嵌入式软件平台的构件化模型研究*)

古幼鹏 桑 楠 熊光泽
(电子科技大学计算机科学与工程学院实时系统研究室 成都 610054)

摘 要 为了提高嵌入式软件的生产率,本文提出了一种基于构件的嵌入式软件平台模型CBMESP。CBMESP将软件开发平台与运行平台以统一的构件模型进行构件化,使其可以应用于各种嵌入式领域而不必更改该模型,只需调整构件库中的具体构件即可,具有普遍适用性。因此,CBMESP不但加强同一领域内,也加强了领域之间的重用性。CBMESP强调并提供了开发平台与运行平台(应用软件)统一的基于构件的定制方式,更好满足了嵌入式软件开发的多样性要求;最后,CBMESP根据嵌入式软件特点提出构件模型由三个可以独立实现和运行的部分组成,并解决了各部分之间信息的传递问题,较好适应了嵌入式软件的交叉开发过程和嵌入式系统资源有限的特点。

关键词 构件,软件重用,嵌入式软件,软件平台

Research on Component-Based Model of Embedded Software Platform

GU You-Peng SANG Nan XIONG Guang-Ze
(School of Computer Science and Engineering, UEST of China, Chengdu 610054)

Abstract To promote embedded software development, a component-based model of embedded software platform, which is called CBMESP, is presented in this paper. CBMESP can apply to a variety of embedded domains by replacing components in its component database; and software developers are able to customize their software platform by selecting components and setting parameters of components. In addition, the component model of CBMESP consists of three parts that can be implemented and executed independently, so it can save memory resource of embedded system and be suitable to the cross development process of the embedded software. At the end of this paper, an example demonstrates these features of CBMESP.

Keywords Component, Software reuse, Embedded software, Software platform

1 引言

嵌入式领域具有以应用为中心的特点,因此,其软件开发大部分采用手工作坊的方式进行,同一领域内部和领域之间存在着大量重复开发,这既影响了软件开发效率,也降低了软件的开发质量。随着嵌入式系统的迅速发展,快速开发出适合市场需要的高质量软件已经成为一个嵌入式产品能否成功的重要条件。为此,面向领域的软件平台被提了出来,如WinRiver公司的面向消费电子的软件开发平台[1],欧洲的面向汽车电子领域的OSEK/VDK[2],高通公司的Brew平台[3]等。通过在同一领域内重用开发方法、开发工具和程序代码,这些面向领域的软件平台对提高各自领域的软件开发效率和质量都取得了一定成效。但是,这些开发平台不能适应多种应用领域的要求,因为它们都是针对具体领域特点提出的,将领域特性和公共特性(即领域间共有的特性)紧密结合在一起,要想把同样的软件平台从一种领域转向另一种领域,需要做相当大的工作。这样就阻碍了领域之间的重用性,另外,嵌入式系统的应用领域是多种多样,不可能每个领域都研制一个对应的软件平台。总之,面向领域的软件开发平台对提高软件生产率仍有较大不足。

目前基于构件的软件开发(Component-Based Software Development, CBSD)[4]引起了研究者的广泛关注,它通过组装可重用的软构件来生成新的软件。CBSD通过重用构件来提高软件开发效率,通过使用经过多次重用证明是可靠的构件来提高软件质量,利用构件接口与实现相分离的机制从而可灵活替换的特点来提高软件的适应性[5],因此将CBSD引入嵌入式软件平台中能够很好满足嵌入式领域的多样性特点以及对开发效率和软件质量的要求。在嵌入式软件平台中采用CBSD已经有一些相关的研究成果,但这些研究都只关注嵌入式软件平台的一部分,例如嵌入式实时CORBA[6]关注运行平台的构件化技术(即嵌入式软件本身的构件化),基于软总线的CASE环境[7]关注嵌入式软件开发平台的构件化技术。事实上嵌入式领域的多样性使嵌入式软件及其开发平台同时具有多样性,二者密不可分,因此一个统一的构件化的嵌入式软件平台(包括运行平台和开发平台)是嵌入式领域的必然要求,但是统一的构件化的嵌入式软件平台的相关研究工作还未发现,因此本文提出了一种统一的嵌入式软件平台的构件化模型CBMESP,它具有1)通用性,通过更换构件库适用于各个领域;2)以统一的方式对运行平台和开发平台进行构件化,并统一进行定制管理;3)适应嵌入式系统资源有限的特点和嵌入式软件的交叉开发过程。本文第2节详细介绍CBMESP模型,第3节简要介绍基于CBMESP模型的嵌入式软件平台实例,最后总结全文。

