以服务器为中心的面向服务网络计算架构的研究与实现

余文骏　　朱永华　　徐炜民

(上海大学计算机工程与科学学院　上海市 200072)

摘　要：本文首先介绍了网络计算现状及本项目发起的原因，然后阐述了一套自主开发的以服务器为中心的面向服务网络计算平台(Service Oriented Infrastructure for Network Computing, SOINC)系统的结构及其工作原理，并以软件仿真验证了该系统实现的可行性。

关键词：网络计算　服务　资源　SOINC

Study and implementation of the Server-Centralized Service-Oriented Infrastructure for Network Computing

ManChon U　　Zhu Yonghua　　Xu Weimin

(Department of Computer Science and Engineering，Shanghai University，Shanghai 200072)

Abstract：This paper first introduces the current development of Network Computing and the reason of starting this project, then expatiates the structure of the Server-Centralized Service-Oriented Infrastructure for Network Computing (SOINC) system. Furthermore, this project validated the feasibility of the system by software simulation.

Keywords：Network Computing，Service，Resource，SOINC

1.　引　言

目前，网络计算(Network Computing)正处于发展阶段，人们对它的定义还没有形成共识，但一个相对可以接受的理解是：网络计算是利用互联网上计算机的 CPU 的闲置处理能力来解决大型计算问题的一种计算科学，它研究如何把一个需要非常巨大的计算能力才能解决的问题分成许多小的部分，然后把这些部分分配给许多计算机进行处理，最后把这些计算结果综合起来得到最终的结果[1]。

集群（Cluster）就是一组计算机[2]，它们作为一个整体向用户提供一组网络资源。这些单个的计算机系统就是集群的节点（Node）。我们在使用集群进行并行计算时，发现有许多串行化程度高的任务可不必占用昂贵的专用并行计算设备去进行计算，由此引发了上海大学网格研究室对本项目— Service Oriented Infrastructure for Network Computing (SOINC) 的构想。本项目的主要目的就是要建立一套能够适应更加通用的多任服需要，并且可以为程序开发用户提供非专用的网络计算服务，以实现更大程度的资源共享和更高效的计算能力。

2.　SOINC 结构剖释

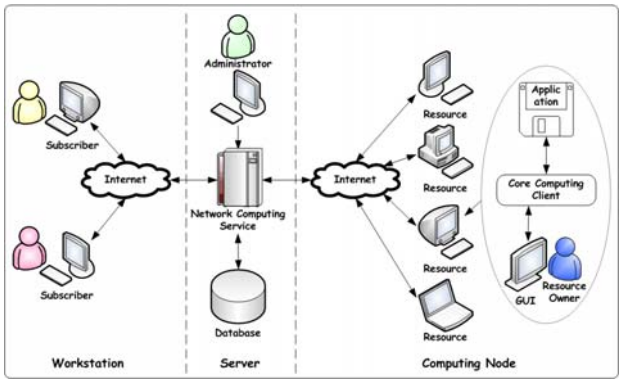

图 1　SOINC 总体结构

整个 SOINC 系统主要由三部分组成(图 1)：

(1) 工作用户(Subscriber)开发、提交、控制任务作业及处理数据的工作站(Workstation) ；

(2) 协调所有工作的网络计算服务(Network

基金项目：上海市教委 E 研究院-上海高校网格项目和上海市教委自然科学项目(03AK70)

作者简介：余文骏(1984-)，男，学士。主研领域：网络计算，高性能计算。(电子邮件: manchonu@yahoo.com)

Computing Service)的服务器(Server) ；

(3) 提供计算资源的计算结点(Computing Node)。

在 SOINC 中能够提供一定计算能力和相关设备的集合就称为服务。比如一台含 SOINC Client 端的计算机就可以封装为一个服务[3]，封装为服务的资源可以在任何时候注册到共同体中。共同体在此扮演着一个资源池(Resource Pool)的角色[3]。

计算结点可以是由 Internet 上的志愿者项献的个人计算机，组织内部非专用的桌面计算机，也可以是分配给系统的超级计算机的一部份专用结点。计算结点上需部署 SOINC 计算客户端软件，并且资源所有者对该软件和机器有绝对的控制权。可以设定结点加入某个网络计算服务站点，并接受其分配的计算任务。也可以加入多个不同组织的站点，并在这此站点间根据优先级进行切换和平衡。

计算客户端软件可以分成三个部分运行。第一部分是核心计算客户端(Core Computing Client)，它负责和服务器通讯，并最终完成所有的通讯、控制和存储任务。把它和 GUI 界面分离也是能为了商应不同的平台情况，从而有效地集中更多的计算资源。

