面向任务的网络虚拟机器人动态更新迁移计划

王洪泊[1] 马忠贵[1,2] 曾广平[1] 涂序彦[1]

(北京科技大学信息工程学院 北京 100083)[1]

(中国科学院自动化研究所复杂系统与智能科学重点实验室 北京 100080)[2]

摘 要 在深入研究与分析 Agelts 系统及其源码的基础上,对网络虚拟机器人 SoftMan 迁移计算涉及到的基本技术问题进行了系统的分析与描述。利用面向任务的动态更新迁移计划方式和动态结构化操作语义,首先把 SoftMan 的移动信息从功能体中分离开来,可以分别对它们进行分析与设计,在设计功能体时无需考虑网络的实际情况,在设计移动信息时也不用考虑功能体的具体实现细节,这样有效地降低了设计和编写 SoftMan 的复杂性,也易于调试和管理。

关键词 网络虚拟机器人,迁移计划,动态结构化操作语义

Task-oriented Dynamic Update Migrating Plan for Virtual Robot in Network

WANG Hong-bo[1] MA Zhong-gui[1,2] ZENG Guang-ping[1] TU Xu-yan[1]

(School of Information Engineering, University of Science and Technology Beijing, Beijing 100083, China)[1]

(State Key Laboratory of Complex Systems and Intelligence Science, The Institute of Automation of the Chinese Academy of Sciences, Beijing 100080, China)[2]

Abstract The paper was dealing with the basic technology problem about migrating computing for Virtual Robot in Network—SoftMan after deeply analyzing and researching on Agelts. Making using of task-oriented dynamic update migrating plan and structured operational semantics, the migrating information is divided from the relative function section, and then they can be respectively designed without thinking about their network practice environment and concrete implementation. The method is not only effective for programmer to low the complexity, but also helpful to test and management.

Keywords Virtual robot in network-softMan, Migrating plan, Dynamic structured operational semantics

1 引言

网络虚拟机器人——"软件人"(SoftMan)[1]在迁移前无法直接制定出自己的旅行图,因为整个迁移过程有时存在关联性,也就是说只有在迁移过程中才能逐步知道下一步迁移目标,同时 SoftMan 所面临的网络环境具有不确定性,因此 SoftMan 的迁移计划需要动态地确定或更新。这些不确定性给 SoftMan 的设计和编写带来了很大的困难。

在 Aglets 系统中,常常把移动信息和任务功能体混在一起[2],这对于任务逻辑比较简单的智体影响不大。但在处理复杂任务时,对迁移及其网络细节的过多考虑直接影响了对应用逻辑的考虑,大大增加了程序员的负担,也给智体程序的调试和维护带来了很大的问题。为克服 Aglets 系统上述缺陷,Concordia 系统采取一个完全独立于智体的数据结构[3,4]。在智体的执行过程中,将在一个相对独立的地方对该智体的迁移进行管理,这就是结构化方式。这种方式能够为智体迁移的定义和跟踪提供十分简单的机制。而且,系统能够允许智体在执行过程中更改其路线,提高了灵活性。然而,Concordia 系统迁移计划的描述能力尚有很大的不足,不能表达多种迁移方式。Mogent 系统采用了结构化迁移机制,弥补了 Concordia 系统的不足。但是在动态制定迁移计划时,Mogent 系统的更新方法需要处理每个旅行步的所有细节[5],这增大了更新方法的设计难度。

本文基于前期对迁移计划模型的研究[6,7,8],提出一种面向任务的动态更新方式制定迁移计划,在一定程度上弥补上述各系统的不足。

2 面向任务的动态更新迁移计划

迁移计划和功能体的结构分离可以实现迁移计划和功能体的动态装配。程序员可以事先完成对 SoftMan 功能体的开发,因为每个 SoftMan 的功能是相对稳定的,所以它所需资源的类型或迁移后采取的操作也是相对固定的,只是具体迁移

到稿日期:2008-07-01 返修日期:2008-09-17 本文受国家自然科学基金项目(60375038),中国科学院自动化研究所复杂系统与智能科学重点实验室开放课题(20060105),国家"十一五"科技支撑计划(2006BAJ04B07-2),北京市自然科学基金项目(4072018)资助。