2 嵌入式软件平台的构件化模型(CBMESP)

CBMESP由开发平台和运行平台两部分组成,每个平台

*)863基金资助(基金号:2003AA1Z2210)。古幼鹏 博士研究生,主要研究方向:软件设计方法学,基于构件的软件重用,实时CASE与软件工程。

又由构件和构件容器组成，构件容器是构件运行的环境。组成开发平台的构件称为工具构件，其构件容器叫做工具总线；组成运行平台的构件称为运行构件，其构件容器叫做运行管理器。所有的构件都存放在构件库中统一管理。

2.1 开发平台模型(CBDPM)

CBDPM采用基于工具总线的构件化体系结构，每个工具都被构件化成工具构件，工具构件都只与工具总线连接。工具向外提供服务或请求服务，工具间通过工具总线以请求——响应的方式交互。如图1所示，工具构件集合包括构件库管理器CDM、系统配置管理器SCM、项目管理器SPM和其它可选的扩展工具构件，它们提供用户需要的开发/管理功能；工具总线(Tool Bus，TB)负责工具间的互操作(消息传递和数据交换)以及工具的"即插即用"管理。CBDPM主要优点有1)工具间只能通过工具总线交互，工具间的相互依赖小；2)用户可以根据需要进行工具的更换、增减和升级；3)工具总线TB屏蔽了部分底层支持环境(包括操作系统和网络)的差异，工具构件不用考虑这些底层支持环境的细节，其移植性和重用性将大大增强。

CBDPM采用"基本工具构件集合＋面向领域的扩展工具构件集合"的定制模式，其中基本工具构件集合(即CDM、SCM、SPM)是任何领域的软件开发平台都必须要有的，并且其作用也是相同的。CDM对构件库(Component database)中的所有工具构件和运行构件执行数据库管理操作，例如加入或删除构件到构件库、查询库中某构件的相关信息等；SCM通过从构件库中选择可选的扩展工具构件并配置这些工具构件来创建一个面向特定领域的嵌入式软件开发平台，通过从构件库中选择运行构件并配置这些运行构件从而创建一个面向特定领域的嵌入式软件应用程序；SPM是CBMESP与软件开发者之间的接口，响应开发者的命令，管理整个软件开发过程及其开发过程中的相关信息，例如创建一个开发项目，记录本项目所用的工具构件集合和运行构件集合及其配置参数，根据用户的命令启动某个工具构件等。

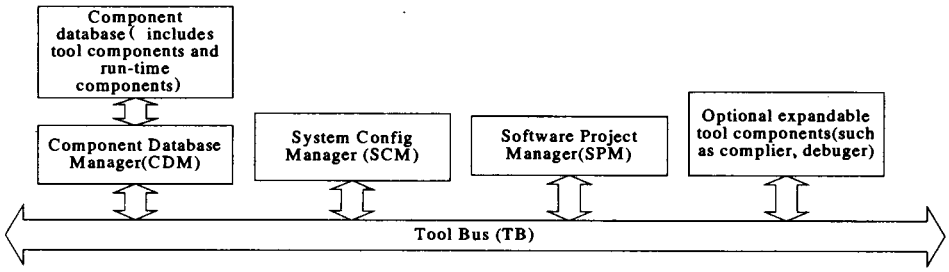

图1 开发平台体系结构图

2.2 运行平台模型(CBRTPM)

