从功能安全视角看软件架构设计

来源：薄说安全 2022-05-07 353

功能安全应该如何考虑软件架构，什么样的架构是符合功能安全标准要求的，对于软件架构工程师和功能安全工程师，很难在两个方面都说得明白，本篇来从功能安全的角度谈谈软件架构设计的基本要求。

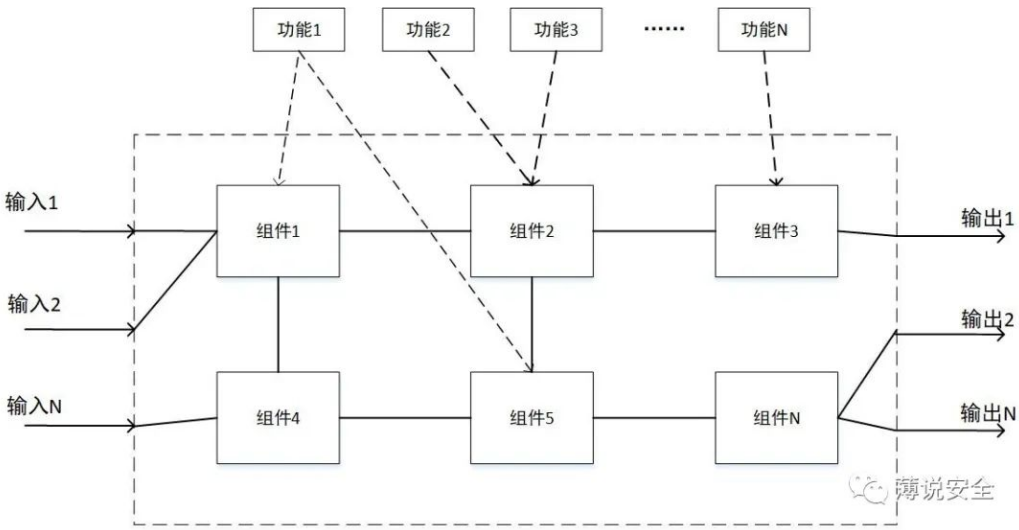
首先，功能安全软件的架构设计是**基于两个层次**的：

第一：选取和建立一个层次分明，易于理解的软件架构；
第二：在第一条的基础上，符合相应功能安全等级要求的软件设计要求。

接下来，以汽车功能安全标准ISO26262-6和轨道交通软件功能安全标准EN50128作为基准，谈谈标准是如何从以上两个层次来做出规定的。

01 软件架构阶段的开始

软件架构设计是软件生命周期的第二个阶段，前面的阶段是软件需求阶段（software requirements specification）,在软件需求设计时，把整个软件当成一个黑盒处理，来确定该软件的所有功能、性能，与硬件的接口定义，与外部其它系统的接口定义，而在软件架构阶段，需要设计一种架构来满足软件需求，通过层次化结构的方式来表示软件架构的组件构成和他们之间的交互方式。以下图为例，虚线框之外是软件需求，虚线框内是软件架构。

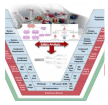


同类文章推荐


ISO 26262 是否足以应对自动驾驶系统...
2021-04-27


万字长文：自动驾驶汽车道路安全探究
2021-02-18


AUTOSAR软件架构 -- 软件分层概述
2021-01-11


干货|功能安全、预期功能安全与信息安...
2020-11-17


浅谈城际铁路信号系统选型
2020-08-26

02 什么是软件组件

上面这个图用于解释软件架构所做的工作，将整个软件划分为功能和接口清晰的组件。在ISO26262-6和EN50128中都有软件组件（component）这个概念，先来看看这个组件的定义：

component
a constituent part of software which has well-defined interfaces and behaviour with respect to the software architecture and design and fulfils the following criteria:

- it is designed according to “Components” (see Table A.20);
- it covers a specific subset of software requirements;
- it is clearly identified and has an independent version inside the configuration management system and is a part of a collection of components (e. g. subsystems) which have an independent version

EN50128对组件的定义

component
non-system level *element* (3.41) that is logically or technically separable and is comprised of more than one *hardware part* (3.71) or one or more *software units* (3.159)

EXAMPLE A microcontroller.

Note 1 to entry: A component is a part of a *system* (3.163).

