行政院及所屬各機關出國報告
（出國類別：實習）

# 參加Wireless IN+IP新服務開發系統平台技術實習

服務機關：中華電信研究所
出國人　職　　稱：研究員、助理研究員
　　姓　　名：韋增炫、陳明惠
出國地點：美國
出國期間：91年10月30日至91年11月10日
報告日期：92年01月10日

系統識別號:C09200569

# 公 務 出 國 報 告 提 要

頁數: 19　含附件: 否

報告名稱:

參加Wireless IN+IP 新服務開發系統平台技術實習

主辦機關:

中華電信研究所

聯絡人／電話:

楊學文／03-4244218

出國人員:

韋增炫　中華電信研究所　91820專案研究計畫　研究員
陳明惠　中華電信研究所　91820專案研究計畫　助理研究員

出國類別: 實習

出國地區: 美國

出國期間: 民國 91 年 10 月 30 日 -民國 91 年 11 月 10 日

報告日期: 民國 92 年 01 月 10 日

分類號/目: H6／電信　／

關鍵詞: Wireless,IN+IP,新服務,平台技術

內容摘要: 職等二人奉派赴美國HP公司參加通訊秘書服務開發平台設備採購案R910234-1之合約：Wireless IN+IP新服務開發系統平台技術之實習，自民國九十一年十月三十日至民國九十一年十一月十日爲期含行程共十二天。實習內容包含：（1）Internetworking Design （2）Enterprise DBA：Architecture and Administration （3）Enterprise DBA： Performance and Tuning

本文電子檔已上傳至出國報告資訊網

# 摘　　要

職等二人奉派赴美國HP公司參加**通訊秘書服務開發平台設備**採購案R910234-1之合約：Wireless IN+IP新服務開發系統平台技術之實習，自民國九十一年十月三十日至民國九十一年十一月十日為期含行程共十二天。

實習內容包含：

（1）Internetworking Design

（2）Enterprise DBA： Architecture and Administration

（3）Enterprise DBA： Performance and Tuning

# 目　　錄

頁次

## 1、目的

職等二人奉派赴美國HP公司參加Wireless IN+IP新服務開發系統平台技術之實習，自民國九十一年十月三十日至民國九十一年十一月十日，含行程共十二天。實習期間在美國洛杉磯接受叢集式資料庫伺服器之操作與系統管理、網路設備之Internetworking Design、叢集式物件關聯式資料庫軟體之Performance and Tuning...等技術。

參加Internetworking Design、Enterprise DBA：Architecture and Administration、Enterprise DBA： Performance and Tuning訓練，將增進所需之資料庫與Internetworking技術，對本所開發有關Wireless IN+IP加值服務助益良多；並提昇本所研發之上線營運系統管理功能。

本案採購之設備系統功能複雜，具資料庫之規劃設計、網路之規劃設計及效能調整等重要功能；另本設備未來將繼續由本所增添新的應用程式，以支援營運單位。為充分有效利用本系統，使工作之順利進行，乃是此實習之最主要目的。

## 2、行程概要

整個行程從十月三十日出發，至十一月十日返國，共計十二天。此次赴美國實習的地點為洛杉磯HP公司，有關出國行程如下所示：

| 日　期 | 概　要 |
|---|---|
| 10月30日 | 中正機場搭機抵達美國洛杉磯機場 |
| 10月31日<br>至<br>11月01日 | 1. Internetworking Design<br>2. Enterprise DBA： Performance and Tuning |
| 11月02日<br>至<br>11月03日 | 整理資料( 星期六、日 ) |
| 11月04日<br>至<br>11月08日 | 1. Enterprise DBA： Architecture and Administration<br>2. Enterprise DBA： Performance and Tuning |
| 11月09日 | 於洛杉磯機場返國 |
| 11月10日 | 抵達中正機場 |

## 3、實習內容

實習內容分為以下部份：

（1）Internetworking Design

（2）Enterprise DBA： Architecture and Administration

（3）Enterprise DBA： Performance and Tuning

以下分別就各部份加以敘述。

### 3.1、Internetworking Design

Internetworking is the communication between two or more networks. It encompasses every aspect of connecting computers together. Internetworks have grown to support vastly disparate end-system communication requirements. An internetwork requires many protocols and features to permit scalability and manageability without constant manual intervention. Large internetworks can consist of the following three distinct components:

- Campus networks, which consist of locally connected users in a building or group of buildings
- Wide-area networks (WANs), which connect campuses together
- Remote connections, which link branch offices and single users ( mobile users and/or telecommuters ) to a local campus or the Internet

Figure 1 depicts a typical enterprise internetwork.

