计算机科学 1997 Vol. 24 №. 1

开放式分布处理的参考模型

郑链浩 陈涵生
(华东计算技术研究所 上海 201800)

摘 要 开放式分布处理的参考模型(RM-ODP)是由国际标准化组织 ISO 和 ITU-T 为开放式分布处理 ODP 而开发的一个协调框架。此模型描述了一个集成的支持分布式、协同工作(interworking)的、可互操作的和可移植的体系结构。RM-ODP 框架用五个观点来定义 ODP 的内容。这五个观点分别是:计划观点、信息观点、计算观点、工程观点和技术观点。本文介绍此参考模型,并且描述其观点和 ODP 的有关功能,以及相应的透明性。

关键词 计算机通信网络,分布式系统

1 引言

目前大多数计算机系统都是异构的、多厂商的分布式系统,比如基于 INTERNET 网络的异构性系统等。因此除了以前的系统互联外,迫切需要系统间的互操作,而异构分布式系统的构造取决于建立适用于系统部件所需行为的标准。十年前开发的开放系统互联标准的基本参考模型 OSI-RM 是为基本通信标准而开发的主要框架,着重于点到点的通信,其范围局限于系统的互联,而分布式处理所需处理的问题远远不止这些。虽然出现了改进 OSI-RM 的应用层次结构 ALS 的标准化活动,但还是不能解决与分布式相关的问题。ISO 和 ITU 不约而同地意识到规范化分布式系统的重要性。虽然二者的出发点不同,但这二者正在合作进行开放分布式处理 ODP 的标准化工作,建立 ODP 参考模型(ODP-RM)的目的,希望把各种分布式系统的标准集成起来,此参考模型除了说明特定接口的行为外,还应支持跨系统的一致性和全局性的规格说明。与官方分布式处理标准化工作并行的还有一些国际化组织开发的分布式处理的体系结构与事实上的标准,例如:OSF 组织开发的 DCE(分布处理环境),OMG 组织开发的 CORBA(公共对象请求代理体系结构)。ISO 和 ITU-T(其前身为 CCITT)已经开发了一个开放的分布处理的参考模型 RM-ODP,创建一个集成的支持分布式、协同工作的、可互操作的和可移植的体系结构,为 ODP 标准提供一个协调的框架。

1.1 RM-ODP 的目标

RM-ODP 目的在于达到:

- 异构型平台之间,应用程序的可移植性,其中包括数据和软件的可移植;
- ODP 系统间的协同工作,即在分布式系统中,信息的有益交换和功能的方便使用,以及系统各部件能无缝地协同工作,尽管在设备、网络、语言、数据库模型或管理权力等方面的异构性。一个 ODP 系统须有支持对用户和应用屏蔽异构性的机制,它们可表达为一系列基本透明性的特征,包括:访问透明性、位置透明性、迁移透明性、联邦透明性、资源透明性、失效透明性、组透明性、复制透明性和事务透明性。

访问透明性:屏蔽了数据表示和激活机制的差别,这样能在异构的计算机体系结构、编程模型和网络协议间协同工作。解决了异构系统交互工作时的许多问题,一般的 ODP 系统都有这种透明性。

位置透明性:屏蔽了界面的位置,包括远程和本地激活的差别。这样就能独立地标识界面,以及在界面重定位后,能继续使用界面。

迁移透明性:是位置透明性的扩展。一个被重定位的对象屏蔽了对象的动态重定位。迁移常被用来达到负载平衡和减少迟延。

联邦透明性:屏蔽了在联邦行政域和异构的技术域间的协同工作界限。

资源透明性:屏蔽了系统能力的差异,以替每个参与动作的对象提供必要的资源。

失效透明性:屏蔽失效以及对象的可能恢复,以加强容错能力。

组透明性:屏蔽成组对象的使用(即对外不显示已使用成组的对象),以支持单一的界面。

复制透明性:组透明性的特殊情况。在此关于它

们的行为，各成员是相互兼容的。它常用来提高性能和可用性。

事务透明性：屏蔽了事务性操作的协同性，它们作用于某一界面上的共享状态，这样就可达到一致性。

·分布的透明性，即对应用程序的编程者和用户隐藏分布式的影响。

·变动的透明性，即系统能集成最新的技术。

ODP_RM 为达到以上这些目标，必须完成二件事：

(1)必须描述一系列的集成功能，能提供所需的透明性。提供这些透明性的现实软件部件称为 ODP 基础设施。在 ODP_RM 下正在开发的推荐书和国际标准将标准化 ODP 基础设施部件。分布式应用部件将通过 ODP 基础设施而互操作，这将使分布式网络计算成为现实的一个平台；

