

***The realization of Work Management System based on
2-Dimensional Distributed Data Driven Architecture***

Kenshi Tanabe, Takashi Kon, Kazuhisa Akiyama, Kazuhide Takahashi†
and Makoto Jinguji‡

†Network Management Development Department

‡Reserch and Development Planning Department

NTTDoCoMo, Inc.

3-5 Hikari-no-oka, Yokosuka-shi, Kanagawa, 239-8536, Japan

TEL: +81-468-40-3310 FAX: +81-468-40-3784

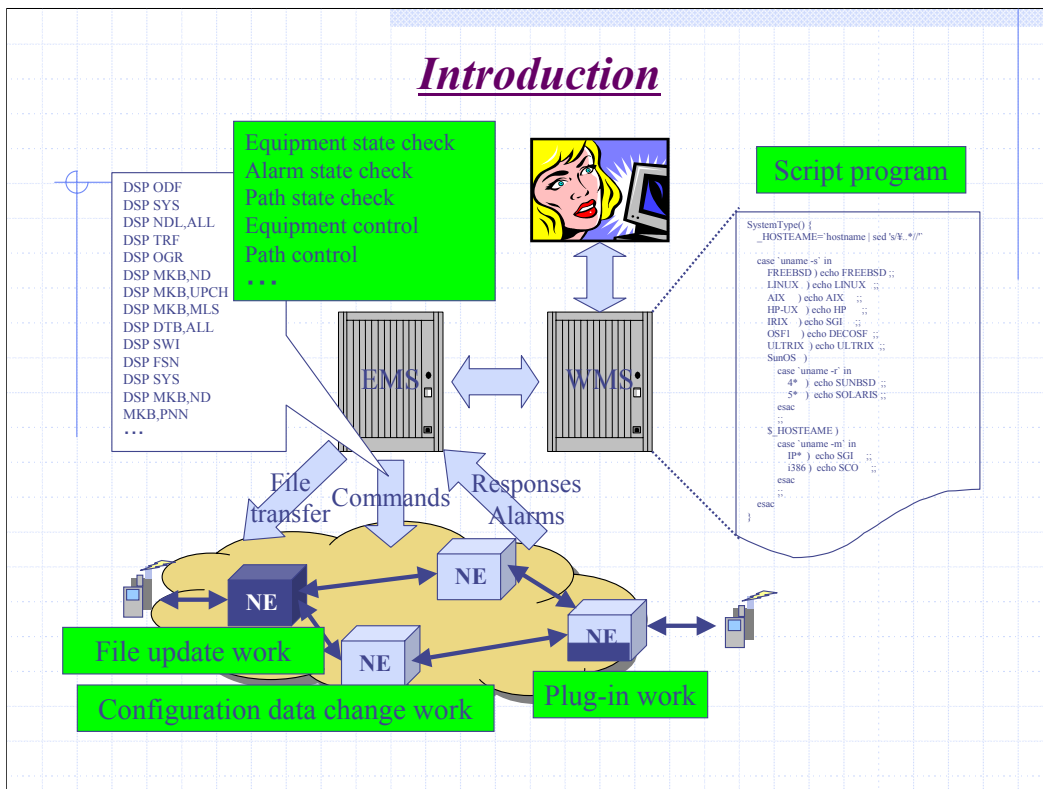
E-mail: {tanabek, kont, akiyamak, takahashikazu, jinguji}@nttodocomo.co.jp

Abstract

A mobile communications network is comprised of a huge number of various NE (Network Element) distributed geographically. A system that supervises messages (including alarms) from NEs and controls NEs by commands to detect failures that occur in the network, is called EMS (NE Management System). We developed an EMS based on the D3A (Distributed Data Driven Architecture) that enables communications carriers to drastically reduce EMS TCO (Total Costs of Ownership) and to change external specifications flexibly and rapidly.

A WMS (Work Management System) is a system that automates network management work such as plug-in work. When we implement a WMS based on D3A, to reduce work execution time is a significant challenge.

In this paper, we explain how we have enhanced D3A by introducing the new idea of “Dimension” to it. The enhanced architecture is called D4A (2-Dimensional Distributed Data Driven Architecture). We also describe the implementation configuration of the practical WMS and the results of performance evaluation.