王洪泊(1972—),男,讲师,博士,CCF 会员,主要研究方向为人工智能、人工生命,E-mail:foreverwhb@126.com;马忠贵(1973—),男,讲师,博士,主要研究方向为智能通信;曾广平(1962—),男,教授,博士生导师,主要研究方向为人工智能、计算机网络;涂序彦(1935—),男,教授,博士生导师,主要研究方向为人工智能、大系统控制、智能管理、人工生命。

到哪个节点上不能确定。因此可以首先制定一个原始迁移计划，在计划中，能确定的因素就先确定，不能确定的因素就暂时置空，在迁移过程中确定后，再用更新方法把它填补上。这样，更新方法只需考虑未知的部分，通常就是目标地址和迁移方式，最终完成迁移计划更新。

2.1 迁移计划的数据结构

一个旅行步指一次迁移所需的所有信息，包括迁移方式、SoftMan_ID、目标地址、迁移条件、入口方法，如表 1 所列。

表 1 SoftMan 旅行步数据结构

旅行步	迁移方式	ID	目标地址	入口方法	更新方法
P	F	ID	A	GM	T

表中各元素的含义如下。

(1) P 是一个二元组，即 $P=(step, status)$。其中 *step* 代表步骤序列，“软件人”一般按序列迁移，只有在更新方法中允许跳步执行时，“软件人”才跳过某些旅行步，执行后面的步骤。*Status* 表示旅行步状态，主要有 3 种状态：①未就绪，说明旅行步中有某些元素尚未确定，“软件人”还不能实施此步的迁移；②就绪，说明此旅行步中已完成更新，可以马上实施迁移；③完毕，说明此步迁移已经实施完毕，迁移控制可以准备下一个旅行步了。一般来说，对 P 的验证在“软件人”迁移之前进行。

(2) F 表示迁移方式，即顺序迁移(SEQ)、选择迁移(SEL)和并行迁移(PAR)。

(3) ID 是指实施迁移的 SoftMan 标识。

(4) A 表示目标地址，指迁移的目标节点地址，说明“软件人”要迁移到哪个节点上去执行任务。

(5) G 是入口方法 M 的前置控制(又可称为卫士方法)，G 方法主要是对环境及“软件人”本身状态进行判断。只有 G 方法返回为真时，才会执行 M 方法。G 方法把任务实现和环境判断分离开，便于提高迁移计划和功能体之间装配的灵活性。M 是真正的入口方法，描述了到达目标节点后要完成的任务。

(6) T 表示更新方法，在本步迁移的任务结束后执行，确定下一步迁移计划。

2.2 迁移计划的生成算法

下面说明迁移计划动态生成过程的算法描述。

Step1 SoftMan 在 A 节点生成，开始了自己的生命周期。程序员在设计“软件人”时就制定了一个原始迁移表，如表 2 所列。这里，能确定的元素都已确定，不能确定的元素等待在迁移过程中动态生成。

Step2 SoftMan 通过环境交互模块向“目录”服务器发出请求，得到唯一的认证服务器(节点 B)的地址(此时可以确定迁移方式为顺序迁移)，此时更新迁移表，如表 3 所列。更新完成后，P_1 显示就绪状态，迁移执行模块按迁移计划实施 P_1 迁移，任务完成后得到两个提供索引服务的服务器地址(节点 C 和 D 的地址)。更新方法 T 通过智能决策系统完成迁移路径的选择，设此时选择下一步向节点 C 迁移，并更新迁移表。

Step3 迁移执行模块按计划实施 P_2 步迁移。任务完成后得到文件 X 与 Y 的存放地(节点 E 和 F 的地址)。调用方法 T，首先生成两个克隆 SoftMan，然后把地址装入迁移表的相应位置并更新，如表 4 所列。

