Fuxi:一种普适计算的语言及平台

王忠斌

（深圳市海创达资讯技术有限公司计算新技术实验室，深圳，518048）

摘　要：普适计算，聚焦于用户任务本身，而不是完成其计算的设备和技术，为我们展示了一种新型的计算前景。如何将不同的计算技术综合到一起，建立一个统一的编程环境，并使用户程序不因系统环境的复杂而增加新的复杂度，将成为实施普适计算的关键。我们根据普适计算的特点，采用面向方面的软件技术，将系统分解为机制和策略两个正交的方面。开发了一种平台无关的、面向对象的说明性语言——Fuxi 语言，用于在策略面进行计算任务的描述；而 Fuxi 平台则从机制面为用户任务的执行提供底层支持。

关键词：普适计算；面向方面；程序设计语言；说明型语言；Fuxi 语言；中间件

1．引言

普适计算[1, 2]聚焦于用户任务本身，而不是完成这些计算的设备和技术，为我们展示了一种新的计算前景。通常，普适计算所涉及的计算设备具有异构性，从可穿带的设备到台式电脑；通信手段具有多样性，包括红外通信、蓝牙[3]、GPRS[4]等；系统环境具有复杂性，从嵌入式的 RTOS 到通用的桌面操作系统。如何将这些不同的技术综合到一起，建立统一的软件架构[5, 6, 7]，并使得用户程序不因系统环境的复杂而增加新的复杂度，已成为实施普适计算的关键。

我们根据普适计算的上述特点，采用面向方面[8, 9, 10]的软件技术，提出了一种普适计算的软件架构。该架构的基本思维是，将系统分解为机制和策略[9]两个正交的方面(Aspects)，在机制面采用高效的过程性程序设计，对普适计算所涉及的技术环境进行封装；在策略面采用具有高表达能力(expressive)的说明性语言编程，描述用户任务；最后由该说明性语言的解释器或抽象机，在用户不可见的情况下，完成机制与策略的编织(Weave)，最终完成用户任务，获得计算的结果。

我们根据上述普适计算的软件架构，设计并实现了一种平台无关的、面向对象的、函数型与逻辑型结合的说明性程序设计语言——Fuxi 语言；Fuxi 语言不涉及任何计算的技术成分(如线程、同步、网络访问、对象迁移等)，而是聚焦计算任务本身的描述。由 Fuxi 平台(包括 Fuxi 抽象机、包体系)，提供对 Fuxi 程序的底层(infrastructure)支持。从用户程序的角度，实现了技术的不可见性(invisibility of technology)。

2．Fuxi 语言简介

Fuxi(伏羲)程序设计语言是一种平台无关的、面向对象的、逻辑型与函数型结合的说

联系作者：王忠斌，E-mail：victor@fuxi.org

明型语言，简单地说，Fuxi 是说明型的 Java。Fuxi 所采用的文法成分都是流行的、为程序员所熟悉的，因此可以让程序员在很短的时间内就可以熟练地使用 Fuxi 语言编写应用程序。

Fuxi 语言的设计哲学是，我们并不试图去发明一项新的语言技术，而是将过去散落在各种语言中优秀的成分，结合计算技术的新发展，经过精心的设计，整合到一个语言中来；以提高语言的建模能力、减小语言同现实世界之间的“语义鸿沟”，提高语言的编程效率，降低程序设计的难度。

2.1 Fuxi 的文法

Fuxi 文法[11]可以概括为以下公式：

Fuxi 文法 = 类似 Java 的类型框架[12] + 模式匹配[13]的方法构成

以下是一个计算 Fibonacci 数的 Fuxi 程序：

```
import fuxi.*
public active class FibonacciApp: Applet
{
    Fib(0) = 1
    Fib(1) = 1
    Fib( int n ) = Fib( n – 1 ) + Fib( n – 2 )
    public OnError(ERROR_MEMORY_ALLOC) -> Console.Println(“内存不足！”)
    public Activate() = let{ int n = Console.Readln().ToInteger() } in
    {
        Console.Println( “请输入一个整数:” )
        Console.Println( “Fib(“ + n + “)=” + Fib(n) )
    }
}
```