1.Introduction

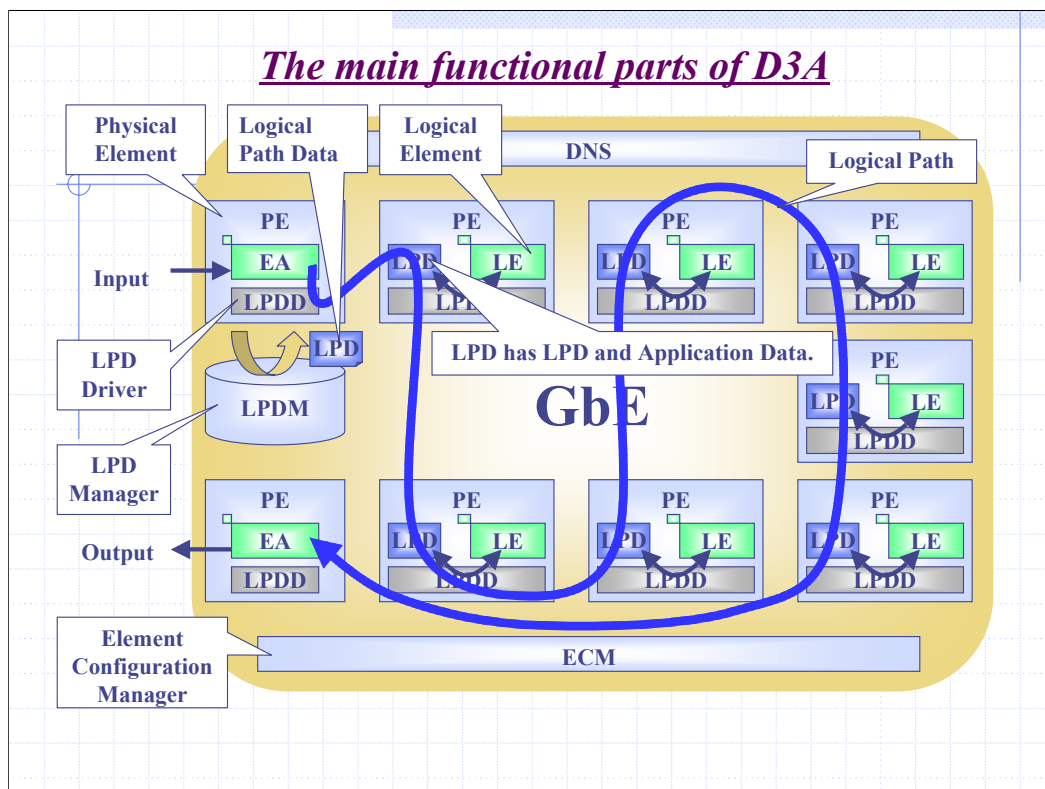
A mobile communications network consists of various network elements (NE: Network Element) distributed geographically. The work which maintains these NEs and the network is called network management work. In order to increase the efficiency of network management works, a large variety of OSS (Operation Support System) have been developed until now. [1,2]

Brief overview of the network management works such as plug-in work, file update work and configuration data change work are described as follows. Partially replacing NE software to fix bugs is a work known as plug-in work. A work to replace the entire NE software to add these functions is known as file update work. It is necessary to change configuration data of some NEs when a new NE is introduced, and this work is called configuration data change work. A mobile communications network needs numerous operations, in order to carry out these works to all NEs, since it consists of a huge number of NEs. We call a system which automated these network management works WMS (Work Management System). Moreover, we call a system which supervise alarms from NEs and control by commands EMS (NE Management System). A variety of WMS and EMS have been developed until now.[3,4]

However, procedures in these network management works are greatly influenced by NE specification changes. Furthermore, a new procedure should be added to improve operating effectiveness.

We have devised D3A (Distributed Data Driven Architecture)[5,6] that can reduce TCO (Total Costs of Ownership) of OSSs and put OSS in practical use. This architecture is comprised of a lot of software components called LEs (Logical Element) logically and a lot of IA servers called PE (Physical Element) physically. Furthermore, this architecture enables flexible and prompt specification change by expressing specification with XML. However, when implementation WMS based on this architecture, there is concern that work execution time increases since WMS applications that automate the works such as plug-in work need huge number of LEs.

In this paper, dimensional structure is introduced into D3A to reduce work execution time. This new architecture is called D4A (2-Dimensional Distributed Data Driven Architecture). Moreover, we explain the implementation structure of WMS that has been put into practical use based on the new architecture, and show the performance evaluation result.



2. Distributed Data Driven Architecture

In this section, overview of D3A is described. This architecture consists of following seven components: PE (Physical Element), LE (Logical Element), LPD (Logical Path Data), LPDD (LPD Driver), EA (External Adaptor), ECM (Element Configuration Manager).

In this architecture, software components that comprise OSS applications are called and hardware components are called PE. PE corresponds to a small-scale IA (Intel Architecture) server such as a rack-mounted type IA server or each blade server built in enclosure. In this paper, a small-scale IA server means a server that implements 1 or 2 CPUs. LE is a unit of parallel execution into which OSS applications are divided. A LE is assigned at a group of PEs that have redundant configuration. In other words, a LE is a unit of functional distribution and a PE is a unit of load distribution. Since a lot of PEs that implement CPUs having more than 3GHz clock frequency are combined with high-speed GbE (Giga bit Ethernet), this architecture enables OSSs to achieve high throughput.