完成每个任务所需要的可执行程序称为应用程序(Application)，Application 和所需要计算的任务(Task)相对应。Application 是由 Core Computing Client 将 Result 回传给服务器。

GUI 是方便资源所有者监视和控制资源运行状态的图形前端程序。它通过本地的 Socket 接口和 Core Computing Client 通讯。每个资源所有者拥有一个账号，资源所做的计算工作都会转换成积分记录在该账号下。这对于鼓励依靠志愿者的计算项目是十分重要的。在网络计算服务的条件下，这也可以作为换取今后用户可使用的计算量的依据。

工作用户(Subscriber)是任务的编写者和最终使用这一计算服务系统的用户。一套网络计算服务应该能为多个工作用户提供服务，就像理想中的超级计算机的使用方式一样。因此，工作用户不必也不要占用服务器，或去过多的承担系统管理员的职责。他们应该舒服地坐在自己出色的工作站前。那里有满足他们需要和习惯的开发工具及专用软件。当他们能专注于思考自己专业领域内的问题并编写出计算程序时，就可以随时通过客户端、脚本甚至内嵌在程序里的 API 向网络计算服务器提交任务。

一个任务中除了主程序外还有若干个分割好的工作单元(Work Unit)。任务可以是运行几个月、甚至上年的巨型任务，也可以是运行几天或是只需几个小时的小任务。因为使用者不需要去为了一个单独的任务从头开始搭建一套服务器，或召集志愿机。这已经是一套准备好的，可以共享的网络计算服务。他们会感到很值得，因为这样总比单台高端的工作站要快许多，而且是相当廉价甚至免费的。Work Unit 可以是在提交 Task 同时确定的，也可在以后分多次逐步添加，尤其是在运行大型任务时。

当任务计算完成或部分计算完成时，工作用户可以下载一组由多个 Work Unit 生成的 ResultSet。然后可以按他们需要和愿意的方式来处理这些数据，例如，可以是由自动提交的程序来等待结果后直接处理，也可以是手工调用另外的程序执行诸如数据合并的工作。

其中在 SOINC 中主管任务调度及资源管理功能的子系统称为 SS&RMS(SOINC Scheduling & Resource Management System)，它被构建于 SOINC 系统的服务器部份。SOINC 系统的任务调度及资源管理子系统已在[4]中得到成功实现，故在此将不作讨论。而 SOINC 各部份之间的运行关系如图 2 所示。

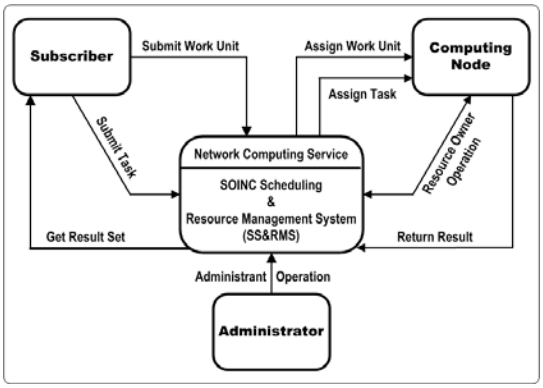

图 2　SOINC 各部份间运行关系

3. 分布式模型的选择

分布式系统，通常有三种编程模型：Master – Worker 模型、Command 模型、MarketPlace 模型。Master – Worker 模型中，分为主控节点和工作节点，主控节点将任务划分成小块放到每个工作节点的空间内，工人发现任务后自动处理，这种模型比较容易实现负载平衡，常在并行计算中使用；Command 模型中也有主控节点和工作节点，不同的是，这个模型中工作节点完全不知道自己该干什么，他将调用任务中某一实现约定的方法来执行工作，这个模型的缺点就是所有任务必须要实现一个固定方法，每个工作节点通过调用这个方法来得知怎么处理任务，这种模型适合与在系统中有多个主控节点，并且执行不同类型的任务时使用。MarketPlace 模型中分为购买者和出售者，整个运作的过程类似投标的过程，首先由购买

者提出标书，出售者看到标书后向购买者投标，最后购买者记录下投标的结果。

经过对 SOINC 系统的结构性的考虑，确定应当使用的是三种模型中的 Master-Worker 模型[5]，由 SS&RMS 首先分配任务，而后按照事先的约定，所有的计算结点各自处理任务，无需从任务中寻找什么特殊的处理方法来调用。因在 SS&RMS 中只会有一个主控节点 — Server，所以采用 Master-worker 的模型。主控节点上部署系统所完成的工作是接受作业提交，负载均衡算法，布置任务环境(也就是将任务运行所必须的档案，发送给获分配该任务的计算结点)，通知工作节点，资源池的维护也是主控节点的工作。任务完成后各计算结点立即通知主控节点，图 3 为 SS&RMS 中的 Master-worker 结构。

