第32卷 第13期 计 算 机 工 程 2006年7月
Vol.32 № 13 Computer Engineering July 2006
·软件技术与数据库· 文章编号：1000—3428(2006)13—0066—03 文献标识码：A 中图分类号：TP311.5

一个嵌入式软件构件的NFA量化度量模型

王志刚，王民北，骆雷飞
(国家数字交换系统工程技术研究中心，郑州 450002)

摘 要：CBSD(Component Based Software Development)在嵌入式开发领域正逐步得到应用，软件构件的非功能属性(Non-Functional Attribute, NFA)对开发成功与否至关重要。现有的模型专注于构件复用框架的建立，通过建立一些框架模型进行定性的描述，缺乏量化的评定。该文通过建立一个层次分析模型，对影响嵌入式软件构件的NFA的各种不可度量的模糊描述，细分为每一个都可以进行测量的子属性，根据考察构件所应用领域的侧重点不同，赋予不同属性不同的影响权系数，进而计算出一个描述NFA的数值，在实际工程中可以作为构件NFA的考察指标。

关键词：嵌入式构件；非功能属性；层次分析；度量

A NFA Quantity Measurement Model for Embedded Software Components

WANG Zhigang, WANG Minbei, LUO Leifei
(National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002)

【Abstract】CBSD is being used in embedded software component areas. Now the models focus on the establishment of component multiple frameworks. NFAs(Non Functional Attribute)are vital for the success of software systems. As to remedy the problems inherent in software development, a model has been developed to deal with NFAs. Analysing every sub-attribute that the NFA of a product depends on, the model's objective is to calculate the NFA-specific requirements. It considers the different extent that the sub-attributes to the NFA value by giving a weight value to each sub-attribute. The model will help to consider design tradeoffs, relate design decisions to NFRs, justify the decisions, and assist defect detection.

【Key words】Embedded software components; NFA; Hierarchical analysis; Measurement

1 概述

嵌入式应用的快速发展迫使人们在嵌入式软件开发过程中引入软件工程的思想，CBSD(Component Based Software Development)作为提高软件开发效率和质量的有效方法，在实践中已经得到了广泛的应用，但在嵌入式领域，还有许多技术问题需要解决。构件的非功能描述就是其中之一。

在嵌入式软件领域，关于CBSD的研究刚刚起步。国外提出的嵌入式组件模型有：比利时SEESCOA项目的CCOM模型，飞利浦公司用于消费电子嵌入式系统的Koala组件模型，ABB等公司参与的用于现场设备中嵌入式系统的PECOS组件模型。这些模型侧重于建立一种构件规范的形式化描述语言，对接口规格进行描述和功能正确性的形式化证明。缺少对组件非功能需求的描述和度量，因而在实际操作中具有很大的局限性。国内的研究则处于起步阶段。

本文通过建立一个层次分析模型，对嵌入式软件构件的NFA进行量化描述和度量，并给出量度标准。可以利用这一模型和度量标准，进行构件组装过程中的构件匹配和对嵌入式软件质量进行评估。

事实上，对软件的NFA至今没有一个一致的定义。下面列出几个关于软件NFA的经典表述。

(1)软件的功能需求定义了一个软件期望做什么，而非功能需求则指定了软件如何运行和功能如何展示的全局限制。

(2)软件的属性可用作描述及评价软件的一种方式。

(3)从最广泛接受的观点上来看，NFA主要包括：性能，可重用性，可维护性，安全性，可靠性等。由于所关注的复用领域的不同，不同领域的研究人员有着不同的理解，其发展源于软件复用技术在实践活动的深入。

由上面的NFA的定义可知：这些属性都是自然语言的描述，是模糊的概念。在设计过程中对构件NFA的分析往往是凭借开发者的经验，因而带有很大的主观性。对于嵌入式软件构件而言，其特点是：代码规模小，与硬件结合紧密，资源有约束（空间和时间有限），对平台依赖强，影响NFA的因素都可以进行细分量化。因此，我们使用层次分析的方式进行度量。

2 影响嵌入式构件NFA的因素

2.1 性能的影响因素分析

嵌入式构件的性能主要是指其各项工作时间、空间性能以及能耗。以互联网关键设备高性能路由器为例，“高性能”意味着较高的I/O带宽和吞度量。要求其调度算法的构件应该以提高程序响应速度为主要目标，空间资源的使用可以相对冗余。而PDA由于其价格和能耗的约束，其存储空间相对较小，对实时性要求不高，在设计过程中可以选取算法相对

