文章编号：1005-8451（2015）03-0014-05

基于SCADE的安全软件开发方法研究

陈淑珍，陈荣武，李　耀
（西南交通大学　信息科学与技术学院，成都　610031）

摘　要：针对传统软件开发方式已经不能满足高安全性系统安全性、完整性的需求，本文提出了基于SCADE的安全软件开发方法，分析SCADE开发的原理、流程及应用方式，并以城市轨道交通列车运行控制系统的区域控制器ZC为例，基于SCADE对ZC列车管理功能进行建模和验证。通过实例分析，证明基于SCADE的软件开发方法，可以有效保障高安全性系统的安全性和完整性，为其提供了一种新的开发方式。

关键词：SCADE；软件安全；软件建模；区域控制器
中图分类号：U284.482：TP39　**文献标识码：**A

Method of SCADE-based safety software development

CHEN Shuzhen, CHEN Rongwu, LI Yao
(School of Information Science and Technology, Southwest Jiaotong University, Chengdu　610031, China)

Abstract: The traditional methods of software development couldn't meet the requirements of high security and integrity of safety critical system. The article proposed the method of SCADE-based safety software development, analyzed the principle, process and application methods of SCADE. The ZC of Urban Transit was taken as an example, through modeling and verifying the train management functions of ZC, it was proved that this method provided a new method which could efficiently ensure the high security and integrity of safety critical system.

Key words: SCADE; software safety; software modeling; Zone Controller(ZC)

高安全性应用开发环境（SCADE，Safety Critical ApplicationDevelopment Environment），用基于模型的方式为高安全性系统提供完整的嵌入式开发解决方案，具有开发质量好、效率高、成本低、风险小、验证时间短等优点。

SCADE开发方式解决了传统方式很多不足[1]。SCADE开发十分方便，且效率高，投入低，能够保证软件的质量，从以“代码”为核心转变为以“模型”为核心。以“代码”为核心的开发方式，开发人员专注于编码，主要关心的是软件功能的实现；而以“模型”为核心的开发方式摆脱了枯燥的编码，开发人员的注意力转移到了软件功能的正确性上，体现了软件设计的真正意义，是软件开发方式的发展方向。SCADE具有严格的数学语义，需求表达明确、无二义，能够确保软件的安全性和可靠性。

区域控制器（ZC，Zone Controller）是基于通信的列车控制系统CBTC的地面核心控制设备，保证列车行车安全。ZC系统安全性要求高，结构复杂，系统软件需要满足EN 50128SIL 4级标准，开发要求极为苛刻，传统方式开发难度大。本文以其列车管理功能为例分析了SCADE开发的原理及流程，证明SCADE开发模式完全能够满足ZC系统需求，而且具有效率高，成本低，结构清楚，软件质量好，安全性和可靠性有保障等优点。

1　理论基础

SCADE开发基于同步假设，而且以Lustre为核心，这两点也决定了SCADE开发的特点。

1.1　反应式系统

反应式系统是相对于转换式系统而言的，其与环境保持不间断交互，随时接受来自环境的刺激，运算后做出响应，环境可能利用该响应继续为系统提供新的输入。这个过程可能不是顺序的，其行为具有并发性、不终止性等特点，并不是最终产生一个输出。

1.2　同步假设

同步理论模型是基于周期执行模型的扩展，在

收稿日期：2014-08-02
作者简介：陈淑珍，在读硕士研究生；陈荣武，高级工程师。

每个周期内，模型以传感器采样或通信等方式获得输入，随即进行处理，最后将处理结果输出给目标模块或系统。在一个计算周期内，模型和环境之间没有任何交互，程序的行为是确定的。