ISO26262对组件的定义

很多人把组件理解成一个函数、或一个包括多个函数的文件，从标准中对组件的定义来看，组件作为一组软件功能需求的集合，有点类似于面向对象语言中的类的概念，是在软件架构中的一个个独立的个体，可以单独替换更新的基本元素。通过软件组件的应用可以达到重复使用和替换的目的，它可以被单独测试和版本管理。

03 软件架构设计原则

如何设计软件架构中的组件，在ISO26262-6中提出了以下设计原则：

Principles		ASIL			
		A	B	C	D
1a	Appropriate hierarchical structure of the software components	++	++	++	++
1b	Restricted size and complexity of software components[a]	++	++	++	++
1c	Restricted size of interfaces[a]	+	+	+	++
1d	Strong cohesion within each software component[b]	+	++	++	++
1e	Loose coupling between software components[b,c]	+	++	++	++
1f	Appropriate scheduling properties	++	++	++	++
1g	Restricted use of interrupts[a,d]	+	+	+	++
1h	Appropriate spatial isolation of the software components	+			++
1i	Appropriate management of shared resources[e]	++	++	++	++

设计原则从两个方面来进行规定：

单个组件：限制组件的规模，限制接口的数量，有限的中断使用，目的在于降低每个组件的复杂度，
多个组件：组件内强内聚，组件之间松耦合，组件之间的空间隔离，组件之间共用资源的冲突管理。

避免出现以下情况：

系统的一个功能分散在不同的组件中，代码多个地方改变同一变量或状态；
未对系统的中断功能进行限制，多个中断造成导致软件的时间约束不受控；
组件不具备可维护性，不可能做到重构其中一个组件；
组件未做到良好的封装或封装不合理，对外的接口过于繁杂或内部状态不可知；
组件设计缺乏可读性，只有专家级人员才能看得懂；

04 软件架构内容要点

划分了层次化的组件后，软件架构重点描述组件之间的关系：静态关系和动态关系。静态设计方面如组件之间的接口、与硬件的关系、组件的分层结构通常比较明确，容易忽视的是动态设计，软件的动态行为需要考虑：

事件和行为的功能；
数据处理的逻辑顺序；
控制流和并发进程；
通过接口和全局变量传递的数据流；
时间约束。

这些内容仅用文字表达容易造成歧义，难以描述准确，因此推荐使用建模和文字表达相结合的方式，下表是EN50128对建模方法的推荐表，虽然标准中仅要求至少使用一种，但从软件架构需要表达的不同动态行为上，强烈建议根据不同的行为采用适合的建模方法。例如采用文字表达难以准确描述不同系统通信交互的时序关系，采用Sequence Diagrams（序列图）可以明确表示交互关系。

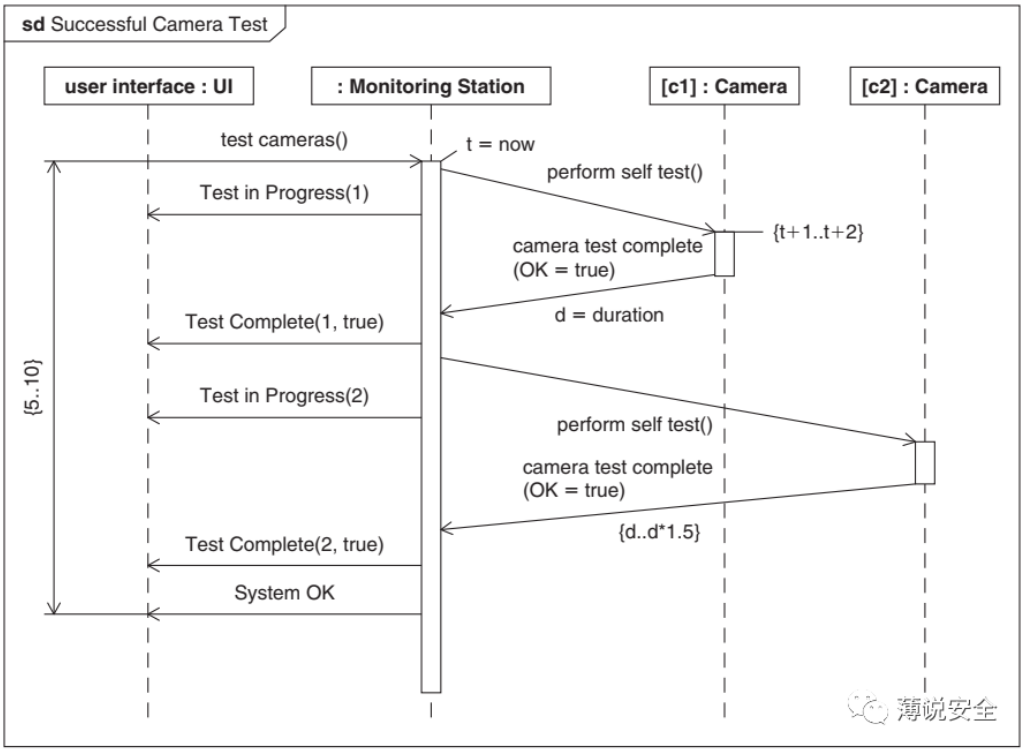
TECHNIQUE/MEASURE	Ref	SIL 0	SIL 1	SIL 2	SIL 3	SIL 4
1. Data Modelling	D.65	R	R	R	HR	HR
2. Data Flow Diagrams	D.11	-	R	R	HR	HR
3. Control Flow Diagrams	D.66	R	R	R	HR	HR
4. Finite State Machines or State Transition Diagrams	D.27	-	HR	HR	HR	HR
5. Time Petri Nets	D.55	-	R	R	HR	HR
6. Decision/Truth Tables	D.13	R	R	R	HR	HR
7. Formal Methods	D.28	-	R	R	HR	HR
8. Performance Modelling	D.39	-	R	R	HR	HR
9. Prototyping/Animation	D.43	-	R	R	R	R
10. Structure Diagrams	D.51	-	R	R	HR	HR
11. Sequence Diagrams	D.67	R	HR	HR	HR	HR
Requirements: 1) A modelling guideline shall be defined and used. 2) At least one of the HR techniques shall be chosen.						