CBRTPM如图2所示，它由运行构件和运行管理器(Run-Time Administrator，RTA)组成。运行构件包括从构件库中选择并配置了参数的运行构件(即重用的构件)和构件库中不存在而需要新开发的运行构件，这些新运行构件可以加入到构件库中以备以后重用。RTA负责加载、卸载、激活运行构件，传递运行构件之间的消息(例如方法调用)。由于RTA可以屏蔽系统底层的差异，构件与底层系统之间(垂直方向)是松耦合的关系；又由于运行构件之间通过RTA间接交互，运行构件间(水平方向)也是一种松耦合的关系，因而运行构件非常独立，重用性很高。另外，系统本身的扩展和用户为特定应用进行的扩展都使用一致的构件替换方式进行，整个系统能够达到更高的扩展性和灵活性。

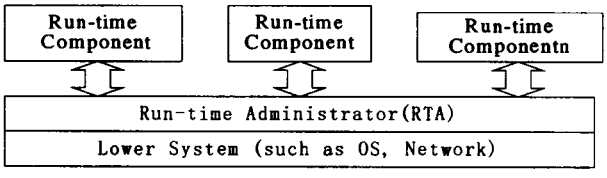

图2 运行平台体系结构图

2.3 构件模型

构件可以看成是接口的集合，确定构件模型的关键就是确定构件应该具有的接口，而构件接口拟定与其使用环境有很大联系。嵌入式系统具有两个特性：1)运行构件的配置过程与运行过程是相互分离的；2)嵌入式系统资源有限。本文定义的构件模型如图3所示(采用伪IDL[8]描述)：每个构件必须包括IComInfo、IComConfig和IComRun这三个接口，每个接口可以采用独立的方式实现。其中IComInfo供CDM使用，它提供构件的描述信息，主要包括构件名段(描述构件的名称、版本、类型等)，运行体的内部结构描述段(主要描述运行体代码是由哪几个独立的代码文件组成，存放位置，哪些是组成构件运行体必须要有的，哪些是可选的)；IComConfig供SCM使用，以完成对本构件的运行参数的配置；IComRun供构件容器(对工具构件来说是TB，对运行构件来说是RTA)和其它构件在运行时使用。另外在图3中省略的接口表示构件提供的具体功能，随构件的具体种类而定，IComRun接口和省略的接口称为构件的运行体。因此，该构件模型可以分成独立的三部分：IComInfo接口、IComConfig接口和以IComRun接口为代表的运行体，这三部分各自可以有不同的实现方式，并可以独立运行。

```
CM={IComInfo,IComConfig,IComRun, ...};
interface IComInfo{
  Cstring Name;
  Cstring Type;
  Cstring Version;
  Cstring locationOfCode;
   ......
}
interface IComConfig{
 bool ConfigPara(void);
 bool GetConfigInfo([out] void *pInfo);
}
interface IComRun{
  void LookupInterface([in] unsigned32 InterfaceId,
                       [out] void **Iref);
  void DuplicateObject(void);
  void ReleaseObject(void);
  void ComponentInitilize([in] void *buffer);
  void Run(void);
}
```

图3 构件模型定义

在一般的构件模型中，构件所有的接口都是在一起实现的，本构件模型分成了三个可以独立实现和运行的部分，它特别适合于嵌入式系统：1）运行构件通常在宿主机上进行参数配置，而在目标机上运行。宿主机通常是 PC 平台，采用 C++ 编程；目标机是嵌入式平台，采用 C 或汇编编写，所以就可以把 IComInfo 、IComConfig 用 C++ 实现，而 IComRun 等运行体部分用 C 语言实现；2）嵌入式系统一般资源紧张，按照本模型，对于运行构件在目标机上执行时只需要调入 IComRun 等运行体部分执行，而不用调入与之无关的 IComInfo 、IComConfig 等部分。

2.4 CBMESP 模型的工作过程

在介绍了 CBMESP 模型中的基本元素后，本小节将具体阐述其工作过程，以便更好理解 CBMESP 模型以及各个部分的关系。在此，我们以一个嵌入式软件开发过程为例子。利用 CBMESP 模型开发嵌入式软件，可以分成三个步骤。