同步假设是假设反应式系统响应速度足够快，系统接收环境输入后立即响应，然后产生输出，并等待下一个输入，在此期间，系统内部状态保持不变，如图 1 所示。

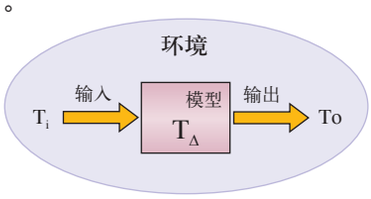

图1　同步假设模型

在实际中，由于技术或成本的限制，一般是通过控制系统获得任意两次输入的时间间隔大于系统的响应时间的方式来实现同步假设的。即：

$\forall i,\ T_{i+1}-T_i>T_\Delta$

T_i—系统第 i 次输入的时间

T_{i+1}—系统第 i+1 次输入的时间

T_Δ—系统响应的时间

1.3　Lustre

Lustre 是基于数据流的同步编程语言，适用于反应系统，通过划分物理时间的方式建立同步模型。Lustre 每一个变量都代表一个数据流，流是一个给定数据类型的值的无限序列，具有数值和时钟两个特性，如等式 $x=f$，表示 $\forall n,\ x_n=f_n$，n 为周期。Lustre 提供的时间运算符，对数据流进行采样，也可以获取之前周期的流值，对控制系统的建模提供了方便。

如一个带有两个输入 a,b 和重启按钮 reset 的计数器，有如下需求：

（1）初始显示 0；

（2）当按下重启按钮时，计数器输出 0；

（3）两个输入和大于 50 时，计数器输出不变；

（4）否则计数器每个周期的值加 1。

该计数器的 Lustre 表达如图 2 所示：

初始化运算符“→”用于初始每个流第一个周期的值，用 n 表示周期，则 $\forall n$，

$$\left(0\rightarrow(a+b)\right)_n=\begin{cases}0 & n=1\\ a_n+b_n & n\neq 1\end{cases}$$

```
node Counter(a,b : int; reset : bool) return(count : int) ;
var c : int;
let
   c = 0 → (a+b);
   count = 0 → if (reset) then 0
               else if (c > 50) then pre(count)
               else pre(count)+1;
tel
```

图2　计数器Lustre表达1

“Pre”运算符表示取上一周期的值，用 n 表示周期，则 $\forall n$：

$$\left(pre(count)\right)_n=\begin{cases}nil & n=1\\ (count)_{n-1} & n\neq 1\end{cases}$$

Lustre 是一种声明式语言，关注计算机应该做什么，不是命令式语言所关注的计算机应该如何做。Lustre 语言的语句具有顺序无关性；例如：此计数器中语句：c=0 → (a+b) 在图 2 和图 3 中的陈述顺序是不一样的，但是它们所表示的节点在本质上一样。

```
node Counter(a,b : int; reset : bool) return(count : int) ;
var c : int;
let
   count = 0 → if (reset) then 0
               else if (c > 50) then pre(count)
               else pre(count)+1;
   c = 0 → (a+b);
tel
```

图3　计数器Lustre表达2

理解理论基础对 SCADE 模型提取、理解等十分必要，特别是复杂系统。

2　模型建立

SCADE 模型是需求的一种明确、无歧义的表达，提供文本和图形两种建模方式，文本方式使用 Lustre 语言，图形方式包括安全状态机和数据流图。这些建模机制都建立在严格的数学模型之上，具有严格的数学语义，保证了模型的完整性、精确性、一致性和无二义性。在实际使用中，图形方式使用相对较多，而且图形化建模时，可以观察到节点的 Lustre 表达方式，还可以自动转化为文本节点。

2.1 安全状态机

安全状态机常用于描述控制系统的状态及逻辑功能，是控制流和判定逻辑的直观模型，常用于离散系统建模，具有顺序控制，并发控制，层次控制的结构特点[5]。安全状态机是有限状态机的拓展，其中的状态也是有限的。

安全状态机用“状态”描述系统的一种模式，用“迁移”描述模式之间的转移，用“信号”反应状态机和环境之间的交互。迁移分为强连接迁移、弱连接迁移及同步连接迁移，每个迁移都具有各自的优先级，迁移由迁移标识控制。迁移触发后，在进入状态，处于状态和离开状态时都可能完成相关操作。根据SCADE的这些特点，安全状态机能够描述复杂系统的逻辑结构，确保模型的确定性。对控制结构丰富而数据处理较少的模型，十分适合使用安全状态机。