基金项目：国家“863”计划基金资助项目(2003AA1Z1020)

作者简介：王志刚(1981—)，男，硕士，主研方向：系统移动通信终端，嵌入式系统；王民北，教授；骆雷飞，讲师

收稿日期：2005-11-07 **E-mail：**wzg@mail.ndsc.com.cn

复杂的构件对程序的存储空间进行巧妙的利用。

可以给出影响嵌入式构件的 3 个非功能因素：

(1)时间属性T_C：CPU 执行周期， I/O时间，通信延迟，算法执行时间

(2)空间属性S_C内存空间需求：

1)需求来源：程序的代码，数据；

2)持续：在组件执行结束后保留现场；

3)静态特性：在执行开始时开辟固定数量的内存空间；

4)动态特性：在执行过程中开辟大小可变的内存空间。

(3)能耗属性P_C：

能耗属性受时间和空间属性影响，在给定平台上，时间属性越高，即组件执行时间越短，能耗越小；空间需求越好，即组件需要开辟的内存空间越少，能耗越小。

需要说明的是，这 3 种属性并不单由构件本身决定。首先，构件执行的平台对这 3 个属性有至关重要的影响。比如，构件运行平台的 CPU 速度，存储器类型和操作系统对时间属性的影响非常大。其次，构件的输入对构件的性能也对这 3 个属性产生影响。还以“高性能路由器”为例子，高的数据输入速率时需要更多的处理时间和缓冲存储空间。这 3 个属性互相影响。如可以通过增加程序的内存空间达到提高执行速度的目的。

为了对这 3 个属性准确测量，需要考虑所有影响这 3 个属性的所有平台因素。如上所述，这些因素主要有：CPU 频率，存储器及其总线速度和类型，I/O 设备延迟和传输速率，嵌入式操作系统类型。

对于应用在同一平台的两个构件而言，其硬件环境是相同的，不再考虑平台因素，而应专注构件本身属性的分析。

2.2 可重用性影响因素分析

影响软件可复用性的因素主要有：

(1)构件复用的频率和效益

在实际开发过程中，构件的可重用价值随着复用频率的增加而增加。同时，复用一个组件所能带来的效益很大程度上取决于设计这个构件所花费的代价。比如在手机上实现一个中文输入法，完全可以复用已经设计好的输入法构件，而不必花费巨大的精力和金钱来重新建立庞大的汉字字库。构件的复用性随着其开发复杂度的增加而增加。复用次数和复用一个构件所节省的代价（时间和金钱）都可以在实践过程中得到具体的数值。

定义 F_C：构件的使用频率，M_C：使用构件与重新开发此构件相比节省的金钱，TS_C：使用构件与重新开发此构件相比节省的时间。

(2)构件的友好度

这是一个主观性的概念，不但指构件在以后开发过程中能够被利用的期望，更强调了它对于复用的人的友好程度。它随着构件的标准化和可理解程度的增加而增加。复用这个构件所需获取的额外相关领域知识越少，它的友好度就越高，其潜在复用价值越大，预期复用此构件的开发者也就越多。

对于友好度的评定，可以让有丰富开发经验的技术人员进行分等级打分，以 1 为基准，选取有效数字为 1 的小数，如 0.5（基于后面的归一化考虑）进行评定。

(3)构件的适配性

构件的适配性主要是指其与上下文环境和构件结合的难易程度。复用设计的目标是：为了使这个构件执行其功能属性，对这个构件进行的使其匹配上下文的修改越少越好。在理想情况下，当复用一个构件时，不需要对它进行任何修改即可使用。

由于构件内部信息被隐藏，与外界交流的渠道只有其接口，因此只能通过研究构件的接口方法评估其适配性。根据嵌入式系统的特点，提出了构件中方法的输入参数和返回值的度量标准。

首先进行定性地分析：对于一个构件，如果其方法不需要任何输入参数和返回值，就认为这个构件的适配性最好，因为不与外部有任何数据依赖；方法中没有输入参数而只有返回值的构件，其适配性与前者相比就稍差一点；方法中既有输入参数又有返回值，其构件适配性与前面二者相比最差。