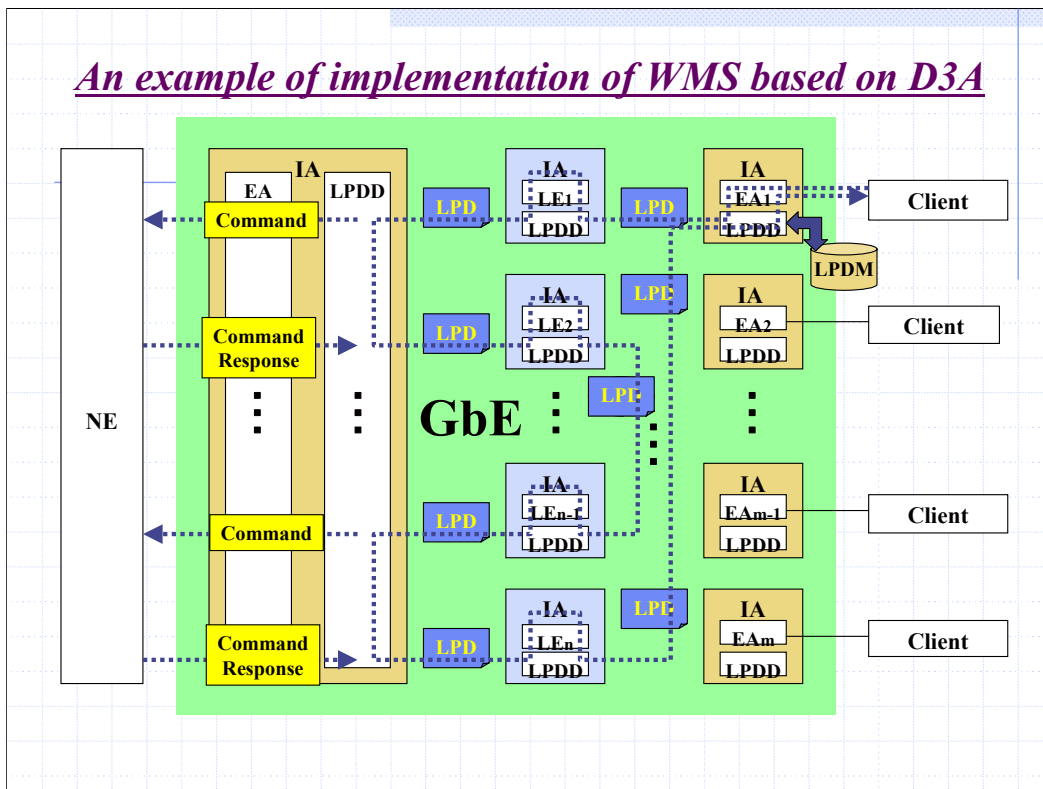
LPD (Logical Path Data) is described in XML and comprised of following two parts: a logic path data section and an application data section. When each LE receives a LPD from the previous LE, it extracts only the application data section from the LPD and processes it. Then, the LE sets the processed result to application data section and sends it to the next LE.

LPDD is a driver that controls logical path and physical path. It logically controls LPD between LEs based on logical path data section on LPD. It also physically controls LPD between PEs based on DNS. In other words, DNS is used for name resolution of LE. LPDD is the platform that controls redundant configuration and manages LEs/PEs failure in this architecture.

LPDM manages original LPD templates. Additionally, EA that fills a gap of between different interfaces and transform them into common interfaces (namely LPD) is located at the outer edge of this architecture. When data is input from external, LPDD on EA reads the original LPD templates (that has only logical path data section) from LPDM and generates application data section on the template.

ECM manages the original ECIF (Element Configuration Information File) that is described in XML. ECIF describes configuration information (relation between LE and PE, the redundant configuration methods, and so on). When a PCDD is initially activated, it read ECIF from ECM. When a PCDD resumes a LE from the failure according to the LE redundant configuration method, a PCDD refers to it. o refer to ECIF at a high-speed, they are cached in a LPDD of PE. When ECIF description is changed by reason of PEs configuration expansion or modification, the ECM deploys new one to all PEs.

An example of implementation of WMS based on D3A



3.Problem of D3A

In this section, the way to implement WMS based on D3A is examined and the problems of D3A are discussed. When D3A is applied to WMS, we are anxious about the next point. When the LEs that comprise WMS applications are distributed to IA servers, the number of the LEs that operate in a work attains to about 30000 at the maximum because granularity of the LEs is very small.