(2)必须提供用于描述界面的技术。ODP 设施将允许 Client 应用部件访问 Server，不管 Client 位于网络中何处，不管 Client/Server 用了什么语言，不管什么样的本地操作系统。一个分布式系统的部件可能由不同的人在不同的时间，用不同的技术开发的，而这些部件必须协同工作。因此，对于 Client 部件的开发者来讲，有一个 Server 接口的精确描述是必要的，其规格说明必须清楚并且能独立地实现。在运行时，接口说明是确保客户所期望的接口与由 Server 提供的相兼容的一种工具，以致基础设施能影响一个类型检查的联编。

参考模型提供了一个"总体构架"，以使把 ODP 系统的各部件组成一个具有内聚性的整体，它既不准备对系统的各部件进行标准化，也不想试图对技术的选择施加什么影响。

参考模型的开发中存在着许多挑战。RM-ODP 的描述应适用于目前和未来的分布式系统。所以 RM-ODP 是抽象的，但不是含糊的。RM-ODP 细致地描述了它的各个部件，但没有规定其实现。

1.2 ODP 参考模型的结构

RM-ODP 标准被称为 ISO 国际标准 10746 或 ITU-T X.900 系列推荐书。由四部分组成：部分 1：使用指南和综述(ISO 10746-1/ITU-T X.901)；部分 2：模型描述(ISO 10746-2/ITU-T X.902)；部分 3：模型规定(ISO 10746-3/ITU-T X.903)；部分 4：体系结构的语义(ISO 10746-4/ITU-T X.904)。部分 1 包含一个 RM-ODP 的综述，给出了范围、理由和关键概念的解释，以及 ODP 体系结构的轮廓；部分 2 给出了在详细说明分布处理系统时，所需概念的精确定义，以及分布式处理系统形式化描述的分析框架；部分 3 规定了开放分布处理系统所需的概念、结构、规则和功能的一个框架，这些对于使得分布式系统成为开放的是必须的，即 ODP 标准须遵守的约束，用了部分 2 中的描述技术；部分 4 为 ODP 模型化概念的形式化，描述了怎样用形式化描述技术来表示部分 2 中的模型化概念。本引论着重于部分 3 所提到的框架。

1.3 RM-ODP 的状况

1995 年 1 月，RM-ODP 的部分 2 和部分 3 成功地投票通过，成为国际标准。通过一些必要的行政处理，马上可得到此标准的官方拷贝。部分 1 和部分 4 是基于部分 2 和部分 3 的，所以它们的标准将在部分 2 和部分 3 的标准出现之后推出。部分 1 和部分 4 当前已是委员会草案，在 1995 年 4 月成为国际标准草案，在 1996 年初成为国际标准。

2. 观点

观点是用来描述界面的抽象。为达到完全的定义(界面做什么及如何做)，须用不同的抽象即不同的观点。每个观点代表原始系统的一种不同的抽象。

RM-ODP 的部分 3 用 ODP 系统的抽象或视图的观点来规定其框架，对每个观点给出了一系列的概念、结构和规则，同时提供这种观点下说明 ODP 系统的"语言"。

RM-ODP 定义了五个观点：企业(Enterprise)观点(目的、范围和政策)、信息观点(信息的语义和信息处理过程)、计算观点(功能的分解)、工程观点(支持分布所需的基础设施)、技术观点(实现技术的选择)。

企业观点，相对于它们交互的环境和系统来讲，着重于商业策略、管理策略和使用者-人的作用。已建的模型也可用来描述置于大量组织间的交互作用之上的约束。关注企业对信息系统要实现的功能需求；

信息观点，着重于信息模型化，提供一个覆盖信息源、槽(sink)和信息源的共同视图。关注企业的信息需求；

计算观点，着重于数据结构，提供了分布式系统的功能，即"系统应做什么"。关注表示信息的数据结构和能自动执行信息处理功能的算法；

工程观点，着重于分布式机制，提供分布式所需的各种透明性，即"系统怎样来完成"。关注计算模型

的实现；

技术观点，构造分布式系统的软、硬件和软件部件。关注构造分布式系统各组成部分的实现细节。

观点并没有形成一个层次结构，这里并没有什么先后次序，特别是没有技术上的顺序。

用每一种观点的语言来详细说明一个 ODP 系统，使此系统的大而复杂的规格说明分成可管理的片。每一片着眼于与开发小组不同成员相关的不同问题。例如，信息分析者的工作在于信息的规格说明，而系统编程者关心的是工程观点。图 1 显示了 RM-ODP 的观点怎样与软件工程的过程相联系起来。

