文章编号：2095-2163(2023)09-0141-06 中图分类号：TP311.5 ；R318.6 文献标志码：A

基于 BehaviorTree 的医疗仪器通信协议解析平台设计

杨 蓉，郑建立，谢雅欣，周文杰
（上海理工大学 健康科学与工程学院，上海 200093）

摘 要： 物联网、大数据等技术的发展，推动了医疗信息化的发展。通过将具备数据输出功能的医疗仪器连接物联网、采集医疗设备大数据需求也越来越普遍。除了影像类医疗仪器广泛采用 DICOM 标准外，其他医疗仪器的信息接口标准化程度相对滞后，医疗仪器的信息集成问题亟待解决。本文提出了一种基于 BehaviorTree 的协议转换方法，该方法具有代码耦合度低、扩展性强、复用性优、决策逻辑和数据分离的特点，避免了针对不同通信协议均要编写相应的解析和处理程序，使协议的解析和处理具有更好的灵活性和普适性，有效降低协议解析的难度。

关键词： 医疗仪器；信息集成；BehaviorTree；协议转换

Design of communication protocol analysis platform for medical instruments based on behavior tree

YANG Rong, ZHENG Jianli, XIE Yaxin, ZHOU Wenjie
(School of Health Science and Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China)

【Abstract】 The development of technologies such as the Internet of Things and Big Data has driven the development of medical informatics. The need to connect medical instruments with data output functions to the Internet of Things and to collect big data from medical devices is also becoming more and more common. Apart from the DICOM standard, which is widely used for imaging medical instruments, the standardization of information interfaces for other medical instruments has lagged behind, and the problem of information integration of medical instruments needs to be solved. This paper proposes a protocol conversion method based on BehaviorTree, which is characterized by low code coupling, high scalability, excellent reusability, and separation of decision logic and data, avoiding the need to write corresponding parsing and processing procedures for different communication protocols, making the parsing and processing of protocols more flexible and universal, and effectively reducing the difficulty of protocol parsing.

【Key words】 medical instruments; information integration; BehaviorTree; protocol conversion

0 引 言

物联网、大数据等信息技术的不断发展，推动了医疗信息化的发展。通过将具备数据输出功能的医疗仪器连接物联网、采集医疗设备大数据需求也越来越普遍。除了医学影像类的医疗仪器，如 CT（Computed Tomography）、MR（Magnetic Resonance）、US（Ultrasonic）等数据接口已经广泛采用了医学数字影像传输标准 DICOM（Digital Imaging and Communications in Medicine）。但其他医疗仪器如监护仪、麻醉机、呼吸机等产品的信息接口标准化程度相对滞后，仍然缺乏标准数据接口，诸多厂商自定义信息接口、数据通信协议以及通信方式，导致医疗仪器信息集成困难[1-2]。

针对医疗仪器信息集成困难，国际组织医疗健康信息集成协会（Integrating the Healthcare Enterprise，IHE）开发了病人护理设备（Patient Care Device，PCD）技术框架[3-4]。PCD 技术框架规范了通信场景以及医疗设备与医疗设备数据接收单元之间的流程，从而促进医疗环境中设备数据的交换效率，但是应用不够广泛。目前常见的办法就是针对已知的协议编写一对一的通信程序，效率低下。研究通用的通信协议转换技术，将非标准化的仪器私有协议转换为标准化协议具有重要的意义。已有研究提出基

作者简介：杨 蓉（1998-），女，硕士研究生，主要研究方向：健康物联网；郑建立（1965-），男，博士，副教授，硕士生导师，主要研究方向：医学信息系统与集成技术、医学仪器嵌入式控制系统；谢雅欣（2001-），女，本科生，主要研究方向：医学信息工程；周文杰（2001-），男，本科生，主要研究方向：医学信息工程。
通讯作者：郑建立 Email：zhengjianli163@163.com

