3E-2

Name Management Algorithm for Consistency Control

Fumiko KOUDA and Hidehiko TANAKA
Department of Electrical Engineering, University of Tokyo

1 Introduction

This paper describes brief certification of the algorithm which is made for name management processing in distributed environment. The algorithm makes decision of the direction of communication at the time when update occurs at the receiver's side after communication request is issued from the sender.

We have designed a name management system [1] which provides users with location transparency, and which hides inconsistency of information among name management systems. However, in [1], we have not mentioned certification of the algorithm.

The scope of the paper is a name management system for the processes concerning with inter-process communication and one-to-one relationship. Addressing is out of scope, because we do not consider it necessary from the users' point of view.

2 Modeling

2.1 Assumption

We assume several kinds of kernel processes in each host : interprocess communication process, name managing process (abbreviate to name manager) etc., and assume user processes which correspond to users.

Name manager has two phases : update and refer.

In the procedures of updating, registering or binding names, the name manager writes new values in some name tables. We call it **U-phase**.

In the procedures of exchanging messages or query on objects, the name-manager consults name tables and performs some work without changing values. We call it **R-phase**.

To achieve transparency to users or to hide inconsistency, we have tried to introduce concept of **naming domains**(abbreviate to **n.d.** or **n.d.s**).

We use four kinds of **naming domains** : **Und**, **Cnd**, **Pnd** and **Dnd**, which are used for users, for common communication control, for local system control and for describing of objects, respectively.

We combine them with concept of capability and construct a name management system to achieve transparency to users and to diminish failures. The sequence of name reference and translation in the **R-phase** and update of names in the **U-phase** when used distinguished names in these management tables is C-list, Und/Pnd, the mapping from Und/Pnd to Cnd, Cnd (in sender's host), Cnd (in receiver's host), the mapping from Cnd to Pnd/Und, Pnd/Und, and A-list[1].

2.2 Process Status

We assume that processes dynamically change. The status related to the name manager is observed :

- creating a process — **register** the name into a naming domain
- informing the presence — **binding** the names and objects by related naming domains.
- prepared with communication —**normal**
- updating a process
  1. process migration — **move** the host
  2. process deletion —**delete** the process

2.3 Abstract Model

Name tables work as *checkpoint* of transformation of names, so we abstract them and call each name table **CheckPoint of Name Status** (CPNS). Name tables are numbered from the mapping order, and call them CPNS-#n. The name status in the naming domains is described in the former section. We split the **move** into two, according to emigration or immigration of hosts. The status is, then **normal, empty, delete, moving, moved** or **binding**. Capability and status of the name mapping is assumed as **True** or **False**. We also call it the name status.

2.4 Update Situation

We concentrate on the problem that the relationship between communication and update processing in distributed environment.

If the update processing has done before the communication from the peer occurs, there is no problem. Because the communication obeys the new name-relations. Another case is that the communication reaches to the peer before update begins, there is no problem as well, because receiver can manage the decision of the next processing.

On the contrary, if communication occurs from the peer at the time of update processing, problems occur. Transmission delay causes the name status in *CPNSs* to be different. We will certify the case in the next section.

We make an indicator to state the direction : *GO* and *RETURN* to show which way to move a communication. *GO* means the communication is to go forward in the direction of receiver's host, and it is represented in the

model that a communication moves from one CPNS to the next. *RETURN* goes back to the sender's host and is represented as a step-move between two CPNSs.

The total model is constructed from the component of the U-phase, the R-phase, eight CPNSs ,and controlled some name status and direction indicator.

3 Algorithm and Certification

3.1 Algorithm

The direction-decision algorithm at each CPNS :

Input : CPNS number, current direction.

Output : If name state of the CPNS is **normal, moved, binding** or **True** then direction of communication is *GO* else if the name state of the CPNS is **moving, delete, empty** or **False** then the direction is *RETURN*.

Especially, the name status **moved** has two meanings. This status is used except in the 'former' host where the object resided.