图 1. 一个简单的 Fuxi 程序

图 1 是一个典型的 Fuxi 控制台应用程序，它计算 Fibonacci 数。从这个例子中我们可以看出：

1. Fuxi 的文法和 C++、JAVA 或 C# 极其相近；
2. Fuxi 采用模式匹配的方式定义方法，如 Fib 函数，包含三个模式 Fib(0)、Fib(1)和 Fib(n)；
3. Fuxi 的方法区分函数、子句和触发器，函数的模式和定义体之间采用‘=’连接，而子句采用‘<-’连接，触发器采用‘->’连接；
4. Fuxi 支持惰性计算，即表达式只有在用到时才计算值。局部变量 n 在建立时并没有被计算，而只是在被引用时才进行计算。

2.1.1 Fuxi 程序的基本结构

Fuxi 程序由一个或多个源程序文件组成。程序定义类型，而类型又包含其成员。这些类型通常又可以组织在一个模块中。类和接口是类型的例子，成员包括字段(Field)和方法(Method)。在 Fuxi 中，方法又可以分为函数(Function)、谓词(Predicate)和触发器(Trigger)等。每个方法都是一个由具有相同模式签名的规则(Rule)组成的结构，该签名也就是方法的签名。

一般而言，Fuxi 程序应该包括以下部分：

1) 导入部分：导入 Fuxi 的基本类库或其它的装配件，如本例中的 Fuxi；
2) 类定义部分：类是 Fuxi 的编程单位，是程序的构成要素。一个 Fuxi 程序至少需要定义一个类；
3) 输出部分：至少包含一个公开的类定义，如本例中的 FibonacciApp。

2.1.2 Fuxi 程序的执行入口

首先，一个值得注意的地方是上述程序没有象 JAVA、C#等那样定义一个执行入口函数 main，数据的输入和输出出现在函数 Activate()中；

其次，在类 FibonacciApp 定义的前面使用了 `active` 关键字。

Fuxi 语言区分主动式对象和被动式对象，只有主动式对象才具有独立的执行线程，而被动式对象必须包含在某个主动式对象中。类定义前的 active 关键字可使类带有主动式风格，带主动式风格的类的所有实例都是主动式对象(不带 active 风格的类也可以定义主动式对象)。

2.2 正交的对象风格化

正交的对象风格化(Orthogonal Stylization of Objects)是对象的正交持续性(orthogonal persistence)[14]的扩展，指的是对象风格和对象类型正交。对象风格指对象在执行过程中表现的姿态，例如，是否拥有自己独立的线程、是否具有持续性、是否可以在网上移动等；而对象类型指对象的结构和行为，对象类型通常是类。我们可以把对象的风格想象成产品的颜

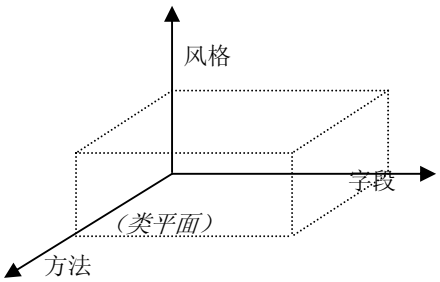

图 2. 实例的三维空间

```
class A { … }
active A a_act // 主动式对象 a_act
mobile A a_mob // 可移动对象 a_mob
persistent A a_per // 持续性对象
                        a_per
remote A a_rem // 远程对象 a_rem
```

图 3. 对象风格的使用

色，产品的设计师通常关注产品结构本身的设计，而忽略对颜色的考虑。然而，当产品成型后，工人可以为产品着上各种不同的颜色。

例如，我们定义了类 A 后，可以定义几种不同风格的对象，如图 3 所示。这四个对象都具有相同的类型，具有相同的调用接口。在程序设计过程中，程序员可以完全不必对这三个姿态不同的对象进行特殊的考虑。因为，对象风格只是在执行时才表现出来。

Fuxi 语言目前支持的风格有 active, mobile, persistent, remote，同时 Fuxi 也提供对用户

自定义风格的接口。Fuxi 在语言级对风格提供支持，可以让程序员只关注类结构本身的设计，关注问题的本身，而忽略对同问题方案正交的某些技术侧面的考虑。其中，mobile 和 remote 风格使得 Fuxi 能够在语言级直接对普适计算予以支持，使其成为一种普适计算的语言。

3. Fuxi 的实现技术

语言的实现，首先需要建立执行模型。然后，根据所建立的执行模型，开发语言的翻译系统（编译器或解释器），并且为该翻译系统所转换的目标代码提供必要的运行环境。从理论上说，每个语言都可以有多种执行模型，可以采用多种不同的实现方案。但在实践上，过程性语言通常使用随机存取模型(RAM)，而说明性语言采用项重写模型(TRM)[13, 15, 16]。