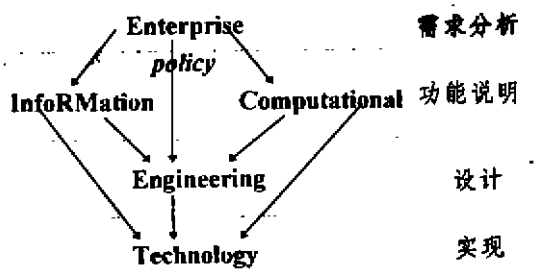

图 1　RM-ODP 观点和软件工程

2.1 企业观点

企业观点适用于机构的需求和结构。此观点中，社会的和机构的策略可用下面的术语来定义：

对象　既有“主动”对象，如银行主管、出纳员、顾客，又有“被动”对象，如银行帐号、钱；

团体　为达到某目的的一组对象。例如，一个银行的分支机构由一个银行主管、一些出纳员和一些银行帐号组成，这个分支机构对某一地区提供银行服务；

团体中对象的作用，可用策略来表达：①准许，即能做什么。例如，钱可存入所开的帐号；②禁止，即不允许做什么。如客户不能在一天内取出多于 $500 的钱；③义务，必须做什么。如在利率调动时，银行主管须通知客户。

企业语言的重点是能改变策略的执行动作，如创建义务或废除准许等。在一个银行里，改变利率是一种执行动作，即它创建了银行主管通知客户的义务。然而获得一个帐号的余额，并不是执行动作，因为义务、准许和禁止都未受影响。因此银行的企业说明不应包括获得帐号余额的动作，它将在计算规格说明中作为功能说明。

当准备一个 ODP 应用系统的企业说明时，策略是由机构决定的，而并非由技术选择（实现角度上）强加于机构的。例如，不应限制一个客户仅拥有一个帐号，这仅仅是为了程序员的方便罢了。

2.2　信息观点

信息观点用于描述 ODP 应用所需的信息，即通过对象的状态和结构的模式来描述其信息。例如，一个银行帐号由余额和“当天所提取的金额”两项组成。

静态模式：抓住了一个对象在某特定时刻的状态和结构。例如午夜时刻的“当天所提取的金额”为 $0。

不变模式：关于所有时间内对象的状态和结构的约束。如：“当天所提取的金额”小于等于 $500。

动态模式：定义对象的状态和结构所允许的变化。如：当从帐号中取走 $X 时，其余额减少 $X。而“当天所提取的金额”将增加 $X。动态模式常受不变模式的限制。因此，在上午提取 $400 是允许的，但在下午提取 $200 是不允许的，因为“当天所提取的金额”将超过 $500。

模式也能描述对象间的关系或关联。例如静态模式“拥有帐号”能把每个帐号与一个客户相关联。

为了描述复杂的或复合的对象，一个模式可由其它模式组合而成。例如，一个银行分支机构由一系列帐号、客户和“拥有帐号”关系所组成。

ODP 应用的信息说明可用许多方法来表达。如实体关系模型、概念模型和 Z 语言的形式描述技术。

2.3　计算观点

计算观点以分布透明的方式来说明一个 ODP 应用的功能。RM-ODP 的计算观点是基于对象的，也就是说：①对象封装了数据和处理过程（即行为）；②对象提供了用于与其它对象交互作用的界面；③对象能提供多个界面。

计算规格说明定义了 ODP 系统中的对象、对象中的活动以及在对象间出现的交互作用。计算（观点）规格说明中的大多数对象，描述了应用的功能，而这些对象通过发生交互作用时的约束（联编）进行联接。联编的对象用于描绘对象间复杂的交互作用。

一个计算（观点）的规格说明中对象可以是应用对象（如银行分支机构），也可以为 ODP 的基本设施对象（如类型公用库或贸易者）。图 2 示例了一个银行分支机构，提供一个银行出纳员界面和一个银行主管的界面。两个界面都可以用来存取钱，但帐号仅能通过主管界面来创建。银行分支机构对象的每个