ZC是典型的高安全性系统，其安全性、可靠性关系到CBTC系统的正常运行，SCADE开发完全满足ZC系统的特点和需求。列车管理是ZC系统的重要功能之一，根据列车的不同行为，可以将列车状态简单概括为以下几种状态：初始状态，正常运行状态，注销状态。ZC的SCADE列车管理功能安全状态机如图4所示。

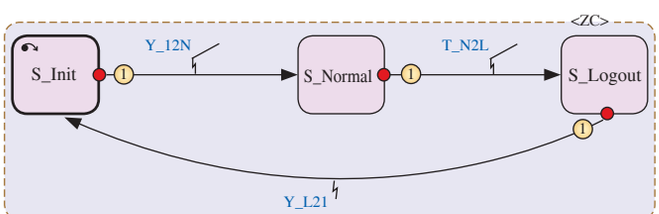

图4 SCADE ZC列车管理功能安全状态机图

SCADE安全状态机支持状态机嵌套，但每个状态机都必须有明确的初始状态，图4中黑粗边框的状态即为初始状态。图4中各状态及变迁含义如表1所示。列车首先处于初始状态S_Init，当T_I2N转移条件满足时，转移到列车正常运行状态；当T_N2L条件满足时，则由正常运行状态转移到列车注销状态；当Y_L2I条件继续满足时，则由列车注销状态转移到列车初始状态。

2.2 数据流图

数据流图善于描述数据的处理过程，反应时序及因果关系，常常用于连续系统的建模。数据流图

表1 列车管理功能状态机图的状态含义

状态	含义	变迁	含义
S_Init	列车初始化状态	T_I2N	初始状态向正常运行状态转移条件
S_Normal	列车正常运行状态	T_N2L	正常运行状态向注销状态转移条件
S_Logout	列车注销状态	Y_L2I	注销状态向初始状态转移条件

通过用户定义的输入变量接收外界信号，经过模型处理后，再以用户定义的输出变量为接口输出给目标模块或系统。节点是数据流图的最基本的功能单元，类似于C语言的函数。数据流图提供算术运算、逻辑运算、比较运算，时间运算等运算符，SCADE利用这些运算符及自定义的运算符，以图形化的方式构建新的更复杂的节点，完成更复杂的功能。对复杂控制结构较少而数据处理较多的模型，适合使用数据流图。

ZC折反请求的SCADE数据流图如图5所示，当ZC获取列车状态运算符GetTrainState解析到列车信息I_Train_Info为折返列车REVERSE_TRAIN并且获取进路类型运算符GetRouteType解析到进路信息I_RoutInfo为折返进路REVERSE_ROUTE时，触发折返转移T_M2R，列车将进入折返状态。

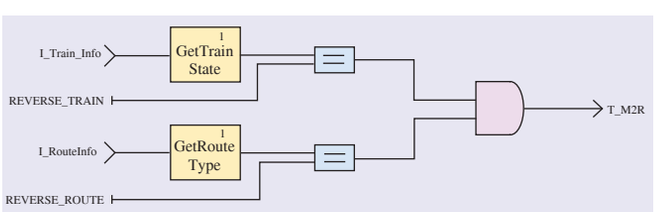

图5 折返条件数据流图

这两种建模方式不是独立的，可相互结合使用。

3 模型验证

系统可靠性、安全性依赖于系统功能的正确性。在软件开发流程中，SCADE的验证处于代码之前，表达需求时就对功能进行了验证。开发人员的注意力转移到了需求的合理性、模型的正确性，提高了软件的质量和开发效率。

对于已经建立的模型，SCADE提供两种方式进行验证：模拟仿真和形式化验证。在系统开发早期使用模拟仿真，而在设计的最后阶段使用形式化验证。

3.1 模拟仿真

SCADE仿真对象可以是整个系统，也可以是一

个模块或子系统。对仿真过程中模型的输入、输出以及内部变量，SCADE 仿真器提供了文本和图形两种观察方式。文本方式展示输入、输出或所观察变量在当前周期的值，图形方式展示了所有周期的数据，直观的反应了数据的变化趋势。

SCADE 的仿真并不是直接对图形模型进行仿真，而是先生成 C 代码，再通过执行 C 代码达到仿真的目的，即是对代码生成器生成的目标代码的执行。这样模拟仿真的结果和将来目标代码的执行结果完全相同。

