# 基于 CCM 的软件构件组装技术的研究*)

许 峰 陈智强 黄 皓 王志坚
(南京大学计算机软件新技术国家重点实验室 南京 210093)
(南京大学计算机科学与技术系 南京 210093)

**摘 要** OMG 在推出的 CORBA 3.0 规范中提出了一个构件模型(CCM)。CORBA 构件不仅定义了对外提供的功能,而且还定义了构件所需要的外部功能,使得构件可以利用接口进行组装。同时 CCM 借鉴了 EJB、COM 等构件模型的优点,并充分利用了 CORBA 的开放性。本文在对 CORBA 构件模型以及模型组装框架中的相关技术进行研究的基础上,努力探索基于 CCM 的构件组装在系统开发中的应用。
**关键词** CBSD,CORBA 构件模型,构件组装

## Research of CCM-Based Software Component Composition Technology

XU Feng CHEN Zhi-Qiang HUANG Hao WANG Zhi-Jian
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093)
(Dep. of Computer Science and Technology, Nanjing University, Nanjing 210093)

**Abstract** The OMG released the CORBA 3.0 in August 2002. A component model is contained in CORBA 3.0, CORBA Component Model (CCM). In this model, a CORBA component defines not only the services it provides but also the services it required. So CORBA components can be assembled via their ports. At the same time, CCM inherits the advantages of other existing component models such as COM, EJB, and it takes full advantages of CORBA. Based on the research of CORBA Component Model and the related technology in the CCM assembly framework, this paper exploresthe application of CCM-Based system development.
**Keywords** CBSD,CORBA component model,Component assembly

## 1 引言

随着构件技术的日益成熟,基于构件的软件开发方法(Component-Based Software Development)被广泛接受,逐步成为解决软件危机,提高软件生产率和确保软件质量的软件开发范例。为了构建企业级应用系统,开发者需要在一个包括事务处理、持久性、事件和命名服务的分布式结构中集成商业逻辑,需要有一个灵活的配置模型。设计、实现这样的应用系统是相当复杂的。虽然 CORBA 的灵活机制给了开发者无数的选择,但仍需要大量的细节规定,无法快速有效地使用。

为了解决这一难题,OMG 在 CORBA3.0 规范中提出了 CORBA 构件模型 CCM(CORBA Component Model)。CCM 标准化了利用 CORBA 作为中间件的构件的开发。OMG 采用 CORBA 构件模型(CORBA Component Model,CCM)[1,2] 来扩展和包容 CORBA 对象模型。CCM 标准不仅大大增强了服务器软件的可重用性,而且为 CORBA 应用的动态配置提供了更大的灵活性。

构件组装技术是基于构件的软件开发的核心技术。目前关于构件组装的理论研究成果有很多,但是整体上还停留在概念阐述与体系建模的探索性阶段。而 OMG 推出的 CORBA3.0 中包含了一个构件模型,它充分利用已规范化和产品化的技术与平台,在面向对象技术的基础上,对对象进行扩充,将构件技术与面向对象技术融合。本文对 CORBA 构件模型以及构件组装技术展开研究,分析了基于 CCM 的构件组装框架,在此基础上讨论了框架中组装器的设计和实现,并给出了一个应用实例。

## 2 CORBA 构件模型

CORBA 构件是基于 CCM 的应用系统中的基本构造块。一个构件由一个构件引用表示,在实现中实际上是由一个对象引用来对外代表这个构件引用的。构件定义实际上就只是对原有的接口(interface)定义的一种扩展。作为类型系统的抽象,构件类型可以被实例化用来创建具体的带有状态和标示的实例。构件类型封装了它的内部表示和实现。

构件提供了多种表面特征(surface features)用于支持用户与构件以及构件间的互连,这些表面特征被称为构件的端口(ports)。构件的 CORBA 构件模型中一共支持四种基本的端口。(1)刻面(Facets):是构件提供的相互独立的命名接口,用于构件与其用户交互。构件通过刻面方式向外提供的接口是相互独立的,它们与构件通过继承支持的接口没有关系。一个构件类型能够以刻面的形式给其用户提供多个相互独立的接口。刻面是构件在运行时把它所提供的功能展示给其用户的一个主要手段。一个构件可以没有刻面也可以拥有多个刻面。(2)接插口(receptacles):是一个抽象的概念,这个抽象的概念具体表现在构件上为一组用来建立和管理连接的一组操作。简单来说,接插口定义了描述构件使用外部对象引用的接口点(一般用于分布构件之间的互连或组装)。构件接插口提供了构件间互相连接的模型描述,用于描述构件与其它被连接对象(构件)接口之间的关系。构件接插口分为单一接插口(Simplex receptacles)和多元接插口(Multiplex receptacles)。(3)事件源(event sources)和事件槽(event sinks):事件槽是 CCM 中的命名连接点,用于接收外界向其