界面只对应于一个客户对象。

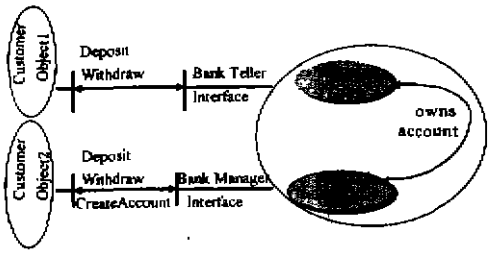

图 2　银行分支机构对象与银行管理者和银行出纳员的界面

2.3.1 计算的交互作用　RM-ODP 提供了对象间交互作用的三种形式：操作式的、面向流的和面向信号的。

操作式的界面对分布计算提供了 C/S 模型：客户对象在服务对象的界面上激活操作(即远程过程调用范式)。操作式的界面由命名的操作组成，带有参数、终止和结果。在 RM-ODP 中的操作可以是询问型(即返回一个终止)，也可以是宣告型(不返回终止)。

例如，一个银行分支机构对象提供了许多 Bankteller 的操作界面，其特征定义如下：

```
BankTeller=Interface Type{
    operation Deposit(c:Customer,a:Account,d:Dollars)
    returns OK(new_balance:Dollars)
    returns Error(reason:Text);
    operation Withdraw (c : Customer, a : Account, d :
      Dollars)
    returns OK(new_balance:Dollars)
    returns NotToday(today:Dollars,daily_limit:Dollars)
    returns Error(reason:Text);
    }
```

注意，上例用到的记号只是示范性的，为定义操作界面类型，RM-ODP 没有规定任何特殊记号。

流界面(逻辑上)提供了生产者和消费者对象之间不间断的信息流。消费者对象与生产者对象的流界面相连，或反之亦然，且多个流可组合在一个界面上。例如，一个听觉流和一个视觉流。为了迎合多媒体和远程通信的应用，流界面已包含在 RM-ODP 中。

操作界面和流界面的基础是信号界面，它提供非常低级的通信动作。OSI 的服务原语(REQUEST，INDICATE，RESPONSE 和 CONFIRM)，即是信号的例子。

2.3.2 界面子类化　界面类型的概念，在 RM-ODP 中特别重要。在计算模型中，界面是强类型的，并且(通常)界面类型的继承将创建子类型的关系。一个界面类型的子类型可代替父类型(超类型)。

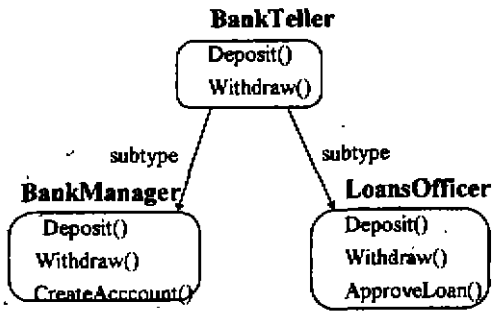

图3　界面子类化的例子

图3举例说明了界面子类化：BankManager 和 LoansOfficer 界面类型是 Bankteller 界面(超)类型的子类型。去执行 Bankteller 所期望的操作 Deposit 和 Withdraw 时，每个都可替代 Bankteller。但 Bankteller 和 LoansOfficer 都不能替代 Bankmanager，因为两者没有一个能提供 CreateAccount 的操作。

2.3.3 计算活动　计算观点也定义了可能在一个计算对象中的动作。它们是：创建和删除一个对象；创建和删除一个界面；界面的"交易"(trading)；界面的联编；对象状态的读/写；在操作界面处激活一个操作；在流界面处产生/消耗一个流；在信号界面处初始化或响应一个信号。这些基本动作可以顺序或并行地组合。假如并行组合，则并行活动可以是相互依赖的，(此活动被分叉，因此相应地必须在同步点相结合)，也可以是独立的(此活动是派生的，不能结合)。

2.3.4 环境合同　计算对象和其界面的精炼可能需要对象或其界面实现时的需求说明(因此对象通过界面交互作用)。例如，银行必须保护客户的钱，且必须保证交互作用的安全，以防大量的诈骗活动，例如捕获和重复操作(Capture-and-replay)。所以实际的交互作用须要么通过安全的网络进行通信，要么须提供端到端的安全检查。

理论上，环境合同将用高级的服务质量(QoS)来表达，而不是诸如：详细说明一个特殊的网络或一个加密模式(二者都需预测此 ODP 系统将运行的环境)。

当前的技术水平缺少这种概念。然而 RM-ODP 须是"防患未然(future-proof)"，应能把当前和将来的技术结合起来，这一点是非常重要的。

2.4　工程观点

工程观点适用于描述一个 ODP 系统中的面向分布式方面的设计。它为分布式系统的基础实施定义了一模型。工程观点不关心 ODP 应用的语义，除了决定用于分布性和分布透明性的需求外。