收稿日期：2022-10-23

于 JS 引擎的医疗仪器协议转换技术的解决方案,通过协议转换引擎执行协议转换脚本,实现对多种消息的医疗协议进行转换[5];基于 OpenPLC 的医疗仪器通信协议解决方案,针对目标协议通过拼接组件生成相应配置文件,实现协议转换[6]。在非医疗仪器领域,有通过协议模版来进行转换的集成网关案例,通过将协议分成帧头、帧尾、位处理以及函数处理 4 部分,对协议模版进行定义[7-8];基于通用描述符的协议转换,通过描述符的生成,处理以及不断修改,完成整个协议转换过程[9]。但现有解决方案,对于结构较复杂的协议,解析流程不直观,可拓展性差。

本文通过对现有的协议进行分析,提出了一种基于行为树(Behavior Tree)的协议描述新方法,利用行为树有序的逻辑分层、代码耦合度低、决策逻辑和数据分离优点解决协议解析流程不直观与可拓展性差问题。

1 BehaviorTree 技术

行为树(Behavior Tree)是包含逻辑节点和行为节点的倒挂型树结构,用模块化的方式描述了一组有限任务之间的切换。其本质是一段逻辑代码,可以通过行为树编辑器转换为各种编程语言代码[10]。行为树通过节点(Node)描述行为逻辑,行为树示意如图 1 所示。

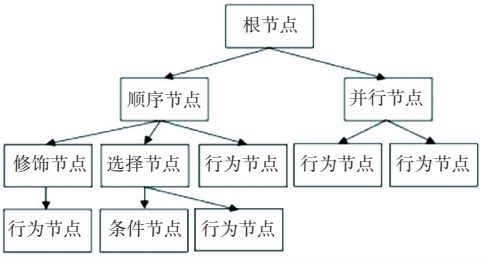

图 1 行为树示意图

Fig. 1 Schematic diagram of BehaviorTree

1.1 节点分类

行为树可以分为 3 部分:树的根部、树的枝干、树的末端。树的根部是根节点(Root),即行为树逻辑执行的起点;树的枝干实现行为逻辑的选择判断,由组合节点(Combination Node)和修饰节点(Decoration Node)构成;树的末端是叶节点(Leaf Node),用来实现具体的动作[11]。行为树节点分类,见表 1。

表 1 行为树节点分类

Tab. 1 Node classification of behavior tree

行为树	树的根部	根节点(Root)	
	树的枝干	组合节点(Combination Node)	顺序节点(Sequence)
			随机节点(Random)
			并行节点(Parallel)
			选择节点(Select)
		修饰节点(Decoration Node)	
	树的末端	叶节点(Leaf Node)	条件节点(Condition)
			行为节点(Action)

1.2 节点行为

行为树在遍历过程中,由组合节点和修饰节点决定下一个执行节点,通过动作节点作出行为决策,执行定义好的动作,当叶子节点执行完毕后,会将执行结果反馈给上层节点[12]。在这一过程中,每个节点在执行后,都会向上层节点返回执行状态,执行状态包括成功(Success)、失败(Failure)、运行中(Running),上层节点会根据子节点的返回状态决定后续的逻辑执行过程。

1.2.1 根节点

根节点是行为树执行的起点,只能是选择节点和序列节点,当对行为树初始化或执行过程中无法找到有效的动作节点时,执行指针回到父节点。

1.2.2 组合节点

包括顺序节点、随机节点、并行节点、选择节点:

(1)顺序节点:该节点的遍历方式为从左到右依次执行其所有子节点,如果有子节点的执行结果返回值是 Success,就会继续执行该节点的其他子节点。顺序节点只有在子节点执行结果返回值都是 Success 时,父节点的返回值才会置为 Success。

(2)随机节点:该节点的执行方式是随机执行其所有子节点,如果有一个节点的执行结果返回值是 Success 或者 Running 时,就会向父节点返回当前这个子节点的执行结果返回值并停止执行其他的所有子节点。