定义 m_i为构件中方法（函数）i 操作数（输入、输出参数，下同）的复杂度。

$$m_i = c_r \times m_r + c_s \times m_s \ ;$$

m_r为返回值的复杂度；m_s为输入参数的复杂度；c_r和c_s为常数；

当没有返回值时，m_r为 0，有返回值时为 1。$m_s = \sum_{i=1}^{k} p_i$，其中：p_i表示每个输入参数的复杂度；k 为参数的个数。当参数为值类型时，$p_i = 1$；当参数为引用或者指针类型时，$p_i = r(r > 1)$。由此可以定义嵌入式构件的适配性 A_C：

$$A_C = \sum_{i=1}^{n}(c_r \times m_r + c_s \times \sum_{i=1}^{k} p_i)。$$

2.3 可维护性影响因素分析

嵌入式构件的可维护性是指：找到并修改构件中一个错误所需要的代价程度。 在设计阶段，发现错误最主要的方法还是进行构件的测试，编写尽可能多的覆盖构件功能的测试脚本，找到错误的代价就是进行测试所花费的时间和金钱。修改错误所花费的代价同样是时间和金钱。M_{MC}:测试构件并发现其中错误所花费的金钱，T_{MC}: 测试构件并发现其中错误所花费的的时间。

2.4 可靠性影响因素分析

度量其可靠性的一个有效方法就是计算其两次平均故障之间的平均时间 ATBF(Average Time Between Faults)。

$$ATBF = ATTF + ATTR$$

其中：*ATTF* 为平均故障时间(Average Time of Trouble Find)；*ATTR* 为平均修复时间(Average Time of Trouble Repair)。

在此使用时间来考虑系统的可靠性，而没有使用构件中错误数来进行度量，是因为在实际应用过程中，用户更加关心的是故障，而不是错误。一个很明显的例子，如果一个构件中有很多错误，但系统使用过程中一直使用其正确的部分，这些错误可能已经保持了 50 年、100 年。而另外一些错误 2 年之内被发现。对于这样的故障率，它们对于软件的可靠性的影响也是可以忽略不计的。

2.5 安全性影响因素分析

安全性是一个架构体系，嵌入式系统每一个层级都有其具体的安全性设计考虑和实现方案。比如，对于一个通信终端而言。顶层协议，底层硬件，中间模块耦合都有其特殊的安全性考虑。对于嵌入式软件构件而言，其安全性主要是指：

(1)可用性

对于嵌入式系统软件构件而言，安全的前提是首先要保证其能够正常工作，即可用性，在给定的时间和环境内，其正常运行的概率。

可用性＝$\frac{ATTF}{ATTF+ATTR}\times 100\%$，如果在(0,t)内，系统没有故障，时间t就是可用性，它是时间的函数。

(2)机密性

也称为信息隐藏性，信息隐藏一方面降低了构件理解的难度，另一方面减少了修改其内部信息可能带来的错误。因此构件应该尽可能多地隐藏其内部信息。

对于这个特性，可以通过分析其输入输出参数中指针的个数来判定。这是因为，指针参数增加了程序越界访问的可能，降低了构件内部信息的隐蔽性。使用指针所带来的危险概率是远大于值类型参数。因此，以指针参数的个数为量度进行安全性评估是合理的。

p_i:构件中方法（函数）I 输入输出参数中指针的个数，P_C:构件中指针的个数，$P_C=\sum_{i=1}^{k}p_i$ ，其中 k 为构件中方法的个数。

2.6 各影响因素的归一化

为了能够比较各个因素对相应属性的影响程度，需要有一个统一的评价标准，在获取详细的量化数值的基础上，采用归一化的方法进行统一：可以以实际工程中的一些历史数据、项目预算和时间估计为参考值，由经验丰富的开发人员将各个属性的测量值与之进行度量，以 1 为基准，各功能属性选取(0，1)中的一个有效数字为 1 位小数(0.1,0.2,…,0.9)进行评价，影响因素越大，其选用的小数越大，这样就可以把每个因素对此性质的的影响程度分为 10 个等级。

3 基于层次分析的评价模型

我们使用层次分析法[5]来分析各个因素对系统的NFA的影响，在不同的领域中，对系统的NFA要求的重点也不一样。比如，航天系统侧重对系统可靠性的要求，而金融系统更侧重于安全性。因此，针对不同的领域，给出的 5 大属性对NFA质量的影响权重也应该随着其领域侧重点不同作相应调整，如图 1 所示。有些属性在一些应用领域甚至可以不予考虑。