SCADE 仿真是仿真器在当前模型下对外部输入的响应，由于模型已经建立，所以可以通过保存仿真过程中所有周期输入数据的方式保存仿真场景。仿真器载入仿真场景后，模型和输入数据都具备，所以可以继续仿真。

SCADE 仿真器支持单步、多步、单步后退及多步后退等仿真方式，还支持批处理模式仿真，在多测试用例中十分方便。

3.2 模型测试覆盖率MTC

仿真是通过观察模型对指定输入的响应情况来分析系统功能的正确性，所以测试案例的数量及质量关系到测试的完备程度及深度。SCADE 提供 DC 和 MC/DC 两种覆盖率准则，定量分析仿真测试的完备程度及深度，包括覆盖率提取，覆盖率分析，覆盖率处理，生成覆盖率报告 4 个步骤。如果覆盖率未达 100%，SCADE 将给出未覆盖的路径。

在开发的早期阶段，覆盖率分析可以帮助开发人员发现众多问题，如模型设计错误，测试案例不足等。模拟仿真是对模型的验证，MTC 是对仿真的验证，即是验证的验证。

3.3 形式化验证

SCADE 提供形式化验证的方式保证安全需求。

传统的形式化验证方法主要缺点是对原始设计进行模型提取时，一般需要使用某种语言构造数学模型，再利用相关工具进行数学推理，这对验证者的数学技能和经验要求较高。SCADE 形式验证避免了这些不足，其建模语言即是验证语言，一种语言完成建模和验证两个功能。

SCADE 形式化验证与仿真测试，如表 2 所示。

3.3.1 提取安全属性

表2 形式验证与模拟仿真的区别

对象 / 类别	SCADE形式验证	模拟仿真
完备性	严格的数学推理，对系统做详尽的验证	不具有完备性
测试案例	不需要任何测试案例	质量依赖于测试案例的数量及质量
输出	每个安全特征模型只有一个简单的输出	可能会有多个输出，观察较为复杂
分析	属于静态分析	属于动态分析

安全属性为保证系统不会发生危险的属性，如“CBTC 系统中，列车不能越过 MA 停车”。安全属性一般来自于设计规范及需求等。

3.3.2 方式的转换

建立安全特征观察器 Observer，即在形式化验证器中用数据流图的方式将安全需求转换为图形表达。Observer 本质上也是 SCADE 的节点，但是多了一些特定的验证运算符，而且输出始终为一 Boolean 变量且应当为 true。

这两步都是人工完成，对设计者的专业知识及技能有一定的要求。验证器不能检测 Observer 模型的正确性，如果提取了错误的安全属性，或 Observer 建立错误，验证器将以错误的安全属性验证模型，这也是 SCADE 的不足之一。

3.3.3 验证安全属性

安全属性的验证是证明该属性在目标模型所有周期和所有场景下都是正确的。SCADE 将待验证模型和 Observer 模型结合，利用待验证模型的输入、输出信号自动验证模型是否满足安全需求。如果不能，Observer 输出 false，同时 SCADE 给出一个可导入仿真器分析的反例；如果满足，Observer 输出 true, 同时 SCADE 给出安全证明。

ZC 列车管理功能有如下安全要求：每一时刻，列车最多只能向一个状态转移。对 SCADE 功能模型，该安全需求的含义是：同一时刻最多只有一个转移条件为真。SCADE 提供了 Sharp 运算符 #，当其输入条件最多只有一个为真时，Sharp 输出 true 信号，其 Observer 模型如图 7，图中变量在表 1 中有注明。模型的输入为列车状态转移的所有条件。当 T_I2N，T_N2L 和 Y_L2I 共 3 个状态转移条件最多只有一个