表 2 原始迁移计划表

P	F	ID	A	G	M	T
P_1		SoftMan-ID		Φ	完成认证并得到索引服务器地址	根据得到地址确定下一步迁移
P_2		SoftMan-ID		Φ	查询文件 X、Y 所在节点的地址	根据得到地址确定下一步迁移
P_3	PAR			Φ	获取文件 X	Φ
P_4	PAR			Φ	获取文件 Y	Φ
P_5	SEQ	SoftMan-ID	节点 A 地址	第 3、4 步中的克隆工作已全部完毕，返回生成地并结束了生命	后续任务	Φ

表 3 迁移计划第一次更新表

P	F	ID	A	G	M	T
P_1	SEQ	SoftMan-ID	节点 B 地址	Φ	完成认证并得到索引服务器地址	根据得到地址确定下一步迁移
P_2		SoftMan-ID		Φ	查询文件 X、Y 所在节点的地址	根据得到地址确定下一步迁移
P_3	PAR			Φ	获取文件 X	Φ
P_4	PAR			Φ	获取文件 Y	Φ
P_5	SEQ	SoftMan-ID	节点 A 地址	同表 2	后续任务	Φ

表 4 迁移计划第三次更新表

P	F	ID	A	G	M	T
P_1	SEQ	SoftMan-ID	节点 B 地址	Φ	完成认证并得到索引服务器地址	根据地址确定下一步迁移
P_2	SEL	SoftMan-ID	节点 C 地址	Φ	查询文件 X、Y 所在节点的地址	根据地址确定下一步迁移
P_3	PAR	克隆 SoftMan1-ID	节点 E 地址	Φ	获取文件 X	Φ
P_4	PAR	克隆 SoftMan2-ID	节点 F 地址	Φ	获取文件 Y	Φ
P_5	SEQ	SoftMan-ID	节点 A 地址	同表 2	后续任务	Φ

可见，当前整个迁移表已经完整，SoftMan 只需按步骤完成 P_3，P_4，P_5 步迁移即可。需要注意的是，在 P_3，P_4 步迁移中，克隆 SoftMan 与原 SoftMan 在逻辑上是一体的，它完成任务后必须返回生成地结束生命，这是 SoftMan 分体式迁移的内在要求。P_3 和 P_4 步迁移任务结束后，仍返回 C 节点，开始 P_5 迁移。至此，整个迁移完成，迁移表可以作为 SoftMan 的路由知识保存起来。在下次做类似迁移时，可直接装配使用，大大提高了整个迁移过程的效率。这也是把移动信息和任务体相分离，分别管理的好处之一。

3 三种迁移模式

下面深入讨论 3 种可行的任务实现方式，并对它们进行简要的对比和分析。

3.1 工作流模式

工作流模式的基本原理是：把任务按序列划分，根据 SoftMan 要迁移的目标节点序列，划分出相应的子任务序列，通过事件驱动来触发 SoftMan 执行对应的子任务。

如图 1 所示，SoftMan 首先在主节点 A 中生成，开始了生命周期。当执行完子任务 1 后，按计划此 SoftMan 要迁移到节点 B。在迁移前触发一个迁移事件，进而调用相应的函数完成迁移准备，然后实施迁移。当到达节点 B 后，触发一个迁移完成事件，进而调用相应的响应函数，启动子任务 2 的执

行。依此类推，当 SoftMan 在节点 C 中完成子任务 3 后，整个任务完成，此时 SoftMan 可以返回主节点 A 报告任务完成情况，然后结束自己的生命周期。