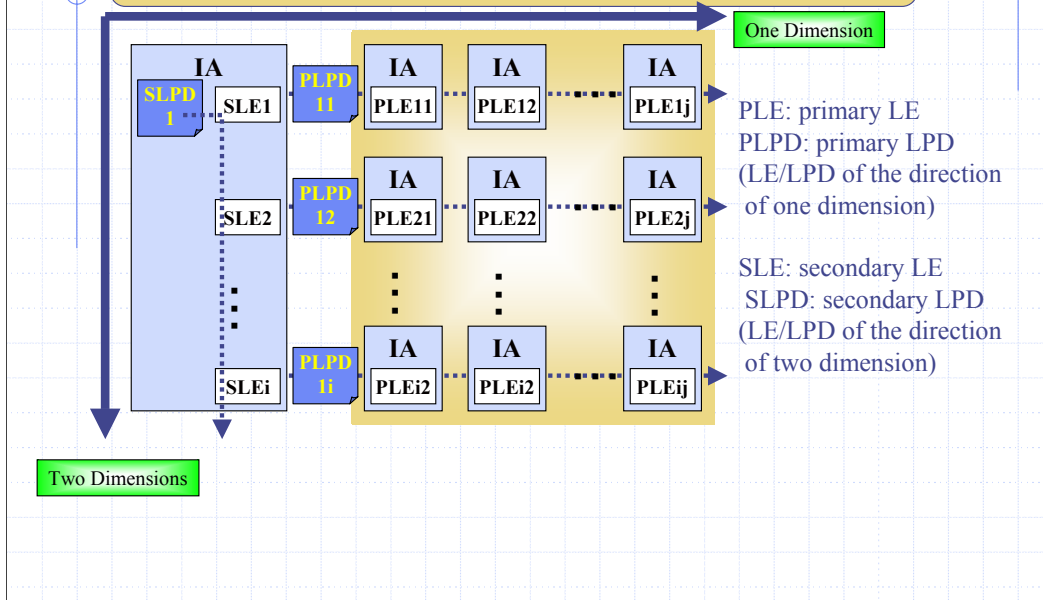
When the number of the LEs that transmit and receive LPD is set to about 30000, the size of LPD may amount to about 100M bytes, making it impossible to also ignore the amount of communications between IA servers. That is, in order to transmit and receive LPD between LEs through a network (GbE: Gigabit Ethernet), we are anxious about increase of the execution time of network management work.

Similarly, we are anxious about increase of work execution time due to the overhead generated by LPD, that surely goes via LPDD in communication between LEs, although the method of implementation LE in the same IA server is also possible.

Moreover, when the work procedure of WMS is divided into LEs, the amount of coding of LE is very small compared with that put in practical use in EMS. When carrying out distributed disposition of these small LEs at many IA servers, as mentioned above, we will see not only the deterioration of processing efficiency but the necessity of collectively managing very many LE(s) of various granularity. Moreover, D3A should be improved to reuse LEs finely so that we can enhance WMS application productivity.

A summary of D4A

The feature of D3A is to drive LE distributed in IA servers
By carrying about LPD between LEs.



4. 2-Dimensional Distributed Data Driven Architecture

The essence of D3A is to drive LEs distributed on IA servers by moving around LPD between LEs at a high-speed. The set of LEs in D3A is modeled as x-coordinate axis in 2-dimensional space that spreads among IA servers. Granularity of these LEs is large. They are called PLEs (Primary LEs). A LPD that drives these PLEs are called PLPD (Primary LPD). In order to shorten work execution time, LEs that has small granularity are allocated in the same IA server so that communication time between IA servers is reduced. A set of LEs allocated in the same IA server is modeled as y-coordinate axis in 2 -dimensional space that spreads within an IA server. These LEs are called SLEs (Secondary LEs). A LPD that drives these SLEs are called SLPD (Secondary LPD). This architecture called D4A (2-Dimensional Distributed Data Driven Architecture). PLEs comprise EMS applications and SLEs comprise WMS applications. SLE generates PLPD automatically and dynamically.