(3)并行节点:该节点与其子节点的执行结果返回值无关,执行方式是左至右开始依次执行所有的子节点。

(4)选择节点:该节点的遍历方式是从左到右依次执行所有子节点,当第一次遇到子节点执行结果返回值为 Success 时,则停止继续执行其他子节点;当子节点执行结果返回值为 Failure 时,则继续执行其他子节点,直至一个子节点返回 Success 或者 Running,否则向父节点返回 Failure;当子节点执行

结果返回值为 Running 时，则停止继续执行其他子节点。

1.2.3　修饰节点

修饰节点（Decorator）只能有一个节点，该节点决定是否可以执行树中的分支或者单个节点，本质上是一个条件，可以用来控制行为树执行流的跳转等。

1.2.4　叶节点

叶节点包括条件节点与行为节点

（1）条件节点：此节点对应编程语言中用到的 if-else结构，用来判断设置的逻辑条件与当前的环境状态是否一致，如果一致，就会向父节点返回 Success，否则的话返回 Failure。

（2）行为节点：用来执行节点关联的特定动作，并根据动作执行状态向父节点返回执行结果。这类节点随应用目的的不同有很大的差异性，需要自行编写，这也是本文主要的工作内容。

2　基于 BehaviorTree 协议解析平台设计

2.1　通信协议解析框架

医疗仪器通信协议的解析是将输入的字节流中医疗仪器数据提取出来，并将其转换为符合目标格式规范的数据帧输出，通信协议解析框架示意图如图 2 所示。输入模块将医疗仪器的数据传入到解析模块；解析模块从输入模块中提取所需的数据项，并将这些数据项传入转换模块，对传入的数据进行校验得到正确的数据帧，再从正确数据帧中，根据协议的定义，将原来的数据项提取出来；转换模块将解析模块所传入的原始数据项，根据要求的输出数据项格式对原始数据进行格式或者数值的转换；输出模块将转换模块所得到的数据帧按照网络协议进行传送。

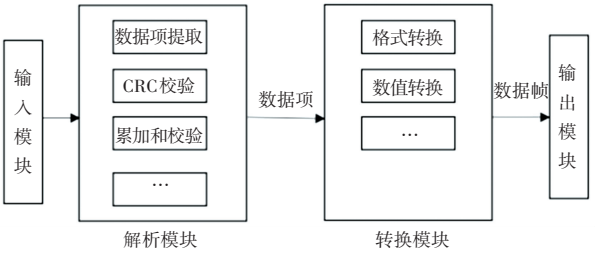

图 2　通信协议解析框架

Fig. 2　Communication protocol analysis framework

2.2　协议解析平台的设计

基于 BehaviorTree 的协议解析平台由协议解析引擎和低代码协议解析行为树编辑器两部分组成，如图 3 所示。基于 BehaviorTree 框架实现协议解析引擎，在 Groot 编辑器上已有的流程控制节点基础上扩展自编的动作节点，构成图形化低代码协议解析行为树编辑器，组合出协议解析行为树并保存为 XML 格式的协议配置文件。协议解析时协议解析引擎动态加载 XML 协议配置文件，即可完成对应的协议数据的解析，通过替换不同的协议配置文件即可实现不同协议数据的解析，而无需重新修改编译协议解析引擎，大大提高了可扩展性，降低了使用门槛。在这个过程中，使用了树的所有节点共享的黑板键/值存储方式对数据进行存储。