*)本课题得到国家自然科学基金(60473091)和国家"八六三"高技术研究发展计划项目基金(2003AA142010)资助。许 峰 博士研究生,主要研究方向为软件构件和分布式计算。陈智强 教授,博士生导师,主要研究领域为软件自动化。谢 立 教授,博士生导师,主要研究领域为分布式计算。

推入的特定类型事件。事件槽体现了构件接受某种特定类型事件的能力，事件槽就其本质而言是一个提供特殊接口类型（事件消费者类型）的刻面，该刻面用于推送某种类型的事件。(4)属性(Attributes)：是实现构件配置的一个主要方法。构件通过访问器(accessor)和增变器(mutators)操作向用户展示的命名值。构件属性是构件向外界显露的可被访问或可被定制的内容，主要用于定制构件。

## 3 基于 CCM 的构件组装

### 3.1 基于端口的构件组装

CORBA 构件模型使用端口(ports)来代表构件对外提供的功能和其要求的外部功能。这些端口定义了构件所有对外交互的信息，构件在实现时不是直接使用其他构件提供的功能，而是必须通过它定义的端口来访问外部功能。构件之间的连接是在所要求的功能和所提供的功能之间进行匹配，因此，通过接口就可以定义系统中构件之间的所有连接。通过这种方式就把传统的构件集成方式中构件之间的固定连接方式变成灵活的连接方式，降低了构件之间的依赖性，提高了构件的独立性和可复用性。

### 3.2 两种交互模式

从体系结构的角度来看，构件主要采用四种基本的方法来与其它构件进行交互：向其它构件提供接口、从其它构件那里获得接口、产生事件和观测事件。CORBA 构件模型为构件定义了四种端口来实现这四种方法：刻面用来向其它构件提供接口、接插口用来从其它构件那里获得接口、事件源用来产生事件、事件槽用来观测事件。同时 CORBA 构件模型还为 CORBA 构件定义了一个旨在提高构件柔性的端口——属性。所谓构件的柔性(flexibility of component)是指构件在不同构件环境中的行为结果的一致性[3]。这些端口为构件间的组装提供了两种交互模式：同步模式和异步模式。

### 3.3 CCM 中的用户角色

在 CCM 中我们根据用户在构造应用程序中所承担的不同任务分为以下 5 种角色：(1)构件设计者：其作用是根据系统需要设计构件，使用 OMG IDL3.0 和构件实现定义语言 CIDL 来定义构件和其实现信息；(2)构件实现者：构件实现者负责实现构件中的商业逻辑操作，这些操作由 OMG IDL 3 经过编译器产生的本地 IDL2 接口映射所定义。构件实现者可以继承由 CIDL 编译器产生的骨架来实现构件。在实现文件中，实现者还可以实现本地的容器回调接口并且调用本地容器接口。(3)构件包裹者：其职责是把构件实现者实现的二进制构件以及构件描述器、属性描述器等描述信息包裹成为单个构件软件包。一般这项工作是使用一个可视化的工具来完成的。(4)构件组装者：负责把多个构件包根据需要组装成一个构件集，构件集中包含有多个可定制的构件包以及该构件集的 XML 描述器(包括构件实例和它们的互连信息以及构件集的逻辑组成)。产生的构件集依然可以拿来和其他构件包或构件集一共组装成为更大的构件集。(5)构件部署者：部署者根据构件集的描述符文件把构件集(或构件)安装在网络上指定的节点上、激活构件实例并连接构件。

### 3.4 框架中的三个层次

基于 CCM 的构件组装框架应该从三个层次来考虑系统构造问题，如图 1 所示。

这三个视角是体系结构层次，构件层次和分布对象基础设施层次。

体系结构层定义了系统中的构件、它们的功能以及它们之间的相互连接关系。构件层关心的是构件的具体实现、组装实现以及部署。分布对象基础设施层则关注在构件之间如何进行通信。从构件组装的角度来看，组装的关键在于基于 CCM 的组装器，该组装器必须把单独实现的各个构件有机地组装成为一个完整的系统或一个更大的构件。