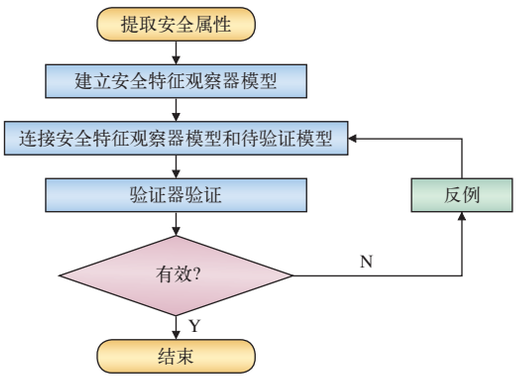

图6　形式验证过程

为真时输出结果 Result 才会为真。

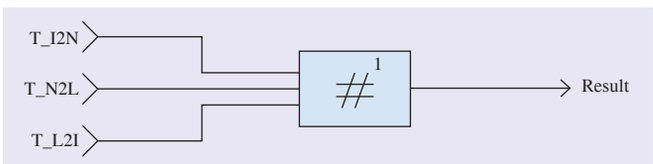

图7　安全特征观察器模型

SCADE 为验证功能提供了大量的预定义运算符，降低了验证的难度和复杂度。通过不到一秒的验证，SCADE 证明模型满足此安全需求。

4　目标代码

SCADE 开发的最终目标就是生成可信代码。SCADE 的代码生成器 KCG 通过了 DO-178B A 级、IEC 61508 SIL3、EN 50128 3/4、IEC 60880 认证，生成的代码满足安全特性，如有界的堆栈。KCG 将 SCADE 的图形模型作为输入，根据用户设定的参数，输出目标代码和可跟踪文件，如图 8 所示。

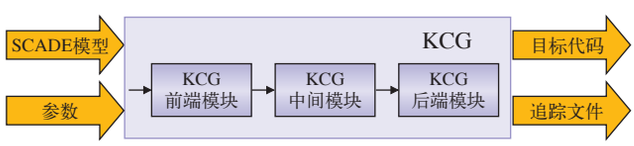

图8　KCG模型

KCG 前端模块将模型文件转换为 SCADE 程序，复制注释信息，删除图形相关信息，并进行语法语义检查；中间模块是编译核心和优化器；后端模块生成目标代码和可追踪文件 [5]。

对于已经完成设计的模型，只需点击“生成”按钮，SCADE 自动将图形模型转换为 ANSIC 或 ADA 代码和可追踪文件，包括模型中的注释信息。

4.1　图形模型转换为Lustre语言

将节点、状态图等图形元素转换成 Lustre 语言描述的模型，并将多个文件模型整合到一个文件中。

4.2　Lustre语言转换为目标语言

目标代码是完全面向工程的代码，可以不做任何修改直接将代码移植到目标模块中，但 SCADE 生成的代码可读性不如人工代码。

SCADE 生成的代码执行效果与之前仿真效果完全一致，在保证功能正确后，不需要对目标代码进行任何验证，从而节省了大量测试时间，而且代码具有可追踪、可移植、模块化、代码优化、有限运行时段等特性，同时 KCG 支持批量生成代码。

5　结束语

SCADE 开发方式覆盖了嵌入式开发从需求到代码的所有流程，包括需求管理、需求建模、模型检查、模拟仿真、测试覆盖率分析、形式验证、文档生成、代码生成等。其操作方便，使用形式化的方法保证设计的无二义性、正确性，自动生成高质量代码，能够在软件开发的早期阶段发现错误，节约软件开发成本，提高软件开发的效率和质量，缩短软件面市时间，提高软件开发的自动化程度，并为软件认证提供支持，非常适合高安全性系统的开发。

但 SCADE 也有不足之处需要改进，如对于复杂的逻辑功能，需要导入用户自定义的节点，但 KCG 生成代码时对此部分没有验证。

参考文献：

[1] 张　路 . 基于 SCADE 的 CBTC 区域控制器软件开发 [D]. 北京：北京交通大学，2010.

[2] 高　霖 .CBTC 区域控制系统中列车管理的建模与分析 [D]. 北京：北京交通大学，2007.

[3] 胡钢伟，李振水，高亚奎 . SCADE 软件开发方法研究 [J]. 系统仿真学报，2008（S2）：286-288.

[4] 林　枫 . 基于 SCADE 的形式化验证技术研究 [J]. 测控技术，2011，30（12）：71-74.

责任编辑　徐侃春