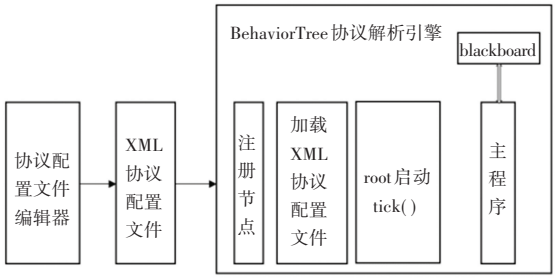

图 3　基于 BehaviorTree 的协议解析平台组成框架

Fig. 3　Framework for protocol analysis platform based on BehaviorTree

2.2.1　低代码协议解析行为树编辑器设计

2.2.1.1　行为树编辑器节点定义

通过分析多种通信协议，定义以下数据处理节点。

（1）取数：通过分析总结出定长、类型-长度-值和分隔符 3 种协议数据封装方式，设计了 3 种取数节点。定长取数节点获取指定长度的值，类型-长度-值节点处理有数据长度字段的变长值，分隔符取数节点处理有特殊字符标识的开始与结束。

（2）校验：验证协议数据单元正确性，包括循环冗余校验节点、校验节点、校验和节点。

（3）运算：包括算术运算节点、逻辑运算节点、关系运算节点、移位运算节点。

（4）转换：转换节点主要包括十六进制、字符串、整型与字符等数据类型之间的两两转换节点共 20 个，例如 整型转为字符串，十六进制转为整型等。

（5）映射：将协议解析结果表示成多种输出格式。例如 json 格式映射节点需要将信息项根据 ID 映射到名称、单位，并填入值。

2.2.1.2　行为树编辑器节点编写

自定义数据处理节点类型后需要对节点进行功能实现。这些节点都以行为树的动作节点形式实

现。节点的定义及编写分为 3 个步骤,以自定义取数节点(GetNByte)为例:

(1)配置 json 文件。打开 Groot 编辑器源码的 json 配置文件,在此处先声明节点名称(GetNByte)与节点的图案;

(2)实现节点功能。GetNByte 是在本文中规划的一种取数行为节点,GetNByte 节点根据输入的起始位置、固定长度取数,因此设置了输入端口获取数据起始位置与数据长度;

(3)注册节点。在行为树工厂函数中通过注册节点类型函数注册节点,此处实际类型与入参 ID 相同,且与 json 配置文件中名称一致;

完成程序编写后,自定义数据处理节点成功内置到低代码协议解析行为树编辑器中,如图 4 所示。

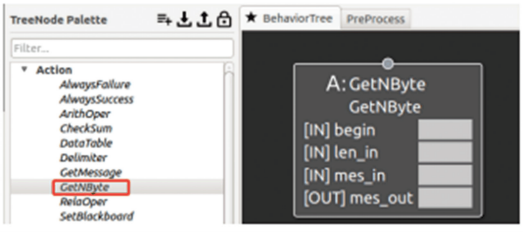

图 4 GetNByte 节点

Fig. 4 GetNByte node

2.2.2 协议解析引擎设计

2.2.2.1 注册节点

行为树中通过行为树工厂类进行节点注册,该类由 3 个容器来保存数据,通过注册节点类型的方法注册自定义的数据处理节点。例如注册一个节点,该节点的实际节点类型和在树中的名称保持一致,实际类型与入参 ID 相同,可减少错误。

2.2.2.2 加载 XML

行为树按照 XML 文件格式来设计、书写和保存,XML 直观显示了不同父树、子树的黑板间的映射关联与节点执行顺序。树加载和创建 XML 通过文本加载方法和文件加载方法实现。文本加载首先加载和解析文本、检查各项元素是否符合行为树的概念要求;其次,创建树和所有节点的实例、构造树之间和节点之间与端口的映射关系,再将树的节点信息绑定给树实例变量。

2.2.2.3 启动函数

行为树是一种树状的数据结构,树上的每一个节点都是一个行为,在行为树中 tick 函数是一个启用信号,行为树的运行方式就是从根节点以一定的频率发送 tick 给其他子节点,行为树执行时逻辑如图 5 所示,本文自定义数据处理节点的实现自行编写在 tick 函数中。

