加速控制策略的开发和早期验证
- Embedded Coder自动生成代码，标准化程度高，执行效率高，可维护性高，可移植性高；
- Polyspace代码静态分析工具，从编码规范到运行时错误，保障单元代码高质量交付；
- MATLAB的持续集成理念与DFCV的自主工具链的发展高度契合，助力自主电控事业腾飞。



# MATLAB EXPO

## Thank you