3.1 Fuxi 的执行模型与抽象机

我们选择了 TRM 的一种，图计算模型(GCM)[15]，作为 Fuxi 的执行模型。Fuxi 编译器将 Fuxi 源代码编译成 Fuxi 对象图(FOG)，Fuxi 抽象机在 Fuxi 基本库的支持下，对 FOG 进行图变换，获取最终结果。

Fuxi 平台是 Fuxi 程序执行的最小环境，它包括 Fuxi 抽象机和基本包。Fuxi 抽象机为一种类似于 WAM[16]的图归约计算器。

3.2 Fuxi 的对象模型

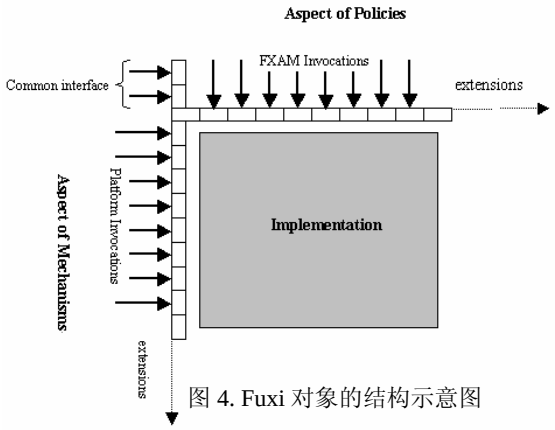

图 4. Fuxi 对象的结构示意图

每个 Fuxi 对象都是从一个 GOBJ (Generic Object)接口派生出来的。Fuxi 的基本库为我们定义了这个对象接口。Fuxi 对象通常需要两个虚拟方法表，分别表示机制和策略[17]两个正交的侧面。其中机制 VTable 是每个对象必须具有的，GOBJ 中的唯一字段就是指向这个 VTable 的指针。机制 VTable 的入口指向由本地机器代码实现的方法。并不是每个对象都包含策略 VTable，它是供 Fuxi 抽象机调用的方法列表。策略 VTable 入口所指的方法既可以是本地机器代码实现的方法，也可以是 Fuxi 计算图中的某个子图。全部策略均为本地实现的对象称为本地对象(Native Object)。Fuxi 平台的基本包均为本地包。

机制 VTable 和策略 VTable 所指的内容存在着很大的不同，前者是由本地机器指令组成的可执行的代码段(过程)，而后者则是由 GMethod(Generic Method)接口派生出来的对象。根据实现机制的不同，这些 GMethod 对象可以是：FuxiMethod、NativeMethod 及 RemoteMethod 等，分别表示由 FOG 实现的方法、本地指令实现的方法和远程方法。通过 GOBJ 中的反射函数 GetMethod，可以获取策略 VTable 中的这些 GMethod 对象。

3.3 包体系(Package Hierarchy)

Fuxi 的包体系是以 Fuxi 的基本库(Fuxi fundamental library)为基础，包括 util、math、

fxawt、shom 等。在 Fuxi 的包体系中，包分为两种类型：语言包和方面包(Aspectual Package)。其中，语言包是供 Fuxi 程序调用的的包，如 SHOM (表单对象模型，一种说明型的 GUI)；而方面包则是 Fuxi 平台对普适计算所涉及的技术方面的支持，供其它 Fuxi 本地类或抽象机调用；而 Fuxi 应用程序不能直接调用方面包(技术的不可见性)。