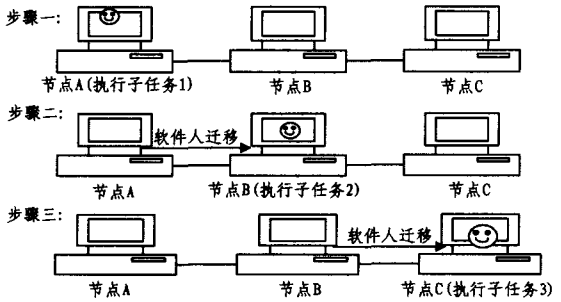

图 1　工作流迁移模式

工作流模式的特点是：整个任务是由 SoftMan 独立完成的，同时在完成任务过程中体现出顺序性。这非常类似于现实世界中的人类处理问题的方式。

3.2　主从模式

主从模式的基本原理：把任务按功能划分为一个或多个子任务，每个从 SoftMan 负责一项子任务的实现，主 SoftMan 负责控制和实现任务的主逻辑并通过分派和管理从 SoftMan，让从 SoftMan 迁移到目标节点并完成具体子任务，从而完成整个任务。图 2 是主从模式图例。首先主 SoftMan 在节点 A 中执行时，发现需要到节点 B 上获取某资源或完成某些操作(可以看作是一个子任务)，此时它本身并不迁移，而是生成一个从 SoftMan，并把它派往节点 B，完成子任务并返回结果。主 SoftMan 得到此结果后，可以按主任务逻辑继续工作，直至整体任务完成。

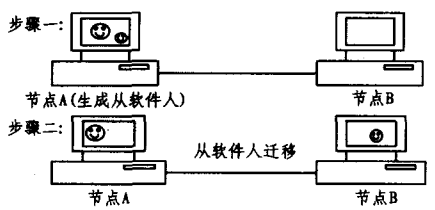

图 2　主从迁移模式

主从模式的特点是：任务是通过多个 SoftMan 之间的协作来完成的。这些 SoftMan 不仅功能不同，角色也不同，它们之间存在着领导与被领导的关系。也就是说，从 SoftMan 总是为主 SoftMan 服务的，并且受主 SoftMan 的支配。这也非常类似于现实世界中人类解决问题的方式。

3.3　分体模式

工作流模式具备完成整个任务的能力，但是对于某些需要通过迁移来完成的子任务，尤其是一些性质相似但要频繁迁移到不同节点上来完成的子任务，显然以工作流模式按序列工作是低效的。此时可生成若干个功能和状态与自己完全相同的克隆 SoftMan，把它们派发出去并行工作，有利于提高工作效率。

需要注意的是，SoftMan 与其克隆在逻辑上是一体的，这样就要求克隆 SoftMan 在任务完成后必须返回出生地，交付结果并结束生命，这作为一种内在机制固化在 SoftMan 内部。这样做的好处是一方面防止 SoftMan 的泛滥，保持其可控性；另一方面在 SoftMan 迁移过程中，我们只需关注原 SoftMan 的迁移情况即可，克隆 SoftMan 的迁移永远只是原 SoftMan 完成任务的一种方式。

用主从模式也完全可以实现派发多个 SoftMan 并行工作，但与分体模式相比，它的成本更高。因为主从模式不仅要设计不同功能的 SoftMan，而且要设计相应的协调机制，因此对于某些相对简单事务，用分体模式更经济实惠。

图 3 给出简单的分体模式图例，SoftMan 在节点 A 中执行时发现需要迁移到 B，C 两节点，执行某类似操作，此时就调用克隆原语。复制出两个状态和功能与自己完全相同的克隆 SoftMan，然后把它们派发到目标节点工作。任务完成后，两克隆 SoftMan 可以返回出生地 A 节点，带回结果并结束生命周期。

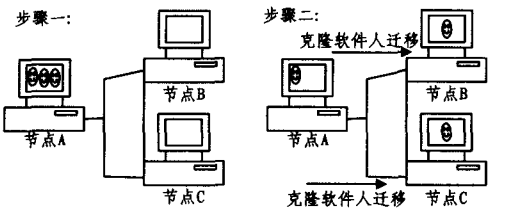

图 3　分体迁移模式

分体模式的特点是：SoftMan 既能独立完成任务，又能在必要时摇身一变，生成多个与自己能力相同的 SoftMan 分头工作。SoftMan 与其克隆人之间只有角色的不同，功能是完全相同的。这恰恰反映了现实人类的一种愿望。例如，任务不方便或没必要另派一个专人去协助完成，但又确实枯燥繁重，此时人们往往会叹息道："我要是能一身二用就好了。"当然，这对于现实的人是不可能的，但计算机却能够做到，这正是网络虚拟机器人优点的体现。

4　迁移机制的操作语义

为了能清楚地定义操作语义框架，这里采用并在选择迁移方面扩充了 Plotkin 提出的结构化操作语义 SOS(Structured Operational Semantics)。为了给出一个程序设计语言的 SOS 描述，应该同时列出 3 部分数据(或者工具)，包括语法范畴、语法规则(含静态语义)及动态语义。