（1）定制软件的开发平台。当启动基于 CBMESP 模型的软件平台时，TB 先被启动，它自动加载基本工具构件集合（即 CDM、SCM、SPM），这样一个基本的开发平台就形成了。然后软件开发人员使用 SCM 提供的开发平台定制功能，SCM 将通过 TB 向 CDM 发送“获取构件库中的工具构件的信息”的消息，CDM 也通过 TB 将相关信息返回给 SCM。SCM 接收到这些信息后，就可以将构件库中有关的工具构件信息显示在 GUI 人机界面上。软件开发人员首先可以选择适合其软件项目的开发工具构件；然后可以对其选择的工具构件进行参数配置（例如为编译器配置其运行时要使用的编译选项参数），其过程如下：当 SCM 捕获开发人员想配置工具构件时，它首先通过 CDM 从构件库中获取对应工具构件的“IComConfig”接口的实现代码；然后调用该接口中的“ConfigPara(void)”方法执行，该方法就出现一个人机界面（例如弹出窗口）供开发人员对该构件进行配置（包括对工具构件的运行体中的可选部分进行选择以及对工具构件运行时需要的参数进行配置）；完成配置后，SCM 调用接口中的另一方法“GetConfigInfo([out] void * pInfo)”得到配置结果；最后，SCM 将开发平台的定制结果（即选择的工具构件和每个工具构件的配置结果）通过 TB 传递给 SPM，SPM 将这些信息作为项目开发信息保存起来。

（2）运行平台（应用软件）的定制。选择应用程序中要使用的运行构件以及对选择的运行构件进行配置称为运行平台的定制。该过程与步骤（1）完全一样，唯一有区别的是选择的构件类型是运行构件，而不是工具构件。

（3）构件的运行。这里的构件运行是指构件的运行体在构件容器中运行。对于工具构件来说，其在宿主机上运行，为开发人员提供工具服务（例如编译程序），它的构件容器是 TB；对于运行构件来说，其在目标机上运行，它们的运行就是开发人员开发的软件的运行过程，它的构件容器是 RTA。尽管二者的运行环境和构件容器不同，但是它们的过程都是一样的。首先，构件容器加载以“IComRun”接口为代表的运行体部分的实现代码（注：图 2 中的运行构件其实只应该包括运行构件的运行体部分，但这并不影响我们对 CBRTPM 的理解）；其次，构件容器先通过 SPM 得到其保存的配置结果信息，然后调用“IComRun”接口中的“ComponentInitilize([in] void* pBuffer)”方法将在步骤（1）或（2）中配置的运行时工作参数传递给相应的构件供其使用（例如编译器就能获得在步骤（1）中为它配置的编译选项参数）；最后，构件容器调用“IComRun”接口中的“Run(void)”方法启动构件运行。构件在运行过程中，构件间的交互就是一个构件需要另一个构件提供的服务（即接口），可以相互调用“IComRun”接口中的“LookupInterface([in]unsigned32 InterfaceId,[out] void** pIref)”、“DuplicateObject()”、“ReleaseObject()”等方法来获得需要的接口指针，控制相应构件的生命期，其过程类似 COM[5]，具体过程限于篇幅，在此不作详细介绍。

另外，在软件开发平台和运行平台定制过程中如果没有找到合适的构件，说明该构件不存在，就需要按照 2.3 小节的构件模型开发该构件，然后将该构件通过 CDM 加入到构件库中，以备以后重用。

3 实例分析

在“面向 PDA 的嵌入式软件平台”开发项目中，为了使该软件平台也能够方便地用在其它领域，我们按照 CBMESP 模型来搭建该软件平台。我们开发了在 Windows 平台上的 TB、CDM、SCM、SPM 以及 GCC 编译器和基于 Intel386 的 GDB 调试器等工具构件，同时开发了基于 Linux 的 RTA、WAP 协议栈、MPEG4 解码器、USB 客户端、嵌入式数据库等运行构件。然后利用该软件平台以及构件库中的构件，按照 2.4 小节的步骤，我们快速地开发出了一个 PDA 软件，该 PDA 软件能够通过无线上网浏览、并可以通过 USB 与 PC 机交换数据。后来另外一个 PDA 产品的软件在该 PDA 产品功能基础上，需要扩充播放网上视频，通过该平台从构件库中多选取了 MPEG4 解码器构件，很快就开发出了新软件。