Fuxi 的包体系如下图所示：

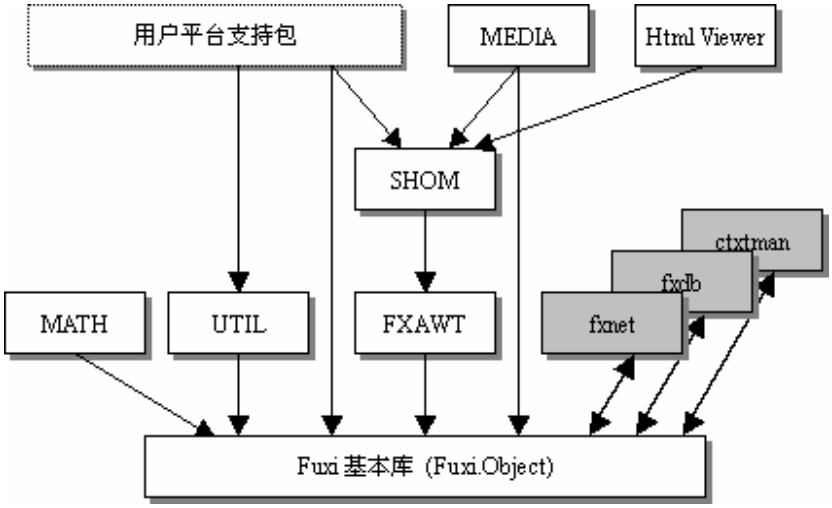

图 5. Fuxi 的包体系

其中，fxnet 提供网络和远程访问支持，ctxtman 为环境(context)管理包，而 fxdb 提供 Fuxi 对象的持续性支持，即对象数据库支持。这三个包通常不对用户开放，由系统自动调用。它们也是 Fuxi 对象的远程、持续性及环境感知(context-awareness)等方面的支持包。

所谓“用户平台支持包”(user-platform support package)，是指我们或用户自己利用 Fuxi 系统开发工具包(Fuxi System Development Toolkit)[16]，开发的用户特定(domain-specific)平台相关的包。我们向用户提供 Fuxi SDK，是 Fuxi 平台开放性的重要表现。

3.4 机制与策略的编织

机制的执行通常是被动的，程序的执行线程在策略面上。当抽象机在执行到 Fuxi 程序的某个特定的位置(称为 Join point)，便自动触发机制面中的某个函数。这些触发点主要是由对象的实现方法决定的(风格控制着实现)。例如，对跨越线程边界的对象的访问，系统将会自动启动消息传递机制。

机制中的函数则通过 Fuxi 语言的反射(reflection)功能，访问 Fuxi 程序的对象。GOBJ 中定义了一组反射函数，其中 GetAttribute 可以用来获取 Fuxi 对象中的属性，而 GetMethod 则用来获取对象中的方法。

4. Fuxi 的应用模式

在 Fuxi 语言的实现过程中，我们已经开展了 Fuxi 的应用研究，探索 Fuxi 的普适计算的应用模式。

4.1 设备端应用模式

图 6 是 Fuxi 平台设备端的一个典型应用模式。其中，GSM-Wrapper 是利用 Fuxi SDK 开发的 GSM 封装类，将原来 GSM Stack 同应用层之间的消息发送转换为 GSM 类中的方法，供 Fuxi 应用模块调用；其“平台支持包”主要是键盘(Keypad)和显示的驱动，实现 fxawt 所定义的用户界面的机制函数，及 Fuxi 与 OS 之间的交互，如事件触发、系统设置等。

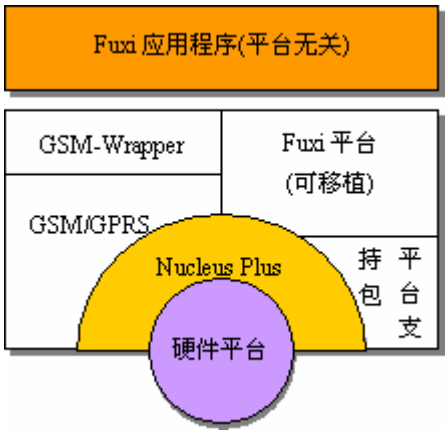

图 6. 一个典型的 Fuxi 应用的系统架构

这一应用模式可以概括为以下公式：

嵌入式 OS + Fuxi 平台 + Fuxi 的平台相关的扩展 + Fuxi 应用模块

该模式具有以下特点：

1) 软件开发容易。由于 Fuxi 是说明性语言，语言的表达能力强；同时 Fuxi 应用程序可以借助于 Fuxi 的集成化开发环境来完成。
2) 可移植性强，能满足用户产品线开发要求。当 OS 或硬件发生变化时，由于用户自己开发的 Fuxi 模块是与平台无关的，而 Fuxi 平台是可移植，因此用户只需对其领域相关的包进行修改即可。
3) 便于最终用户的个性化应用。由于 Fuxi 语言是一种易学易用的说明性语言，也便于最终用户使用。同时，Fuxi 编译的结果为 Fuxi 模块文件，本质上和图象、文档一样地属于数据文件。可以方便地输入到嵌入式设备里，替换掉原来的应用模块即可。