EN50128 Table A.17 建模技术

在上表中，常用的建模方法有：

数据流图——描述数据如何由输入逐步流向输出的过程；
控制流图——描述由输入经过一系列控制动作到输出的过程；
状态机图——描述系统不同状态之间的转换关系；
真值表——描述一个复杂的组合逻辑关系；
序列图——描述不同组成部分通过信息交互的时序关系；

结构图——描述组件之间的层次关系。



序列图示例

这些软件建模方法属于软件通用的设计方法，在UML、SysML软件建模语言中就有上述建模方法，属于半形式化类方法。

注意这些建模方法在项目中使用，需要让项目中与软件架构关联的人理解一致，需要建立建模方法的使用指南，以规范其编写要求。

以上作为软件架构的通用性要求，软件缺陷为系统性失效，不存在失效概率的问题，因此，如果写的代码没有bug，它百分之百是按照定义的需求去执行。但是，有两个问题是安全软件需要考虑的，第一，软件不可避免会存在bug；第二，软件的实现与它所运行的硬件，与它所接口的外部系统相关联，任何与它关联的外部环境发生改变，都会对软件的预期行为产生影响，因此，安全软件不仅要考虑正常情况下的预期行为，也要考虑故障和干扰情况下的预期行为。

05 软件架构设计应用技术

在EN50128中的A.3表，列举了软件架构可供选择的技术方法，其中

2-14,16项是底层的安全设计技术，其中较为常用的是Fault detection & Diagnosis，与硬件或外部接口相关联；Graceful degradation作为fail-operational的一种实现方式，用于确保故障情况下的功能依然保持一定的可用性。对于软件的安全技术，应该适当地选择使用，毕竟增加了软件的复杂度，也加大了系统性失效的可能，而且安全技术往往难以兼顾可测试性。

防御性编程作为SIL1-SIL4都高度推荐使用的技术，是最常用的软件安全技术，用于检查软件执行中不正确的数据流、控制流和数据值情况下的预期行为，一种是防护软件自身设计缺陷造成的问题，如变量的范围检查、检查输入值的可信性、程序入口检查入参的类型、大小和