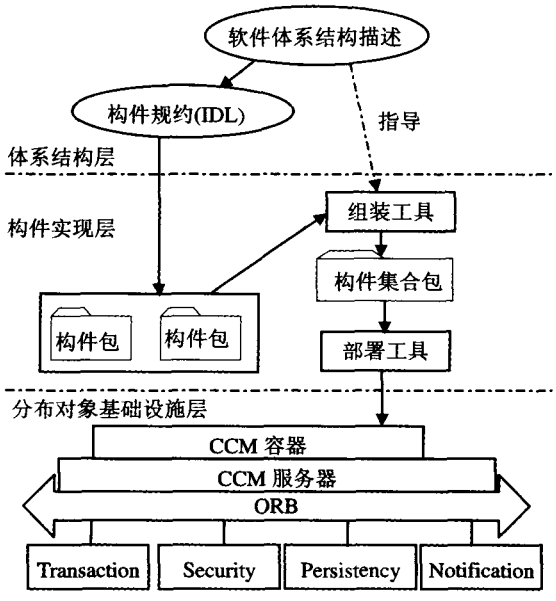



图 1 基于 CCM 的构件组装框架

## 4 基于 CCM 的组装器的设计与实现

### 4.1 组装器的设计

组装工具应完成的以下三个主要任务：(1)载入所需构件包，读取相关描述信息并把用于构件连接的端口信息以图形化的方式显示给用户。(2)用户结合系统需求，根据体系结构描述与组装器进行交互。使用组装器提供的描述方法描述构件间的连接关系和构件实例的创建信息。(3)最后组装器根据与用户的交互信息生成一个构件集描述器，并把相关文件压缩成为一个构件集文件(.aar)作为组装器的输出。

### 4.2 构件组装的描述

复用构件强调构件的集成，只有构件按某种形式集成到构架中，才能发挥它们功能上的互补性。为了能够让系统组装者能够使用组装工具把构件组装成为系统，除了接口定义以外，在整体上我们还需要一种方式来表示构件的端口信息和构件集中包含的构件及这些构件之间的组装关系。这方面，XML 文档的内容模型提供了一种良好的集成机制。通过 XML 文档用户可以一目了然地知道单个构件的对外交互的端口信息或实现复合构件需要哪些构件的组装。

CCM 一共定义了四个基于 XML 语法的描述器来对构件或构件集进行描述。同时 CCM 为这四个描述器定义了四个 XML DTD。它们分别是 CORBA 构件包描述器(softpkg.dtd)、CORBA 构件描述器(corbacomponent.dtd)，构件集描述器(componentassembly.dtd)和属性描述器(properties.dtd)。CORBA 构件模型中的描述器均采用 W3C 的开放软件描述语言(Open Software Description，OSD)来描述。OSD 是一个基于 XML 词法的描述器语言。CCM 规范在不丧失 OSD 普适性的情况下轻微地扩充了 OSD 语言用于支持 CORBA 构件的描述。

### 4.3 组装器的实现

我们选择微软的 VB6 和 MSXML 解析器来实现我们的组装工具。MSXML 解析器读取一个 XML 文档，然后将其内容解析到一个抽象的节点容器中，这些节点代表了文档的结构和内容。(1)组装工具首先载入所有 CORBA 软件包，并读取在该软件包的顶级目录 meta-inf 下的软件包描述器文件(*.csd)。(2)根据软件包描述器文件中的〈descriptor type="CORBA Component"〉元素的地址信息找到该 CORBA 构

件的构件描述器(*.ccd)。(3)读取 CORBA 构件描述器中的〈componentrepid〉元素的 repid 属性可得到该主构件接口在接口仓库中的标示符。(4)组装器拿主构件的接口标示符与描述符中的每一个〈componentfeatures〉元素的 repid 属性进行比较,如果两者相符则证明已找到主构件的描述元素。(5)读取〈componentfeatures〉元素中的〈ports〉子元素,获得构件用于对外连接的端口信息的描述,并把这些信息以图形化的方式向用户表示。(6)用户根据系统的体系结构描述和构件的端口信息把构件连接起来,并给定创建构件实例信息。(7)组装工具根据与用户交互情况,生成一个构件集描述器和一个包括该系统中所有的构件文件的构件集压缩文件(.aar)。