4.2 组网模式

设备之间的网络协同、计算任务的迁移及分布是普适计算的基础。Fuxi 平台为我们提供了一个便捷的组网模式。系统将网络分为若干个部门(Department)，每个部门包括一个服务器(运行 FuxiServer，称为 Fuxi Site)和若干个客户端(设备或电脑)；部门内部为 C/S 网络结构，而部门之间为 P2P 组网模式(如图 7 所示)。

部门中的设备分为私有设备和借过设备(passing-by)。未经注册的借过设备，不能访问部门中资源，但可以借用部门的网络同宿主部门通信。

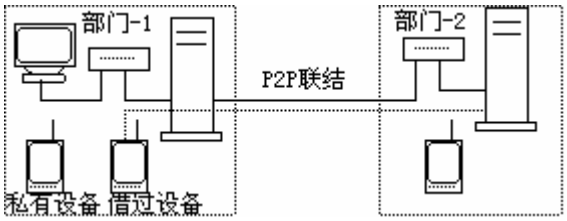

图 7. Fuxi 平台的组网模式

5. 结论

我们根据普适计算的基本要求，采用了面向方面的软件技术，设计并实现了 Fuxi 语言，用于普适计算的用户程序设计；而 Fuxi 平台则为 Fuxi 程序的执行提供底层支持。

目前，Fuxi 语言已经在 WinCE、Nucleus Plus 及 Linux 的 ARM 嵌入式环境和 Windows、Linux 的桌面环境中实现。平台的通信能力已经在 LAN 上通过仿真测试，现正进行蓝牙和

GPRS 的无线通信实验。此外，Fuxi 语言及 Fuxi 平台的一部分，已经作为一种嵌入式开发环境，在嵌入式产品中应用。

经过我们在嵌入式和桌面环境中的应用测试，表明：

1) Fuxi 语言在描述用户任务，开发应用程序上确实表现的强大的表达能力，能够节省开发时间；
2) Fuxi 平台的这种机制和策略的正交分离的方案，确实能够到达模块化普适计算的不同关切(concerns)，屏蔽技术实现的目的；
3) Fuxi 语言表现了良好的执行效率，这主要得益于 Fuxi 的基本包为本地包，采用 C 代码实现，而实际通过抽象机解释执行的只有应用层的用户程序。

作为普适计算平台的 Fuxi 项目尚未完成，我们正在对 Fuxi 平台作进一步的开发和测试，特别是平台的环境感知系统(context-awareness)。同时 Fuxi 语言的一个集成化开发环境也在开发中。

最终，我们将把 Fuxi 平台实现为一个支持多 CPU 类型，多操作系统环境及多通信手段的，并且开发环境友好的普适计算环境。

致谢：在本项目的开发过程中，得到了“深圳市卓讯达技术发展有限公司”的支持，为我们无偿提供了其手机开发平台和 GPS 开发环境，并对我们的软件进行了测试，提出了许多宝贵的建议。在此向“卓讯达”的总经理郑欣先生及其他同仁表示衷心的感谢。

参考文献

[1] Weiser, M. “ The Computer for the 21st Century”. *Scientific American* , Sep. 1991. pp.94-100

[2] M. Satyanarayanan, “Pervasive Computing Vision and Challenges,” *IEEE Personal Comm.*, vol. 6, no. 8, Aug. 2001, pp.10–17

[3] Wireless LAN Alliance, “The IEEE 802.11 Wireless LAN Standard,” available at: http://www.lana.com/intro/standard/intro.html

[4] P. Rysavy, “General Packet Radio Service (GPRS),” *GSM Data Today J.*(online), Sept. 1998.

[5] Garlan, D. et al, “Project Aura: Toward Distraction-Free Pervasive Computing”, *IEEE Pervasive Computing*, Jul. 2002, pp.22-31

[6] Chen, H., et al. “An ontology for context-aware pervasive computing environments”, *Knowledge Engineering Review*, Vol. 18:3, pp.197–207. 2004, Cambridge Uni. Press

[7] Grimm, R. et al., “A system architecture for pervasive computing”, Proc. of 9th workshop on ACM SIGOPS European, 2000, pp.177-182

[8] Kiczales, G., et al. “Aspect-Oriented Programming”, *ACM Computing Surveys* 28(4es), Dec. 1996, Available at: http://www.acm.org/pubs/citations/journals/surveys/1996-28-4es/a154-kiczales/.