在“面向 PDA 的嵌入式软件平台”开发项目完工不久，我们又接到了一个航空电子嵌入式产品的开发任务，该产品采用 C 语言，具有 USB 接口、需要嵌入式数据库，我们发现“面向 PDA 的嵌入式软件平台”中有相应的构件，决定就使用“面向 PDA 的嵌入式软件平台”，按照 CBMESP 的构件模型，开发目前构件库中没有的构件（例如基于 Motorola 处理器的调试器及与航空电子特定相关的一些功能构件）并将其加入到构件库中，然后利用该平台快速组装出了产品软件。

通过构造基于 CBMESP 模型的软件平台和利用该开发平台开发软件的实践表明，CBMESP 模型将嵌入式软件平台面向领域的多样性用构件化封装技术来解决，并且将开发平台和运行平台以统一的方式构件化，不但同一领域内，而且不同领域之间都能共享同一软件架构、支持环境以及各个具体的构件（工具构件与运行构件），最大限度地提高了软件重用性，因而提高了软件开发速度和开发质量。从上面例子可以看出，只需要调整构件库中的构件，“面向 PDA 的嵌入式软件平台”就变成了“面向航空电子的嵌入式软件平台”，而整个系统不作任何改变，从而在 PDA 领域和航空电子领域之间共享整个模型架构、基于该模型的开发过程与方法以及构件库中的某些具体构件（GCC 编译器、USB 客户端、嵌入式数据库等构件），提高了构件库中的构件在领域之间的重用性。

总结 因为嵌入式软件以应用为中心的特点，使嵌入式软件开发中的重复性工作很多。面向领域的开发平台虽然提高了对同一领域内的软件开发的重用性，但是不同领域间的转换以及领域之间的重用比较困难。基于构件的软件开发（CBSD）可以较好克服多样性问题，提高重用性，但是目前的 CBSD 在嵌入式领域要么关注软件开发平台的构件化，要么关注运行平台的构件化。实际上嵌入式软件的开发环境和运

（下转第 248 页）

第一，寻找适合于软件维护领域的灵活的数据预处理技术，从数据源中获得尽可能多的信息，并以标准的形式提供给数据挖掘算法。灵活的数据预处理技术可以根据不同的应用（如程序理解、程序修改等）处理不同形式（如程序文档、程序源代码等）、不同粒度（如程序源代码中的文件、函数、语句等）的原始数据。

第二，寻找适合于软件维护领域不同应用的高效挖掘算法，进一步加强数据挖掘在软件维护领域的应用。高效的挖掘算法是指可以根据不同的需要快速、准确地挖掘出有用的信息。

第三，制定统一有效的评估标准，进一步加强对挖掘结果的评估。

第四，构建适用于软件维护领域的、支持多种数据挖掘功能同时每一种功能又支持多种方法的自动化数据挖掘工具，加大数据挖掘技术在软件维护领域的实际应用。

参 考 文 献

1 陈世鸿，朱福喜，等．软件工程原理及应用．武汉：武汉大学出版社，2000. 243～252

2 Oca C M D，Carver D L. Identification of Data Cohesive Subsystems Using Data Ming Techniques. In：Proc. of the Intl. Conf. Software Maintenance，1998. 16～23

3 Chen Y-F，Nishimoto M Y，Ramamoorty C V. The C Information Abstraction System. IEEE Transaction on Software Engineering，1990，16(3)：325～334

4 Grass J E. Object-Oriented Design Archaeology with CIA＋＋. Computing Systems：The Journal of the USENIX Association，1992，5(1)：5～67

5 Narat V. Using a relational database for software maintenance：a case study. In：Proc. of the IEEE Conf. on Software Maintenance，1993. 244～251

6 Grubb P，Takang A A．Software Maintenance：Concepts and Practice. Second Edition，Hackensack：World Scientific Publishing，2000