## 5 原型系统

### 5.1 原型系统设计

汛情监视系统为防汛值班人员提供实时汛情信息服务。汛情监视系统应提供对河道测站、水库测站、闸坝测站、雨量测站、潮位测站以及控制测站等的实时监视,各类测站的实时、历史水文信息以及工情信息、考证数据等应以简洁、直观和形象的方式标示出来,并且系统应自动采用不同颜色标识测站超警、超保、超汛限等状态。同时,系统还提供了对水情信息的自动监视,险工险段、洪峰、主要测站等专项监视,以关注特定测站或观测点的实时变化情况。我们从汛情监视系统中抽取了水位报警子系统作为我们的应用实例。下面,我们将以水位报警子系统中的构件设计及集成来讲述基于 CCM 的构件组装的具体应用。

实验原型系统包括以下三个构件:(1)水位监视构件定义了两个端口:一个是发布 WarningStationInfo 事件类型(超警戒水位的测站详细信息)的"事件源"。另一个是连接 AccessDB 接口(访问数据库的接口)的"接插口"。(2)数据访问构件定义了一个提供 AccessDB 接口(访问数据库的接口)的"刻面"。(3)用户构件定义两个端口:一个是接受 WarningStationInfo 事件类型(超警戒水位的测站详细信息)的"事件槽",另一个是连接 AccessDB 接口(访问数据库的接口)的"接插口"。

### 5.2 原型系统实现

构件 IDL 定义:我们使用 OMG IDL3 文件 WrService.idl 定义以上三个构件:

```
Import Components;
Module WrService {
    Interface AccessDB //访问数据库接口
    { Void ExecuteSQL (in String strsql)//执行传入的 SQL 语句
    }
    eventtype WarningStationInfo //超过警戒水位的闸坝站实时数
        据信息
    { string stcd(); //站码
      string stnm();//站名
      string time(); //超警时间
      string wrz(); //警戒水位
      string zu(); //闸上水位
      string zd(); //闸下水位
    };
    Component Monitor//水位监视构件
    { Uses AccessDB adb;//声明该构件要求使用 AccessDB 类型的
      接口
      Publishes WarningStationInfo info;//声明构件发布 Warning-
      StationInfo 类型的事件
    }
    home MonitorHome manages Monitor {};//定义管理水位监视
        构件的宿主
    Component DBService //数据库访问构件
    { Provide AccessDB adb;//声明构件提供一个 AccessDB 类型
      的接口
    }
    home DBServiceHome manages DBService {};//定义管理数据
        库访问构件的宿主
        Component User //用户构件
        { Consumer WarningStationInfo info;//声明构件接收
          WarningStationInfo 类型的事件
          Uses AccessDB adb;//声明构件使用 AccessDB 类型的
          接口
        }
        home UserHome manages User {};//定义管理用户构件
        的宿主
};
```

构件实现:以上 IDL3 文件经过 IDL3 编译器产生了与 IDL3 文件对应的本地(Local)接口,实现构件也就是在实现文件中实现这些本地接口。下面以数据访问构件为例介绍构件实现方法:

接口 AccessDB 的本地接口映射:

```
local interface CCM_ AccessDB: AccessDB {Void Exe-
    cuteSQL (in String strsql)
}
```

在 AccessDBImpl.java 中实现 CCM_ AccessDB 接口:

```
public class AccessDBImpl extends org.omg.CORBA.
    LocalObject
    implements CCM_ AccessDB {
        Void ExecuteSQL (in String strsql) {
        ……//方法实现
        }
}
```

构件 DBService 映射到下面的本地接口:

```
local interface CCM_ DBService :
                ::Components::EnterpriseComponent
                {
    CCM_ AccessDB get_adb();
}
```

在 DBServiceImpl.java 中实现 CCM_ DBService 接口:

```
public class DBServiceImpl extends AccessDBImpl
    implements CCM_ DBService{
        public CCM_ AccessDB get_adb(){
            ……//方法实现
        }
}
```

构件组装:我们通过输入水位监视构件软件包、用户构件软件包和数据访问软件包,利用组装工具按照系统设计时的方案把这三者组装起来。完成这些操作后,组装工具将输出一个扩展名为".aar"的构件集文件。这个构件集文件中包含的构件集描述器描述了组装信息。下面我们以水位监视构件为例,简单介绍在组装过程中输入的软件包描述器、构件描述器和输出的构件集描述器的实现要点。