In migrated host, **moved** means under the processing of movement. In the other hosts, the **moved** status is used for only in **Cnd**, indicating that the destination host to deliver messages has changed.

While the object is emigrating to the other, the name status **moving** is used in the 'former' host. If the name status in **Cnd** in the host is **moving**, the messages must be transmitted to the new immigrated host.

3.2 Certification

In this section we show that the algorithm can prevent from incorrect access. The model is based on the Fig.1.

Case : Migration of Objects

If the name manager in R-phase finds the updated name status in the host 3 : **moved**, then the name manager readdresses the communication to the host 2, according to the algorithm.

If the updated status is found in host 1 there are three cases :

a) if CPNS-#4 in host 1 is **moving**, then the name manager redirect the communication to the new host.

b) if CPNS-#4 is **normal** and CPNS-#2 is **moving**, then the communication has come to CPNS-#2, but the name manager returns it to CPNS-#4, and performs the same step as in **a**). Time interval during the move from CPNS-#2 to CPNS-#4 in host 1 is shorter than that of update move around the host 2 (there are more than 6 steps.). Therefore, the move from CPNS-#2 to CPNS-#4 is possible.

c) if CPNS-#2 and #4 is **normal**, but CPNS-#1 is **false**, then the communication has come to CPNS-#1, but the name manager returns the communication from CPNS-#1 to CPNS-#4 via CPNS-#2, and performs the same step as in **a**). This is possible from the same reason as in **b**).

At the third host, if CPNS-#4 in host 2 is **moved** or **normal**, the communication *GO*es, else if it shows **empty**, the communication is *RETURN*ed to the first host.

This is because that the U-phase arrives later than the R-phase in the host 2.

In the stable state, the model works well because **Und** supports transparency to users and name status in each CPNS shows **True** or **normal**, so that the direction is always *GO*.

4 Conclusion

The certification shows that the algorithm prevents from the wrong access of a communication during the U-phase.

Users may not notice the migration of the communication peer, if the U-phase processing is faster at the immigrated host than that of the R-phase. This shows that the algorithm brings users for the location transparency.

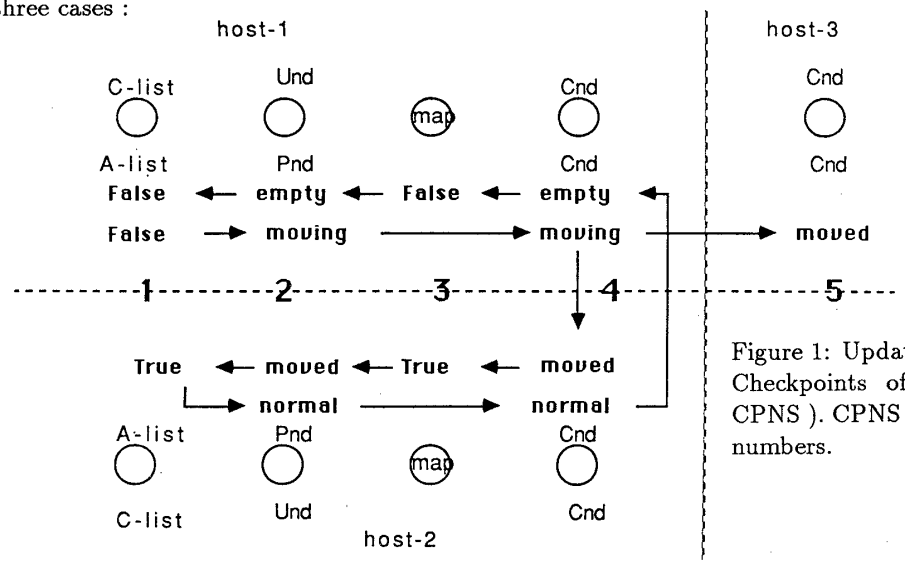

Figure 1: Update procedures and Checkpoints of name status (CPNS). CPNS are shown in bold numbers.

Reference

[1] F. KOUDA and H. TANAKA. The name management in distributed systems. In *3rd International Joint Workshop on Computer Communications*, pages 121–130, July 1988.