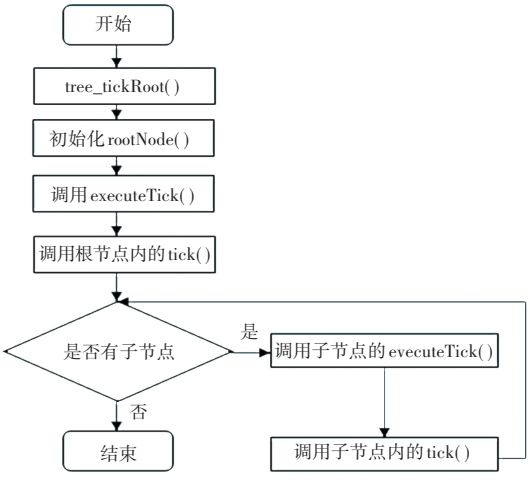

图 5 行为树执行时逻辑

Fig. 5 Behavior tree execution logic

2.2.2.4 调用黑板

黑板是树中节点传输数据的方式,所有节点共享。每棵树都有自己的黑板,通过编辑 XML 显式创建不同的父树、子树的黑板间的映射关联,调用黑板设置函数将数据写入黑板。

3 协议解析测试

本文选择医疗设备某呼吸机,呼吸机数据及说明见表 2。分别基于 C 语言的编程方法与基于行为树方法进行数据帧测试,测试该协议能否正确校验数据帧,并输出对应的解析结果。

表 2 呼吸机数据及说明

Tab. 2 Ventilator data and description

数据字节	说明	测试数据
1	SOH,为 01	01
2	Echo,为 24	24
3-4	DataCode1	36,39
5-X	Data1	20,30,2E,34
…	…	…
(n-2)-(n-1)	Checksum	43,31
n	CR,为 0D	0D

由表 2 可知,协议数据字节的长度不是固定的,以 0DH 作为协议数据单元的结束。该协议数据字节第 1 字节 01H 为协议开始标志,第 2 字节 24 为开始响应标志,从第 3 字节开始到第 $(n-3)$ 为数据位,第 $(n-2)$ 到第 $(n-1)$ 字节为从第 1 字节到第 $(n-3)$ 字节的 16 位累加和校验值,第 n 字节 0DH 为 Ascii 回车字符。根据输入数据帧协议,可将协议解析分为 3 步:第一步根据协议的开始与结束标志