The feature of D4A

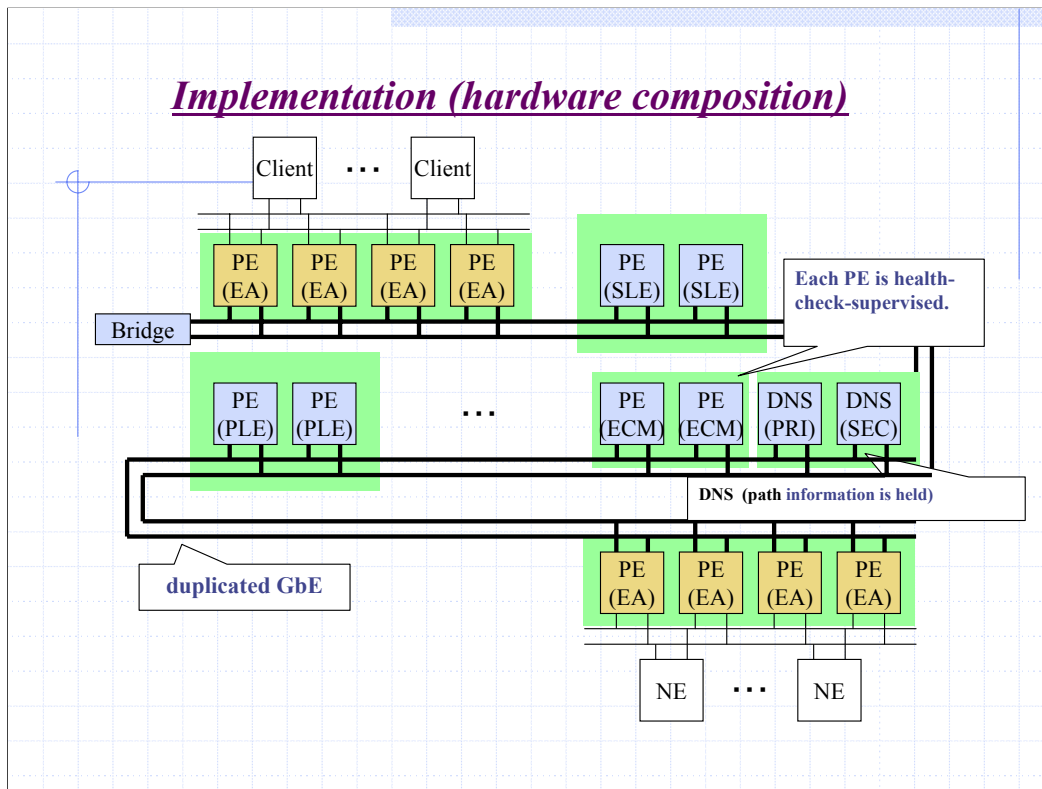
The features inherited from D3A

- **Drastic reduction of TCO**
 - Adoption of IA server
 - Positive adoption of an open source (middleware)
- **A functional addition is flexibly possible.**
- **Release from vendor lock-in**

The features peculiar to D4A

- **Work efficiency improves as a small LE group is arranged in the same IA server.**
- **Program appropriation is enabled and an additional change of the new work procedure can be made flexibly and quickly.**

The features that D4A inherits from D3A include followings: flexible and prompt addition of functions, release from vendor lock-in, serious TCO reduction by adopting IA servers, open source software and XML. Then, the features peculiar to D4A are described as follows. A SLE generates a PLCD as needed and the PLCD autonomously ceases when it finishes the processing. Since many factors may change network management work procedures and they may be determined dynamically, SLEs must be designed to change easily. Consequently, a SLPD is designed to express execution sequence of SLE as the work procedures. When the work procedures must be changed dynamically, a SLE autonomously generates a SLCD.