4.1　语法范畴

(1) 控制方法集：p_1，p_2，…。控制方法是一个布尔方法，它根据当前状态进行判断，返回一个布尔值(真或假)，不修改"软件人"的状态。

(2) 主机地址集：a_1，a_2，…。主机地址唯一标识在网络中的一台主机，一般为 IP 形式。

(3) 卫士方法集：g_1，g_2，…。它是一布尔方法，返回布尔值，不修改"软件人"状态。

(4) 入口方法集：m_1，m_2，…。入口方法实现"软件人"的任务逻辑，修改其状态。

(5) 计划更新方法集：t_1，t_2，…。计划更新方法可以动态修改"软件人"的迁移计划，让"软件人"根据新的计划继续迁移。

(6) SoftMan 标识名集：ID_1，ID_2，…。SoftMan 以并行方式迁移时，它本身与其克隆"软件人"之间有着不同的 ID，从而便于互相区别。

(7) 迁移方式集：SEQ，SEL，PAR。说明了迁移计划中

各个旅行步的解释方式。

(8) 旅行步集：S_1，S_2，…，其中每个 S 的结构形如[P，F，ID，A，G，M，T]。

(9) 顺序迁移计划集：seq_1，seq_2，…，其中每个 seq 的结构形如[SEQ，[s_1，s_2，…，s_n]]，$n\geqslant 0$。

(10) 选择迁移计划集：sel_1，sel_2，…，其中每个 sel 的结构形如[SEL，[s_1，s_2，…，s_n]]，$n\geqslant 0$。

(11) 并行迁移计划集：par_1，par_2，…，其中每个 par 的结构形如[PAR，[s_1，s_2，…，s_n]]，$n\geqslant 0$。

4.2 静态语义

空顺序迁移计划的形式为[SEQ，[]]，不包含任何旅行步。空选择迁移计划的形式为[SEL，[]]，不包含任何旅行步。空并行迁移计划的形式为[PAR，[]]，不包含任何旅行步。

在顺序迁移计划中增加一个旅行步[SEQ，[s_1，s_2，…，s_n]]是一个顺序迁移计划，包含 n 个旅行步。S 是一个旅行步，则[SEQ，[s_1，s_2，…，s_n，s]]也是一个顺序旅行计划，反之则是在顺序迁移计划中删除一个旅行步。

4.3 辅助方法

结合结构化动态更新迁移计划，下面给出几个辅助方法定义。

(1)状态转换格局：是一个三元组（*Location*，*state*，*scheme*），通常用 γ 表示。其中 *Location* 是“软件人”当前所在的主机；*state* 表示“软件人”所处状态，包括内部状态（所有“软件人”变量的值和可以访问的环境变量的值），一般用 δ 表示；*scheme* 表示当前的迁移计划结构。

(2)bool 函数：以 g 表示一个布尔方法，在迁移计划中可能是控制方法或入口卫士方法，γ 表示当前格局，则 bool(g，γ)表示在当前格局 γ 下计算方法 g 所得到的布尔值。

(3)exec 函数：以 m 表示一个入口方法，γ 表示当前格局，则 exec(m，γ)表示当前状态下执行方法 m，返回新的状态 δ'。

(4)entry 函数：以 m 表示一个入口方法，以 g 表示入口卫士方法，γ 表示当前格局，则 entry(g，m，γ)=(bool(g，γ)->exec(m，γ)，true->δ)，表示在当前格局 γ 下判断入口卫士方法 g。若 g 返回为真，则执行入口方法 m，返回新的状态 δ'，否则返回旧状态 δ。

(5)update 函数：以 t 表示一个迁移计划更新方法，γ 表示当前格局，则定义 update(t，γ)表示在当前格局下执行迁移计划更新方法，返回一个布尔值，说明是否顺利更新了迁移计划，使“软件人”可以开始下一步迁移。