7 李莹，张琴燕．程序理解．计算机应用研究，2001(6)：40～43

8 李必信，郑国梁，李宣东，张勇翔，梁佳．软件理解研究与进展．计算机研究与发展，1999，36(8)：897～906

9 Mancoridis S，et al. Using Automatic Clustering to Produce High-Level System Organizations of Source Code. In：Proc. of the 6th Intl. Workshop Program Understanding，1998. 45～53

10 Tjortjis C，Layzell P J. Using Data Mining to Assess Software Reliability. In：Proc. of the 12th Intl. Symposium Software Reliability Engineering，2001. 221～223

11 Tjortjis C，Sinos L，Layzell P. Facilitating Program Comprehension by Mining Associated Rules from Source Code. In：Proc. of the 11th Intl. Workshop Program Comprehension，2003. 125～132

12 Kanellopoulos Y，Tjortjis C. Data Mining Source Code to Facilitate Program Comprehension：Experiments on Clustering Data Retrieved from C＋＋ Programs. In：Proc. of the 12th Intl. Workshop Program Comprehension，2004. 214～223

13 Shirabad J S，Lethbridge T C，Matwin S. Supporting Maintenance of Legacy Software with Data Mining Techniques. In：Proc. of the 17th IEEE Intl. Conf. on Software Maintenance，2001. 22～31

14 Ying A T T，Murphy G C，Ng R，Chu-Carroll M C. Predicting Source ode Changes by Mining Change History. IEEE Transaction on Software Engineering，2004，9：574～586

15 Zimmermann T，Weigerber P，Diehl S，Zeller A. Mining Version Histories to Guide Software Changes. In：Proc. of the 26th Intl. Conf. on Software Engineering，2004

16 Shirabad J S，Lethbridge T C，Matwin S. Mining the Maintenance History of a Legacy Software System. In：Proc. of the Intl. Conf. on Software Maintenance，2003

17 Michail A. Data Mining Library Reuse Patterns using Generalized Association Rules. In：Proc. of the 22nd Intl. Conf. on Software Engineering，2000

（上接第 218 页）

行环境都具有多样性，所以其开发平台和运行平台应该同时构件化并要有机联系起来，才能最适合嵌入式软件的特点。为此，本文提出了 CBMESP 模型，它同时关注了开发平台与运行平台的构件化，并将二者的构件模型统一起来，从而使其可以应用于各种嵌入式领域而不必更改该模型，只需调整构件库中的具体构件即可，具有普遍适用性。这样，CBMESP 不但加强同一领域内，也加强了领域之间的重用性（包括共享整个模型架构、基于该模型的开发过程与方法以及构件库中的具体构件）；CBMESP 强调并提供了开发平台与运行平台（应用软件）统一的基于构件的定制方式，更好满足了嵌入式软件开发的多样性要求；CBMESP 根据嵌入式软件特点提出构件模型由三个可以独立实现和运行的部分组成，并解决了各部分之间信息的传递问题，较好适应了嵌入式软件的交叉开发过程和嵌入式系统资源有限的特点。在文章最后，我们用实际例子说明了 CBMESP 模型的特点。

下一步的研究工作包括不断地充实我们的构件库以及开发基于其它平台的构件容器，使构件容器具有分布式处理能力，以便使本模型提出的规范能够应用到更为广泛的领域，以便提高相应领域的嵌入式软件开发效率。

参 考 文 献

1 WindRiver corporation. http://www.windriver.com/products/platforms/consumer_devices

2 OSEK Group：OSEK/VDX Network Management-Concept and Application Programming Interface. Version 2.51，2000

3 QUALCOMM Incorporated：http://brew.qualcomm.com/brew/en

4 杨芙清，王千祥，梅宏，陈兆良．基于复用的软件生产技术．中国科学(E辑)，2001，31(4)：363～371

5 Rogerson D. Inside COM. Microsoft Press，1997

6 Schmidt D C，Kuhns F. An Overview of Real Time CORBA Specification. IEEE Computer，2000，33(4)：56～63

7 Verral M S，Morgan L. Tool Integration in CASE Environments：the Software Bus. Computer-Aided Software Engineering，Fifth International Workshop，1992．46～49

8 Henning M，Vinoski S. C++ Based CORBA Advanced Programming. 北京：清华大学出版社，2000