水位监视构件的软件包描述器:我们用 author 这个元素描述了构件的作者的详细信息,idl 元素则用来指定 idl 接口文件,descriptor 元素指定了该软件包中的构件描述器,implementation 元素则详细说明了软件包中的实现文件的相关信息(包括实现所依赖的操作系统、编程语言、执行文件名字以及对其它类库的依赖性)。

水位监视构件的构件描述器:在这个描述器中使用 componentrepid 元素和 homerepid 元素分别定义构件和构件宿主的类型,componentfeatures 元素用来定义构件支持的端口的详细信息。

系统的构件集描述器:我们使用组装工具把用户构件的接受 WarningStationInfo 类型事件的"事件槽"与监视构件的发布相同类型事件的"事件源"连接在一起,并且把监视构件的接受 AccessDB 接口的"接插口"与数据访问构件的提供该接口的"刻面"连接在一起。构件组装工具根据这些输入生成了一个扩展名为".aar"的构件集文件。在这个构件集中包含一个构件集描述器。构件集描述器主要包括三大部分:componentfiles 元素列出了在构件集中的用到的所有构件,partitioning 元素的描述创建构件和构件宿主的实例以及它们的部署信息,而 connections 元素则描述已部署的构件及其宿主之间的连接信息。

**结束语** 本文通过对 CORBA 构件模型和软件构件组装技术的研究,剖析了基于 CCM 的构件组装框架,并详细分析了框架中构件间交互机制及构件描述机制;使用 VB6 和微软

的 MSXML 解析器(XML 的 DOM 解析器)实现了框架中的组装工具。在此基础上给出了一个具体的应用原型系统。

## 参 考 文 献


1 CORBA Specification Version 3.0, formal/02-06-33, July 2002. http://www.omg.org/cgi-bin/doc? formal/02-06-33
2 OMG CCM Implementers, Group, MARS PTC & Telecom DTC, OMG Meeting, Orlando, USA, CORBA Component Model Tutorial, June 2002
3 艾萍. 构件柔性组装描述的形式化方法研究及其在水利领域的应用:[博士学位论文]. 2002.12
4 杨芙清. 软件复用及其相关技术. 计算机世界 C 版,1999(3)
5 McIlroy M D. Mass Produced Software Components. In: P. Naur and B. Randell, eds. Software Engineering. NATO Science Committee, January 1969
6 马亮,孙艳春. 软件构件概念的变迁. 计算机科学,2002,29(4):28~30
7 张士琨,王立福,杨芙清. 基于 COTS 构件的系统开发. 计算机科学,2000,27(1)
8 Brocjschmidt K. Inside OLE2. Microsoft Press,1994
9 Gamma E, Helm R, Johnson R, Vlissides J 著. 李英军,等译. 设计模式. 机械工业出版社
10 Navarro A, White C, Burman L. XML 从入门到精通. 电子工业出版社
11 Henning M, Vinosni S. 基于 C++的 CORBA 高级编程. 清华大学出版社


(上接第 228 页)

**引理 6.2** 系统(1)和系统(3)具有相同的解。

证明:假设 $\Phi$ 是(1)的解,则由引理 6.1 知,它满足(2.1)且 $\Phi_t=-F$。由(1.4)知,(2.2)成立。由(1.3)知,(2.3)成立。故(1)的解满足(2)。

假设 $\Phi$ 是(2)的解,令 $F=-\Phi_t$,则 $\Phi_t+F|\nabla\Phi|=0$,即满足(1.1)。又 $\nabla F\cdot\nabla\Phi=\nabla(-\Phi_t)\cdot\nabla\Phi=-\frac{d}{dt}|\nabla\Phi|^2=0$,即满足(1.2)。由(2.2)及(2.3)知,(1.3)和(1.4)成立。故(2)的解满足(1)。

**定理 6.3** 在一维的情况下,系统(2)存在惟一的解 $\Phi(x,t)\in R,(x,t)\in R\times R$。

证明:由 $\frac{\partial\Phi}{\partial x}=1$ 知 $\frac{\partial(\Phi-x)}{\partial x}=0$,故 $\Phi(x,t)-x=H(t)$。下面我们确定 $H(t)$。