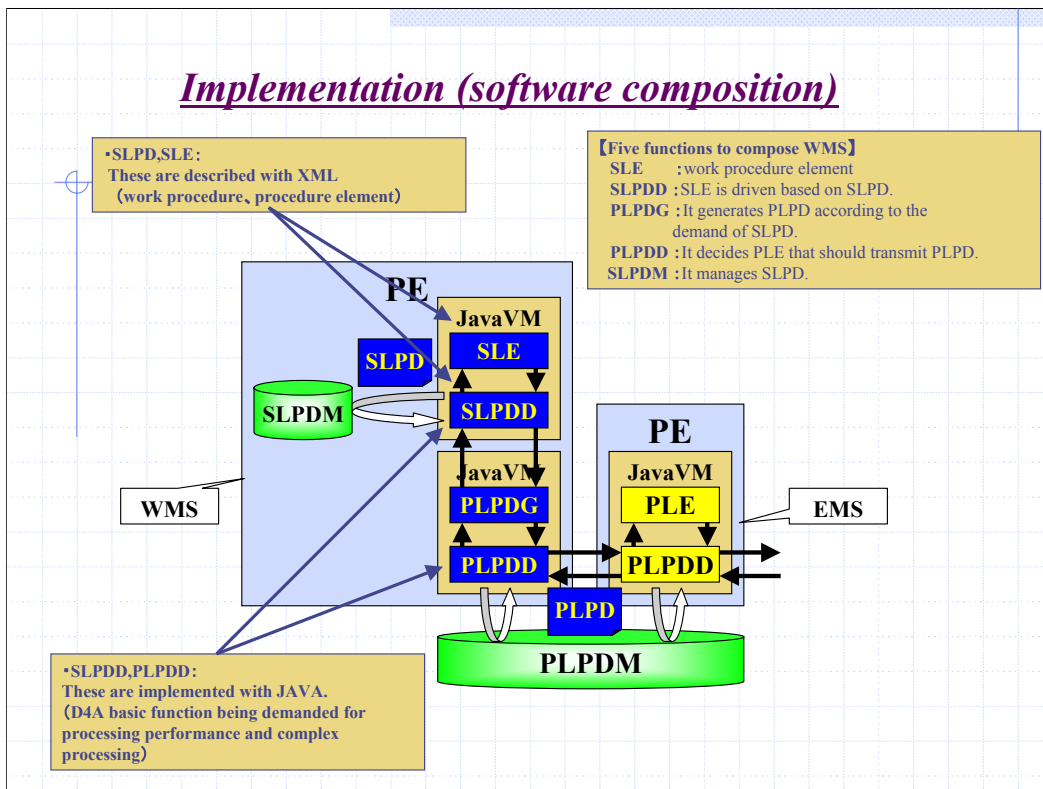


5.Implementation

We have put a WMS and an EMS in practical use based on D4A. This section explains hardware and software configuration of the WMS and the EMS.

5.1 Hardware Composition

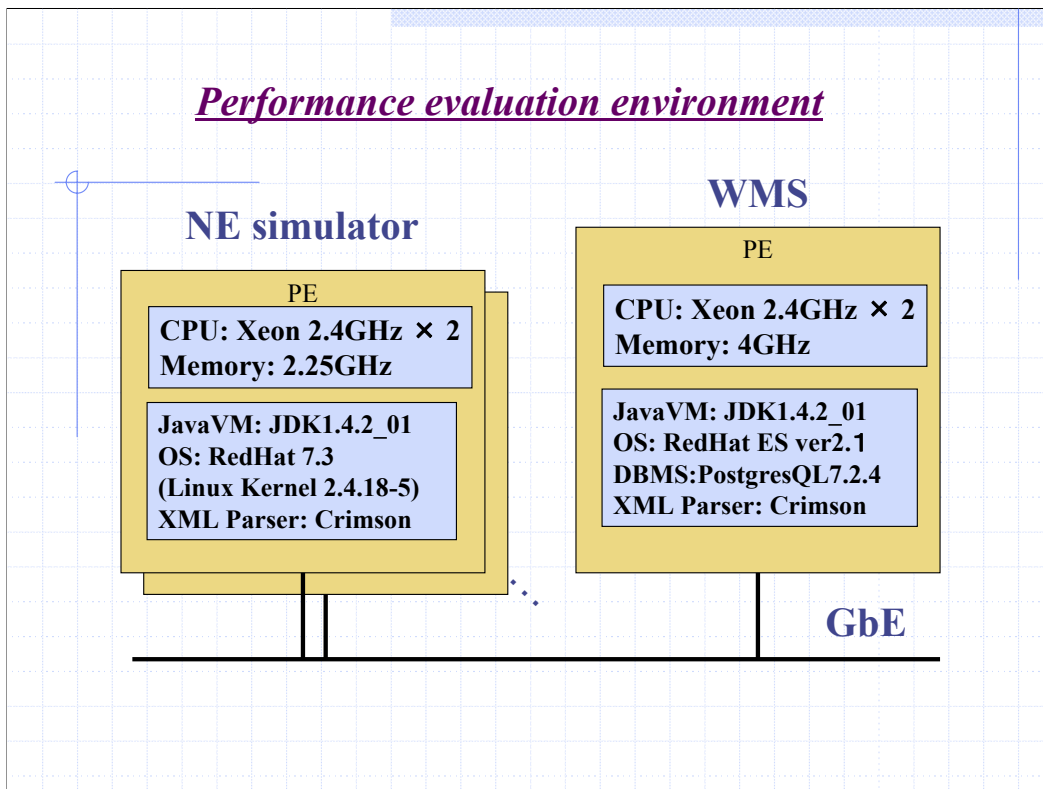
A unit of hardware that implements PLE and SLE is called PE (Physical Element) as well as that of D3A. A PLE is allocated at a PE that comprises the EMS. SLEs are allocated at a PE that corresponds to the WMS. With regards to SLE, all SLEs required for a work are allocated at one PE. The n-ACT/m-SBY configuration is applied to the WMS. The ECM (Element Configuration Manager) supervises all PEs by health check as well as D3A. Additionally, in order to transmit and receive about 10 Kbytes of PLPD frequently between PLEs, high-speed GbE (Giga bit Ethernet) is used for a network that combines PEs. Moreover, duplicated GbE is used in order to provide network reliability.