范围；另一种是防护外部环境输入的不受控造成的问题，如检查物理变量值输入的有效性、滤波处理、配置数据的完整性和软件自身的完整性。

TECHNIQUE/MEASURE	Ref	SIL 0	SIL 1	SIL 2	SIL 3	SIL 4
1. Defensive Programming	D.14	-	HR	HR	HR	HR
2. Fault Detection & Diagnosis	D.26	-	R	R	HR	HR
3. Error Correcting Codes	D.19	-	-	-	-	-
4. Error Detecting Codes	D.19	-	R	R	HR	HR
5. Failure Assertion Programming	D.24	-	R	R	HR	HR
6. Safety Bag Techniques	D.47	-	R	R	R	R
7. Diverse Programming	D.16	-	R	R	HR	HR
8. Recovery Block	D.44	-	R	R	R	R
9. Backward Recovery	D.5	-	NR	NR	NR	NR
10. Forward Recovery	D.30	-	NR	NR	NR	NR
11. Retry Fault Recovery Mechanisms	D.46	-	R	R	R	R
12. Memorising Executed Cases	D.36	-	R	R	HR	HR
13. Artificial Intelligence – Fault Correction	D.1	-	NR	NR	NR	NR
14. Dynamic Reconfiguration of software	D.17	-	NR	NR	NR	NR
15. Software Error Effect Analysis	D.25	-	R	R	HR	HR
16. Graceful Degradation	D.31	-	R	R	HR	HR
17. Information Hiding	D.33	-	-	-	-	-
18. Information Encapsulation	D.33	R	HR	HR	HR	HR
19. Fully Defined Interface	D.38	HR	HR	HR	M	M
20. Formal Methods	D.28	-	R	R	HR	HR
21. Modelling	Table A.17	R	R	R	HR	HR
22. Structured Methodology	D.52	R	HR	HR	HR	HR
23. Modelling supported by computer aided design and specification tools	Table A.17	R	R	R	HR	HR

Requirements:

1) Approved combinations of techniques for Software Safety Integrity Levels 3 and 4 are as follows:
   a) 1, 7, 19, 22 and one from 4, 5, 12 or 21;
   b) 1, 4, 19, 22 and one from 2, 5, 12, 15 or 21.
2) Approved combinations of techniques for Software Safety Integrity Levels 1 and 2 are as follows: 1, 19, 22 and one from 2, 4, 5, 7, 12, 15 or 21.
3) Some of these issues may be defined at the system level.
4) Error detecting codes may be used in accordance with the requirements of EN 50159.

NOTE Technique/measure 19 is for External Interfaces.

EN50128 A.3

06 已存在软件组件的使用

在ISO26262和EN50128中都规定了在安全软件中如何复用一个已存在软件组件，存在两种情况，会使用已存在组件：

来自于公司外部的CTOS组件；
以前开发组件的再利用。

首先一个组件能够被重复使用，它的接口必须能清晰识别，确定其应用环境，实现的规格也是明确的。在EN50128中，如果应用于SIL3和SIL4，需要分析已存在软件可能的失效对整体软件的影响，以及检测已存在软件失效的策略，如包装技术。在ISO26262.8中，第12章规

定了对已存在组件的鉴定要求。两个标准均要求对已存在软件进行鉴定，确定可用的功能、组件版本与配置、应用环境的假设、关联的安全完整性等级、组件残余缺陷情况，并对鉴定过程进行验证。

07 软件组件的相互影响

当软件由不同安全完整性等级的组件组成时，在EN50128 7.3.4.9和ISO26262-6 7.4.8的要求一致：

除非有证据表明高级别组件和低级别组件之间彼此独立，从时间分区和空间分区两个维度，其它情况都应按照最高等级要求开发。

在ISO26262-6提出有两种不同组件分区的方法，第一种是软件分区，从执行时序、数据保护、组件之间的数据交互方面考虑组件之间的干扰影响，第二种是硬件保护机制的支持，如MPU；第三种是操作系统或虚拟化层对不同组件互不干扰的支持。

最后，回顾一下五方面主要内容：

软件需求、软件架构与组件的关系；
软件架构需涵盖的内容；
安全软件应用技术；
如何应用已存在软件；
不同安全等级软件的影响分析

在不同标准中，架构设计还有各自侧重的部分，ISO26262-6对软件安全分析有相应要求，EN50128安全分析的工作放在系统层面进行，要求从系统功能和接口的角度进行分析。EN50128在架构设计阶段对软件设计方法（建模指南、设计指南和编码规则）有更为详细的定义，并需要在架构阶段完成软件集成测试规范和软硬件集成测试规范。

(/)

地址：上海市普陀区云岭西路600弄6号楼7层

邮编：200333

电话：+86 21-62655001-5886

邮箱：marketing@ticpsh.com

上海控安 (http://www.ticpsh.... 启明星辰知白讲堂 (http://ww... SHCERT (https://www.cert.or... 数字工
上汽培训中心 (https://learnin... BSI英标 (https://www.bsigro... 中国赛宝实验室 (https://www.... SAE Ir
富士通南大 (https://www.fujit... TUV NORD (https://www.tuv... SGS (https://www.sgsonline.... 焉知汽
机械工业出版社 (http://www.c... 工业得到 (https://app8nzxnlb... 博勘咨询 (http://www.borsco... 上汽零