(6)more 函数：以 s_i 表示一个顺序迁移计划，其中包含 n 个旅行步，每个旅行步的控制方法分别为 p_i，γ 表示当前格局，则定义

more(s_i，γ)=(bool(p_1，γ)->ture，bool(p_2，γ)->ture，…，bool(p_n，γ)->ture，true->false)

表示该迁移计划中是否有剩余的有效旅行步。

(7) left_iti 函数：以 s_i 表示一个顺序迁移计划，γ 表示当前格局，t 表示迁移计划更新方法，则定义 left_it$_i$(s_i，γ，t)=(more(s_i，γ)->si，true-> update(t，γ))。

表示“软件人”还有未处理的迁移计划。

4.4 各迁移计划的动态语义

(1)顺序迁移计划的动态语义：

if (n>0) then

[A_0，δ_0，[SEQ，[s_1，s_2，…，s_n]]]->

(bool(p_1，γ_0)->[A_1，entry(g_1，m_1，γ_0)，left_it$_i$($SInew$，(A_1，δ_{new}，SI_{new})，t_1)]，

True->[A_0，δ_0，SI_{new}])

注：SI_{new}=[SEQ，[s_2，…，s_n]]

δ_{new}= entry(g_1，m_1，γ_0)

(2) 选择迁移计划的动态语义

[A_0，δ_0，[SEL，[s_1，s_2，…，s_n]]]->

(bool(p_1，γ_0)->[A_1，entry(g_1，m_1，γ_0)，[SEL，[]]]，True->[A_0，δ_0，SI_{new}])

注：SI_{new}=[SEL，[s_2，…，s_n]]

(3) 并行迁移计划的动态语义

[A_0，δ_0，[PAR，[s_1，s_2，…，s_n]]]->

[bool(p_1，δ_0)->[A_1，δ_0，[SEQ，[S_1]]]，

bool(p_2，δ_0)->[A_2，δ_0，[SEQ，[S_2]]]，……

bool(p_n，δ_0)->[A_n，δ_0，[SEQ，[S_n]]]]

通过综合运用上述各种定义及 3 种基本迁移方式的动态语义，我们可以表达出各种不同的迁移现象，为编程实现奠定了理论基础。

结束语 使用工作流模式，可以让 SoftMan 在网络中独立迁移，并按部就班地完成自己的工作。对于一些逻辑结构复杂，用工作流单纯的序列结构难以表达或非常低效的任务，主从模式显然是一个很好的选择。主从模式以其特有的协作方式，使 SoftMan 各司其职，共同完成任务。当然，主从模式也有它的缺点：由于涉及到多个 SoftMan 以及它们之间的协调[8]，因此编程相对复杂，同时在工作时也有更大的通信开销。分体模式既具有工作流模式的某些特征，也具有主从模式的某些特征。因此，对于某些用工作流模式解决太低效，而用主从模式解决又显得小题大作的任务，用分体模式显然是一个很好的折中方法。可见，3 种模式结合使用，可以为完成任务提供更多、灵活的手段。

参 考 文 献

[1] 曾广平，涂序彦. “软件人”的概念模型与构造特征[J]. 计算机科学，2005，32(5)：135-136，143

[2] 葛占华，夏克俭，涂序彦. 基于 IBM Aglet 的软件人系统研究[J]. 微计算机信息，2006，22(8-2)：299-301，232

[3] Komiya T，Enokido T，Takizawa M. Mobile Agent model for *transaction processing on distributed objects*[J]. *Information* Sciences，2003，154(1/2)：23-38

[4] Baumann J，Hohl F，Roterhmel K. MOLE：A mobile agent system [J]. Software，2000，32(6)：575-603

[5] 刘大有，杨博，等. 基于旅行图的移动 Agent 迁移策略[J]. 计算机研究与发展，2003，40(6)：838-845

[6] 赵耀东，王成耀，曾广平，等. 软件人在迁移过程中的任务实现方式探讨[J]. 计算机应用研究，2006(1)：49-51

[7] 王洪泊，班晓娟，曾广平，等. 网络环境下虚拟机器人-“SoftMan”系统平台总体设计[J]. 计算机科学，2005，32(8)：152-154

[8] 曾广平，涂序彦，王洪泊. “软件人”及应用[M]. 科学出版社，2007