首先,对于 $\Phi(x,t)=0$,有 $x=-H(t)$ 和 $\frac{\partial\Phi}{\partial t}=H'(t)=f(x)=f(-H(t))$。由(3.3)知 $\Phi(x,t)|_{t=0}=x-x_0$,故 $H(0)=-x_0$,原问题变成常微分方程的初值问题 $\begin{cases}H'(t)=f(-H(t))\\H(0)=-x_0\end{cases}$,由文[8]知该常微分方程存在惟一解 $H(t)$,从而得到 $\Phi(x,t)=x+H(t)$ 为(3)的惟一解。

在二维情况下,令 $\vec{x}=(x,y)^T\in R^2$,则 $dist((x,y),S_0)$ 是 $f_0(x,y)$ 的重写。由于零水平集未知,可能有一些奇异的性质,需要将它转化成正则化系统,应用窄带法和隐函数理论[8],可证在二微空间中也是成立的,同时可扩展到三维或更高维空间。

## 7 实验结果及分析

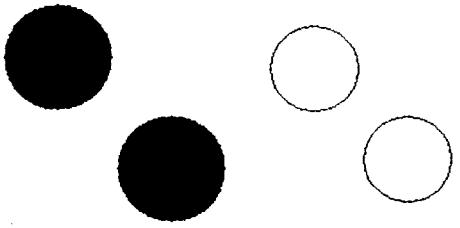
表 1

| 图像 | 像素大小 | 迭代步长 | 迭代次数 | 耗费时间 |
|---|---|---|---|---|
| (a)Circle | 128×128 | 0.5 | 2 | 4.59 |
| (b)Dragon | 128×128 | 4 | 4 | 8.84 |
| (c)Star | 134×139×3 | 40 | 2 | 8.75 |
| (d)Leaf | 174×192×3 | 20 | 6 | 39.43 |

本文是在 pentium(R) CPU 2.60GHz,256 MB 内存,Windows XP 的环境下,应用 MATLAB 6.5 做实验。如图 1 所示,其中(a)(b)均为二值图,(c)为带噪星形图,(d)为带噪树叶图,均取得了良好的效果。在实验中,我们统一取 $\alpha=1$,$\beta=0.01$,$\gamma=1$,对于某些图像,调整这些参数可以得到更好的效果。同时也可以看出随着 $\tau$ 值的变大,抗噪能力越强。

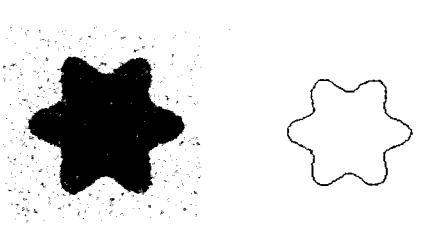
## 参 考 文 献


1 Kass M, Witkin A, Terzopoulos D. Snakes: Active contour models. International Journal of Computer Vision, 1998. 321~331
2 李培华,张田文. 主动轮廓线模型(蛇模型)综述[J]. 软件学报,2000,11(6): 751~757
3 Malladi R, Sethian J A, Vemuri B C. Shape modeling with front propagation: A level set approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995,17(2):158~174
4 Kimmel R. Fast Edge Integration, Geometric Level Set Methods in Imaging, Vision and Graphics. In: S. Osher and N. Paragios, eds. Springer Verlag, 2003
5 Russon M, Deriche R. A Variational Framework for Active and Adaptative Segmentation of Vector Valued Images: [Technical Report 4515]. INRIA, France, 2002
6 Gomes J, Faugeras O. Level sets and distance functions. In: Proc. 6th Europe Conference on Computer Vision, Dublin, 2000
7 Sethian J A. Level set methods and Fast Matching Methods. Cambridge, U.K.: Cambridge Univ. Press,1999
8 Wang Dejun, et al. Level Set Methods, Distance Function and Image Segmentation. In: Proceeding of the 17th International Conference on Pattern Recognition, 2004

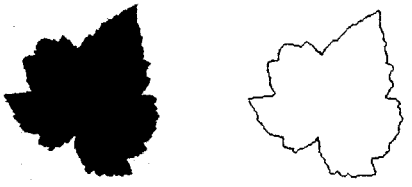



(a) Circle

(b) Dragon



(c) Star



(d) Leaf

图 1 (a)(b)(c)(d)原图(左)及其对应目标轮廓线(右)