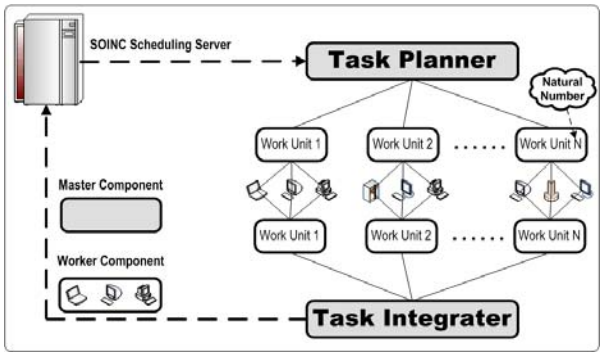

图 3　SS&RMS 的 Master – Worker 结构

4. 软件仿真结果及分析

我们以 Java 语言编写了 SOINC 的核心部份 SS&RMS 及客户端软件，并模拟了多结点使用 SOINC 时的情况。

当 SS&RMS 发现通过合法用户组请求加入 SOINC 的未注册计算资源后，将其注册并纳入到可用资源池中。若该资源为已注册计算资源，则直接将其纳入到可用资源池中。已加入到 SOINC 中的可用资源可能因各种原因(机器故障、断线等)于过期时间前重新请求加入 SOINC，SS&RMS 便重新给予其一个新的过期时间。计算任务子单元被分配到可用资源后，SS&RMS 发现 SOINC 中再无可用资源分配，便输出计算资源不足、正等候计算资源的字句。SOINC Client 被分配计算任务后，随即接收到从 SS&RMS 发来的属于分配到该计算资源上的任务子单元的计算文文件。SS&RMS 接收到计算资源的结果反馈同时把该计算资源重置为空闲可用资源，之后再由不同线程分别对该计算结果进行处理及继续进行调度分配。空闲可用或已被占用计算资源(R1)的过期时间(T1)早于 SS&RMS 的检查点时间(T2)，故当 T2 到来时，R1 被当作失效资源处理。计算任务子单元(WU1)的最久计算到期时间(T1)早于 SS&RMS 的检查点时间(T2)，故当 T2 到来时，WU1 被彻消并重新入队等候调度分配。每个计算任务子单元分配给三台计算资源去计算，若计算资源两台以上反馈的结果相同，便认为该结果正确，写最终结果。若计算资源反馈结果没两个或以上相同，则认为该任务子单元计算失败，将其入队等候重新分配。

5. 总　结

本文描述了能够应用于公共资源计算模式上的 SOINC 系统的结构及其运作模式，通过实验分析，该系统具有以下优点：(1)可伸缩性。服务结点可以动态地增加和删除，系统将自动完成配置，实现了“即插即用”的特性。(2)健壮性。由于 SOINC 中的注册机制，使得单点失败得到有效的处理，整个系统可以在不需要人工干预的条件下自动清除失败的结点，不会由于单点失败而降低性能。(3)安全性。SOINC 中核心功能模块均是是使用 Java 编写的，因此可以充分利用 Java 的安全特性。

整个 SOINC 项目到目前为止只完成了第一期工程，故 SOINC 还有很大的改进空间。例如，由于本期工程中请求加入 SOINC 的计算结点均被假设为可信赖结点，在实际使用中，为防止存在不法用户对本系统的恶意攻击，故在下一期工程中可针对 SOINC 系统安全性方面作更深入研究，以增加 SOINC 整体的安全性及健壮性。

参 考 文 献

[1] David P Anderson. BOINC: A System for Public-Resource Computing and Storage[J]. GRID, 2004, 4~10.

[2] 徐炜民，严允中. 计算机系统结构[M]:电子工业出版社，2003 年 7 月第 2 版.

[3] Zoltan Juhasz, Arpad Andics, Szabolcs Pota. JM: A Jini Framework for Global Computing [J]. CCgrid'2002, May 21 - 24, 2002 Berlin, Germany.

[4] 余文骏. 开放式网络计算平台—任务调度及资源管理系统[D]. 上海大学计算机工程与科学学院.

[5] Kento Aida, Wataru Natsume, Yoshiaki Futakata. Distributed Computing with Hierarchical Master-worker Paradigm for Parallel Branch and Bound Algorithm [J]. CCgrid'2003, P.156.