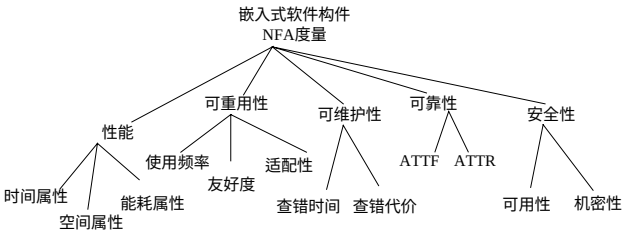

图 1 嵌入式软件构件 NFA 度量层次模型

分别用 W_p 代表性能属性对 NFA 的影响权重，W_r 代表可重用性对 NFA 的影响权重，W_m 代表可维护性对 NFA 的影响权重，W_d 代表可靠性对 NFA 的影响权重，W_s 代表安全性对 NFA 的影响权重。

由此，可以给出评估嵌入式构件的公式：

$$NFA=W_p*\sum_{i=1}^{3}p_i+W_r*\sum_{i=1}^{3}r_i+W_m*\sum_{i=1}^{2}m_i+W_d*\sum_{i=1}^{2}d_i+W_s*\sum_{i=1}^{2}s_i$$

其中：p_i为影响性能的 3 个因素的归一化系数，r_i为影响可用性的 3 个因素的归一化系数，m_i为影响可维护性的 3 个因素的归一化系数，d_i为影响可靠性的 3 个因素的归一化系数，s_i为影响安全性的 3 个因素的归一化系数。

在不同的嵌入式应用领域，这些权重的值是不同的，如在金融系统中，安全性 W_s 的权重值就相对大一些；在网络关键设备上（如路由器），V_{A_C} 性能属性 W_p 的权重值就大一些。

在实际应用过程中，需要根据构件应用领域的不同作出相应的调整。还以高性能路由设备为例，作为网络上的关键设备，对系统的性能和安全性要求较高，而且设备开始运行后，不会允许进行频繁的维护，因此对可维护性和可重用性要求较小。可以对图中的层次模型进行修正，得到图 2。

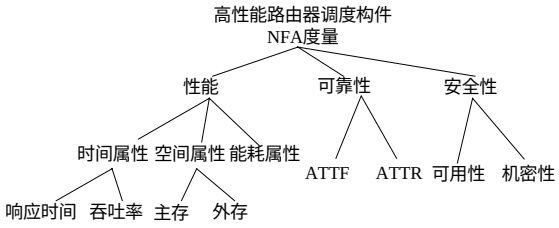

图 2 高性能路由调度构件 NFA 度量模型

在此基础上，给出一个嵌入式构件的复用匹配框架，如图 3 所示。

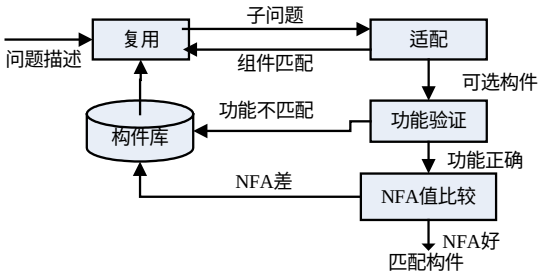

图 3 嵌入式复用构件选取框架

框架中描述了从构件库中选取构件的过程：

(1)将可以利用构件库中构件的问题分解为子问题；

(2)对可以使用构件的子问题选择可以使用的构件；

(3)对这些构件进行功能的验证；

(4)在功能验证通过的构件中选取 NFA 值最好的为匹配构件。

4 结论

在许多实际系统的开发中，开发人员经常忽视软件的非功能属性，灾难由此发生。软件结构需要重新变化，大量代码需要重新编写，需要更多的时间、人力和物力。基于层次的构件 NFA 属性度量模型，解决了目前无法对嵌入式软件构件的非功能属性进行度量的问题，在实际项目中，其潜在应用价值巨大。

参考文献

1 Rosa N S, Justo GRR, Cunha PRF. Framework for Building Non-functional Software Architectures[C]. Proceedings of the 2001 ACM Symposium on Applied Computing. ACM Press, 2001.

2 Franch X, Botella P. Putting Non-functional Requirements into Software Archietecture[C]. Proceedings of the 9th International Workshop Software Specification and Design. IEEE, 1998: 60-67.

3 Mylopoulos J, Chung L, Nixon B A. Representing and Using Nonfunctional Requirements: A Process-oriented Approach[J]. Transaction on Software Engineering, 1992, 18(6): 483-497.

4 Chung L, Nixon B A. Dealing with Non-functional Require- ments: Three Experimental Studies of a Process-oriented Approach[C]. Proceedings of the 17th International Conference on Software Engineering, 1995-08.

5 姜启源. 数学模型(第 2 版)[M]. 北京: 高等教育出版社, 1992.