5.2 Software Composition

The software configuration of the completed WMS is shown above. The WMS is comprised of the following six functions: SLE, SLPDD (SLPD Driver), PLPDG (PLPD Generator), PLPDD (PLPD Driver), PLPDM (PLPD Manager) and SLPDM (SLPD Manager). The SLPDD that determines the next SLE that it should call by analyzing the syntax of SLPD described in XML. PLPDG generates PLPD according to the demand of SLE. The PLPDD determines the next PLE to which it should transmit PLPD by analyzing the syntax of PLPD described in XML. The SLPDM manages SLPDs. A SLPDD drives SLEs based on a SLPD.

Since SLPDD and PLPDG are the functions for which much throughput and complicated processing are required, it is difficult to code SLPDD and PLPDG with an interpreter language. Then, we decided to code these functions with a compile language (Java). However, these functions are the fundamental functions of D4A, for which a functional addition does not tend to occur after initial development. Moreover, as for SLPDM, PostgreSQL (which is open source software DBMS) was adopted from a viewpoint of restriction of initial cost and maintenance cost. SLPD and SLE respectively correspond to the work procedure and procedure element that are specifications of the WMS. Although these functions should require a functional addition frequently even after their initial development, data-level description in XML for them enables the WMS specifications to add easily.



6. Performance evaluation result

6.1 Performance evaluation item and its purpose

Since XML is used abundantly in D4A, the parse processing that loads heavily must be evaluated. The degree of execution multiplex of work, such as file update work, influences the increase in efficiency of network management work. Therefore, the maximum of the degree of execution multiplex of each work should be clarified. Moreover, since D4A is implemented on JavaVM, it is indispensable for this architecture to conduct a tuning of a garbage collection. For the above-mentioned reasons, we have decided to measure the size of the heap area of JavaVM, and the generating frequency of a full garbage collection, in order to clarify the optimal heap area size.

6.2 Performance evaluation method

With regard to file updating work, time change of the CPU usage rate and the memory usage rate of a heap area are measured from an operating start to an operating end, increasing the degree of execution multiplex (namely, number of NE) from 1 to 200.

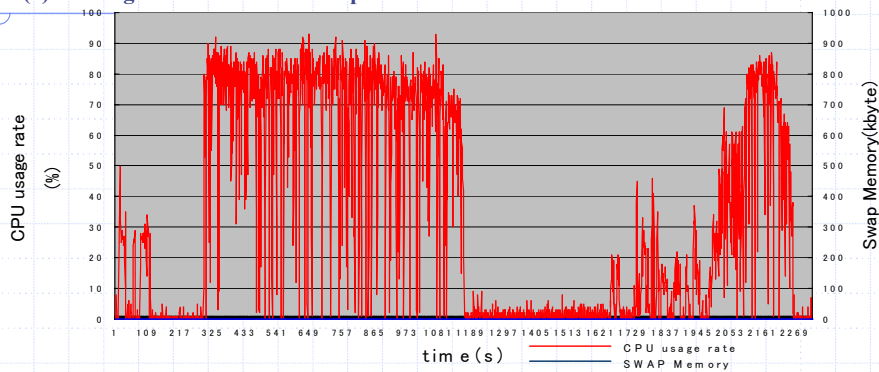
6.3 Performance evaluation environment

The performance evaluation environment is shown above. An IA server which carries two Intel Xeon 32bitCPUs are used for the hardware of the completed WMS. As for the software of the WMS, Linux ES Ver2.1 is used as an OS, J2SE 1.4.2_01 as a Java execution environment, and PostgresQL7.2.4 as a DBMS. Crimson is used as an XML purser. The hyper threading function of Xeon processor was enabled. An IA server which carries two Intel Xeon 32bitCPUs is used for the hardware of the EMS. As for the EMS software, Linux 7.3 OS is used, and J2SE 1.4.2_01 is used as a Java execution environment. Crimson is used as an XML purser. The same hardware as the WMS is used for the NE simulator.