工程观点描述的基本实体是对象和通道。工程观点中，对象可分为二大类：基本工程对象（相应于计算规格说明中的对象）和基础实施对象（如协议对象，见下）。通道对应于在计算规格说明中的一个联编或一个联编对象。

2.4.1 通道 提供了通信机制，并包含或控制基本工程对象所需的透明性功能，正如计算规格说明中的环境合约所阐述的。图4示意了图2中所示的 Customer 对象和 BankBranch 对象间的通道。其中阴影部分为通道，由体裁根（stub）、联编器（binder）和协议对象（protocol object）组成。stub 和 binder 用来提供各种分布透明性。

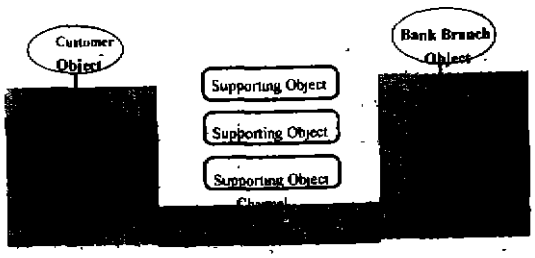

图4 通道的结构

当透明性涉及一些应用语义的知识时，要用 stub。例如为了查帐审计而维护的操作日志文件。

当透明性不涉及一些应用语义时，要用 binder，它们仅仅传送信息（位流）。binder 负责管理基本工程对象间的联编。例如 binder 可用顺序号来阻止 Capture-and-play 的企图。

协议对象通过通信界面交互作用，这样就模化了网络。

在通道外面，支持对象（Supporting object）协助通道内的 stub、binder 和 protocol object。特别地，支持对象是 stub、binder 和 protocol object 所需信息的公用库。例如，为了达到位置透明，binder 通过一个称作重定位器（relocator）的支持对象来登记和遍历界面的位置。

2.4.2 工程结构 RM-ODP 工程观点规定了 ODP 系统的结构。其结构的基本单元为：①簇 Cluster。一些相关的基本工程对象，它们常被协同定位。②舱 Capsule。一组簇、每个簇的一个簇管理者，一个舱管理者和与它们的界面相连通道的部分。③核心对象 Nucleus object。一个支持 ODP 的（扩展的）操作系统。④结点 node。一个计算机系统。图5示意了一个结点的结构。

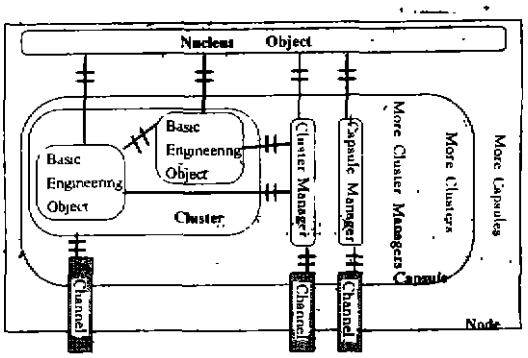

图5 结点的结构

给定了这些定义后，可定义以下结构化规则：①一个结点有一个核心对象；②一个核心对象可支持多个舱；③一个舱可包含多个簇；④一个簇可包含许多基本工程对象；⑤一个基本工程对象可包含许多活动；⑥所有簇间的通信是通过通道的。

ODP 系统的一个实现可进行选择来限制此结构化，例如：每个簇仅一个对象；每个舱仅一个簇。

2.5 技术观点

一个 ODP 系统的技术说明描述此系统的实现和测试所需的信息。对于技术说明 RM-ODP 没什么可用的规则。

3. ODP 功能

ODP 功能是一个 ODP 系统所必需的一些功能的集合，以支持计算语言（trading 功能）和工程语言（如重定位器）的需要。下面列出了 RM-ODP 中的主要功能组。为了示意 RM-ODP 的范围，一些功能作了详细讨论。

3.1 管理功能

RM-ODP 定义了许多管理工程结构的功能，包含：①结点管理功能（由核心对象提供），用于创建舱和通道；②舱管理功能（由舱管理者提供），用于簇的实例化、及舱中簇的检查点与去激活；③簇管理功能（由簇管理者提供），用于检查点、去激活和簇的迁移；④对象管理功能（由基本工程对象提供），用于检查点和删除基本工程对象。