[9] Xerox Parc. The Aspect-Oriented-Programming website. http://www.parc.xerox.com/csl/-projects/aop/

[10] Tarr, P. et al., “N degrees of separation: Multi-dimensional separation of concerns”, Proc. ICSE 1999, pp.107-119

[11] Hitrend, *The specification of Fuxi programming language*. Available at: http://www.fuxi.org

[12] Arnold, K., Gosling, J., Holmes, D., *The Java Programming Language*, Addison Wisley Longman, Inc. 2000

[13] 郑纬民等, *函数型程序设计语言*, 清华大学出版社 1997

[14] Atkinson, M.P. et al., "Orthogonally persistent object systems", *VLDB Journal*; 4(3), 1995 pp.319-401

[15] Peyton Jones, S. L., *The Implementation of Functional Programming Languages*, Prentice-Hall, 1987

[16] 王鼎兴等, *逻辑程序设计语言及其实现技术*, 清华大学出版社 1996

[17] Morrion, R. et al. "A Compliant Persistent Architecture", *Software—Pract. Exper.* 2000(30): pp.363-386

[18] Hitrend, *The guide of Fuxi System Development Toolkit (Fuxi SDK)*. Available at: http://www.fuxi.org

Fuxi: A Language and Platform of Pervasive Computing

Victor Z. Wang

（Laboratory of Novel Computing, Hitrend Information Technologies, Inc., 518048, Shenzhen）

E-mail：victor@fuxi.org

Key words： pervasive computing, aspect-oriented, programming language, declarative language, Fuxi language, middleware

Abstract：Pervasive computing, with its focus on the users and their tasks rather than on the computing devices and technology, provides an attractive vision of the future of computing. But how to integrate the heterogeneous devices and technology together, and how to build an uniform environment, which facilitates the pervasive computing, but adds little more complexity to the system, become the Hotpoint of the domain. Here, we propose an architecture for pervasive computing, which separates the mechanism and policy, and makes them two orthogonal aspects. And we designed an architecture-independent, object-oriented declarative language – Fuxi, as the task description in the policy aspect; and Fuxi platform as the infrastructure supporting the execution of user tasks in the mechanism aspect.

Fuxi:一种普适计算的语言及平台

王忠斌

（深圳市海创达资讯技术有限公司计算新技术实验室，深圳，518048）

摘　要：普适计算，聚焦于用户任务本身，而不是完成其计算的设备和技术，为我们展示了一种新型的计算前景。如何将不同的计算技术综合到一起，建立一个统一的编程环境，并使用户程序不因系统环境的复杂而增加新的复杂度，将成为实施普适计算的关键。我们根据普适计算的特点，采用面向方面的软件技术，将系统分解为机制和策略两个正交的方面。开发了一种平台无关的、面向对象的说明性语言——Fuxi 语言，用于在策略面进行计算任务的描述；而 Fuxi 平台则从机制面为用户任务的执行提供底层支持。

Fuxi(伏羲)程序设计语言是一种平台无关的、面向对象的、逻辑型与函数型结合的说明型语言，简单地说，**Fuxi 是说明型的 Java**。Fuxi 在语言级对风格提供支持，可以让程序员只关注类结构本身的设计，关注问题的本身，而忽略对同问题方案正交的某些技术侧面的考虑。其中，mobile 和 remote 风格使得 Fuxi 能够在语言级直接对普适计算予以支持，使其成为一种普适计算的语言。Fuxi 所采用的文法成分都是流行可以概括为以下公式：

Fuxi 文法 = 类似 Java 的类型框架 + 模式匹配的方法构成

Fuxi 的详细资料可参考：http://www.fuxi.org

关键词：普适计算；面向方面；程序设计语言；说明型语言；Fuxi 语言；中间件

Fuxi: A Language and Platform of Pervasive Computing

Victor Z. Wang

（Laboratory of Novel Computing, Hitrend Information Technologies, Inc., 518048, Shenzhen）

E-mail：victor@fuxi.org

Key words： pervasive computing, aspect-oriented, programming language, declarative language, Fuxi language, middleware

Abstract：Pervasive computing, with its focus on the users and their tasks rather than on the computing devices and technology, provides an attractive vision of the future of computing. But how to integrate the heterogeneous devices and technology together, and how to build an uniform environment, which facilitates the pervasive computing, but adds little more complexity to the system, become the Hotpoint of the domain. Here, we propose an architecture for pervasive computing, which separates the mechanism and policy, and makes them two orthogonal aspects. And we designed an architecture-independent, object-oriented declarative language – Fuxi, as the task description in the policy aspect; and Fuxi platform as the infrastructure supporting the execution of user tasks in the mechanism aspect.