Work execution multiplicity

(I) The degree of execution multiplex of work

The degree of work execution multiplex = 200



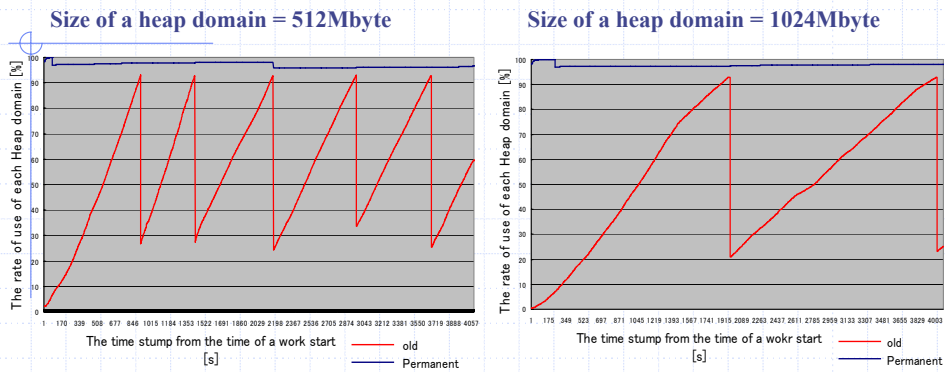
The number of commands / NE	The degree of work execution multiplex	work execution time	When in-series execution is carried out
249	1	16 minutes and 9 seconds	
	200	33 minutes and 14 seconds	

6.4 Performance evaluation result

6.4.1 The degree of execution multiplex of work

We show the time change of CPU usage rate and swap memory use with the degree of execution multiplex of work of 200. Because it tells us that the changes in the CPU and swap memory usage have been stable, it clearly shows that no bottleneck was generated in the hardware resources up to the execution multiplex of 200. The 200 or more degrees of execution multiplex were not able to be carried out due to the shortage of throughput of NE simulator. The result shows that it took 16 minutes and 9 seconds and 33 minutes 14 seconds to complete the whole work with the degrees of execution multiplex of 1 and 200, respectively. When 200 NEs were implemented in series under the above-mentioned condition, the time to perform the work would take about 54 hours, since each of the 200 NEs require 16 minutes 9 seconds for completion of the work as opposed to 33 minutes and 14 seconds for the same 200 NEs with the execution multiplex of 200. That means that we can achieve 98 % reduction of the work execution time by implementing the multiple work execution scheme.

Garbage collection frequency and time of JavaVM



Heap area size	Frequency	Average time
512Mbyte	1/800[s]	1.3[s]
1024Mbyte	1/2000[s]	2.6[s]

Recommendation size

6.4.2 Full garbage collection frequency and average time

We show the measurements of full garbage collection frequency and its average time for 512Mbyte and 1024Mbyte heap areas with the degree of work execution multiplex of 100. Moreover, we show the time changes of memory use rate during the file update work when the heap area size are set to 512 Mbytes and 1024Mbytes.

The result demonstrates that as the heap area increases, the frequency of full garbage collection falls, but the time required to complete one round of full garbage collection increases. However, even that long full garbage collection would not cause any trouble to our business since its duration is not so significant in comparison with the whole operating time. According to this measurement result, we can draw a conclusion that 1024Mbyte is a recommendation value for the heap area.

Conclusion

(1) Performance verification aimed at expanding the number of executions, envisaging future application to larger-scale networks.

(2) SLE standardization for more efficient design and manufacture in view of the likely presence of a great variety of SLEs due to different work procedures for every NE.

7. Consideration

A communication network today consists of various NEs. In WMS, the number of work multiplex execution impacts work efficiency directly as the preceding chapter also described. Therefore, if we envisage its application to larger-scale networks, performance verification aimed at further increasing the number of execution is a future subject. Moreover, since a work procedure is different for every NE, there is a possibility that various SLEs exist. Communalization of SLE, aiming at improving design and manufacturing efficiency, is also one of our future subjects.

8. Conclusion

In this paper We described D4A that was the model that improve D3A. As stated in Chapter 1, D4A was applied to the WMS (Work Management System), which automates work, and fundamental performance evaluation was performed to it. It has been clearly shown that this WMS can fully satisfy the requirement as a commercial system. The commercial experiment of the WMS was started in June, 2004, and it started to provide commercial services in September in the same year. The system has continuously performed stable operations ever since.

Reference

- [1] M. Ohnuki, N. Tanigawa, K. Hara, K. Terunuma, etc., "Special Issue on Operation System", DoCoMo Technical Journal, vol. 8 No.1, pp6-30, Apr.2000.
- [2] H. Sakuramoto, "Traffic Control System for I-mode Service Network to Enable Flexibility in Changing the Congestion Control Algorithm", APNOMS2002 Proceedings, pp.452-463, September. 2002.
- [3] T. Watanabe, "Network Element Facility Design System with Knowledge Database for IMT-2000 Network", APNOMS2002 Proceedings, pp.452-463, September.2002.
- [4] K. Takahashi, "Creation of Mobile Communication Network Provisioning Systems Based on an Organization Hierarchy Agent Model", JAWS2002, pp.289-296, November. 2002.
- [5] T. Shono, "The Proposal of the NE-OSS Constructing Method by Distributed Data Driven Architecture", IEICE, B-6-102, Mar.2003
- [6] I.Kaneko, "Reliability Evaluation of Distributed Data Driven Architecture", vol, 104, No.164, TM2004-25, July.2004