3.2 协调功能

为了产生某种一致的整体影响，RM-ODP 定义

了许多功能，目的在于协同对象、簇或舱的动作。包括：检查点和恢复、再激活和去激活、事件通告、组和复制、迁移、事务。

3.2.1 事务功能 信息观点中，状态改变作为单个不可分割的动作出现。然而用计算和工程观点，这个状态可被分布于整个 ODP 系统并且可被许多并行活动并发地访问。为了开发可靠的 ODP 系统，有必要协同对象的行为以达到期望的程度：

可视性 操作的中间结果对其它操作的可见程度；

可恢复性 操作(哪些结果不能再现)失败后的状态；

稳定度 对于已完成的操作，操作失败的后果(其结果改变吗?)。

RM_ODP 定义了一个很一般化的事务功能，这是 RM_ODP 中"future_proof"的另一个例子。实际上，ACID 模型将在多年内成为大多数 ODP 系统支持的事务机制的唯一风格。因此，RM_ODP 定义了一个 ACID 事务功能作为一般化事务功能的特化。

3.3 公用库功能

除了一个一般的存储功能和一个一般的关系仓库外，RM_ODP 还定义了许多特殊的仓库功能，主要关于维护一个特殊类型信息的数据库。

3.3.1 类型公用库 在大多数计算机系统中，在一个系统中没有显式地维护类型定义。事实上，类型归档在手册中或根据本地规范来定义(如记忆性文件名的使用)。ODP 系统须在整个 ODP 系统中可用；在'交易'和界面联编期间，最主要的需求是支持类型检查。

在 RM_ODP 中，类型公用库是类型定义的登记处，特别对于界面类型。类型登记处维持一个类型层次(子类型关系)和类型间的其它关系。

3.3.2 交易者(Trader) 它是一个特殊的体系结构功能，需要标准来支持与类型化对象界面的动态匹配。

ODP 的交易者提供了"给对象的适时服务"，其目标在于提供允许运行时发现服务，支持动态联编。交易者是服务广告的公用库。

服务器对象通过交易者来广告它们的服务，此服务广告说明了界面类型和服务属性。服务器通过使用交易者提供的 export 操作来处理它们的服务广告。客户通过在 import 操作中说明所需的类型和属性来选择服务。

ODP 的交易者也是标准化的内容，但独立于 RM_ODP。

4. 透明性

透明性概念是 ODP 体系结构的基石。透明机制提供了一个增强环境，它位于分布式平台的低级操作系统和通信设施的上层。

4.1 访问透明性

对客户对象隐藏了存取一个给定服务器对象的机制，包括数据的具体表示及调用机制。访问透明性屏蔽了提供远程/本地服务间的差别，客户不必知道在服务器界面处的存取机制。

4.2 重定位的透明性

重定位的透明性使一个基本工程对象(和对象的编程者)无需知道一个交互对象是否被重定位。

可用联编器(binder)来配置通道以达到重定位的透明性。即：①通知通道支持的界面所在位置的重定位器；②从重定位器得到与通道相联的其它界面的位置。联编器将特别存储位置信息。假如一个界面的位置改变了，使用老的位置将造成错误。具有重定位的透明性，联编器将自动地从重定位器得到新的位置，重新联接通道，重复交互作用。基本工程对象不关心位置的改变。

4.3 事务透明性

不象访问和重定位透明性(通过用聪明的部件来配置工程通道而达到的)，事务透明性不能单独通过这种机制来达到。

事务功能的正确操作需要报告某些"有益动作"的执行或取消。如读或写一段事务管理的数据。这些事件在对象内部发生，且它们对被配置在通道内的体裁根或联编器是不可见的。所以事务透明性须包括一更新：把一个事务透明性规格说明更新成这么一种规格说明，它汇报对事务功能来讲有益动作的执行情况。

总结 RM_ODP 是一个参考模型，并不是一个实现标准；它为开放的分布式处理系统定义一个框架。RM_ODP 定义了五种观点，通过侧重于独立的主题，它们分解了一个 ODP 应用的规格说明。企业观点定义用于策略分析的模型，而信息观点为信息分析提供了一个模型。计算观点为分布式编程语言定义一个模型。由基于工程观点模型的分布式系统的基础设施和 ODP 基础设施功能来支持这些语言的运行。而技术观点模型用于描述已实现了的系统。

参考文献

[1] Gerd Schurmann, The evolution from open systems interconnection (OSI) to open distributed processing (ODP), Computer Standards & Interfaces 17, 1995

[2] Kerry Raymond, Reference of Open Distributed Processing (RM-OPD): Introduction, ICODP'95