取出协议数据;第二步为累加和校验;第三步协议数据转换。

3.1　基于行为树协议解析

基于行为树方法的协议解析步骤分为 3 步：

第一步:绘制行为树。利用低代码协议解析行为树编辑器绘制呼吸机数据协议解析行为树如图 6 所示，将该协议解析行为树导出为 XML 文件；

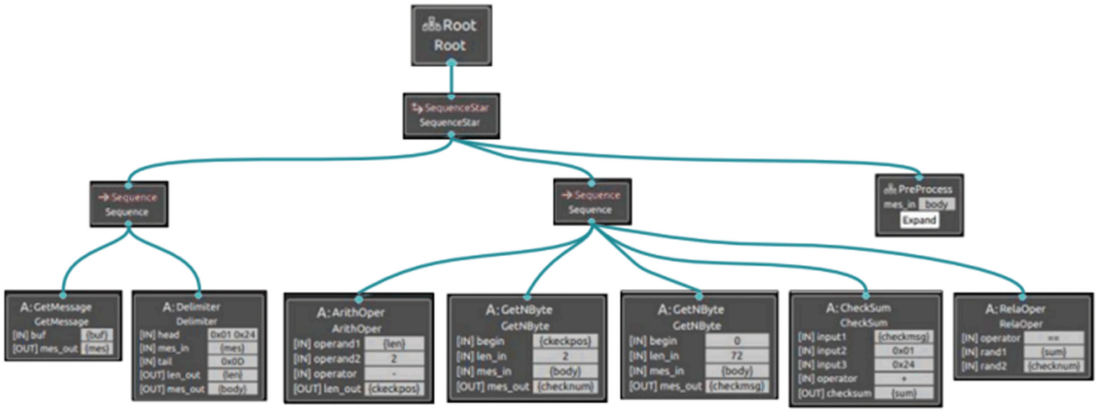

图 6　呼吸机数据协议解析行为树

Fig. 6　BehaviorTree of ventilator data protocol analysis

第二步:协议解析引擎加载 XML 文件。测试数据如图 7 所示,根据 XML 文件描述依次进行取数、校验、转换等过程得到测试结果如图 8 所示。

```
01 24 36 39 20 30 2E 34 36 46 20 30 2E 35 37 33 20 20 30 20
37 36 33 32 37 37 37 41 30 2E 30 30 37 44 20 20 30 20 42 35
20 20 30 20 42 38 30 2E 30 31 44 36 20 36 30 20 45 37 20 31
2E 30 45 38 20 31 2E 31 46 30 20 32 31 20 43 31 0D
```

图 7　测试数据

Fig. 7　Test data


```
{
    "ItemInfo":[
        {"itemID":"69","itemName":"Plateau time","itemVal":" 0.4","itemUnit":"s"},
        {"itemID":"6F","itemName":"Inspiratory time","itemVal":" 0.5","itemUnit":"s"},
        {"itemID":"73","itemName":"Mean airway pressure","itemVal":"  0 ","itemUnit":"mbar"},
        {"itemID":"76","itemName":"Flow peak","itemVal":"3277","itemUnit":"mL/s"},
        {"itemID":"7A","itemName":"Spontaneous minute volume","itemVal":"0.00","itemUnit":"L/min"},
        {"itemID":"7D","itemName":"Peak airway pressure","itemVal":"  0 ","itemUnit":"mbar"},
        {"itemID":"B5","itemName":"Spontaneous respiratory rate","itemVal":"  0 ","itemUnit":"1/min"},
        {"itemID":"B8","itemName":"Respiratory minute volume","itemVal":"0.01","itemUnit":"L/min"},
        {"itemID":"D6","itemName":"Respiratory rate","itemVal":" 60 ","itemUnit":"1/min"},
        {"itemID":"E7","itemName":"I:E I part","itemVal":" 1.0","itemUnit":" "},
        {"itemID":"E8","itemName":"I:E E part","itemVal":" 1.1","itemUnit":" "},
        {"itemID":"F0","itemName":"Insp. 02","itemVal":" 21 ","itemUnit":"%"}
    ]
}
```


图 8　基于行为树方法测试结果

Fig. 8　Test results based on behavior tree method

3.2　编程方法与行为树方法对比与分析

选择同一组测试数据分别基于 C 语言编程的方法与基于行为树方法进行协议解析测试,分别记录两种方法运行程序需要的时间,记录 100 组,并求出平均值与标准差,结果见表 3。

表 3　两种方法程序运行耗时

Tab. 3　Two methods of program running time consuming

方法	平均值/ms	标准差/ms
基于 C 程序	3.23	0.925 797
基于行为树	5.34	0.851 117

由表 3 可知,基于行为树方法程序耗时为 5.34 ms,基于 C 程序方法程序耗时为 3.23 ms。两种方法均需使用者对协议内容充分掌握,基于 C 程序的解析方法程序运行耗时短但对使用者的编程能力有较高要求,而基于行为树的解析方法仅需使用者根据协议内容对行为树编辑器节点进行组合,无需掌握较高编程能力,具有较好的普适性,此外基于行为树的解析方法还有以下优点：

(1)易用性强:低代码协议解析行为树编辑器隐藏了代码实现细节,每个行为节点以图形化方式展现；利用基于行为树方法进行协议解析,只需根据协议要求绘制协议流程图,技术门槛低；

(2)扩展性强、重用性高:行为树的行为逻辑和状态数据分离,可以反复利用；

(3)直观易于理解:行为树呈树状结构,解析流程直观易于理解。

(下转第 152 页)