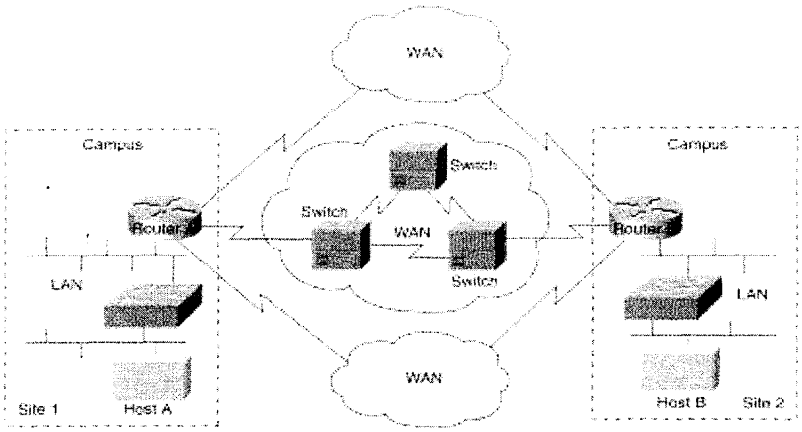


Figure 1. A typical enterprise internetwork

Designing an internetwork can be a challenging task. To design reliable, scalable internetworks, network designers must realize that each of the three major components of an internetwork have distinct design requirements. An internetwork that consists of only 50 meshed routing nodes can pose complex problems that lead to unpredictable results. Attempting to optimize internetworks that feature thousands of nodes can pose even more complex problems.

Despite improvements in equipment performance and media capabilities, internetwork design is becoming more difficult. The trend is toward increasingly complex environments involving multiple media, multiple protocols, and interconnection to networks outside any single organization's dominion of control. Carefully designing internetworks can reduce the hardships associated with growth as a networking environment evolves.

### 3.1.1 、Internetworking Devices

Designing an internetwork have four basic types of internetworking devices as follows :

- Hubs (concentrators)

  Hubs (concentrators) are used to connect multiple users to a single physical device, which connects to the network. Hubs and concentrators act as repeaters by regenerating the signal as it passes through them.

- Bridges

  Bridges are used to logically separate network segments within the same network. They operate at the OSI data link layer (Layer 2) and are independent of higher-layer protocols.

- Switches

  Switches are similar to bridges but usually have more ports. Switches provide a unique network segment on each port, thereby separating collision domains. Today, network designers are replacing hubs in their wiring closets with switches to increase their network performance and bandwidth while protecting their existing wiring investments.

- Routers

  Routers separate broadcast domains and are used to connect different networks. Routers direct network traffic based on the destination network layer address (Layer 3) rather than the workstation data link layer or MAC address. Routers are protocol dependent.

Data communications experts generally agree that network designers are moving away from bridges and concentrators and primarily using switches and routers to build internetworks.

### 3.1.2 、Determining the Internetworking Requirements

Designing an internetwork can be a challenging task. The first step is to understand your internetworking requirements, then select internetwork capability and reliability options that meet these requirements.

Internetworking devices must reflect the goals, characteristics, and policies of the organizations in which they operate. Two primary goals drive internetworking design and implementation:

- *Application availability*—Networks carry application information between computers. If the applications are not available to network users, the network is not doing its job.
- *Cost of ownership*—Information system (IS) budgets today often run in the millions of dollars. As large organizations increasingly rely on electronic data for managing business activities, the associated costs of computing resources will continue to rise.

A well-designed internetwork can help to balance these objectives. When properly implemented, the network infrastructure can optimize application availability and allow the cost-effective use of existing network resources.

### 3.1.3 、Identifying and Selecting Internetworking Capabilities

After understanding your internetworking requirements, you must identify and then select the specific capabilities that fit your computing environment. The following provides a starting point for making these decisions :

- Identifying and Selecting an Internetworking Model

  Hierarchical models for internetwork design allow you to design internetworks in layers. To understand the importance of layering, consider the Open System Interconnection (OSI) reference model, which is a layered model for understanding and implementing computer communications. By using layers, the OSI model simplifies the task required for two computers to communicate. Hierarchical models for internetwork design also uses layers to simplify the task required for internetworking. Each layer can be focused on specific functions, thereby allowing the networking designer to choose the right systems and features for the layer.

  Using a hierarchical design can facilitate changes. Modularity in network design allows you to create design elements that can be replicated as the network grows. As each element in the network design requires change, the cost and complexity of making the upgrade is constrained to a small subset of the overall network. In large flat or meshed network architectures, changes tend to impact a large number of systems. Improved fault isolation is also facilitated by modular structuring of the network into small, easy-to-understand elements. Network mangers can easily understand the transition points in the network, which helps identify failure points.

- Choosing Internetworking Reliability Options

  One of the first concerns of most network designers is to determine the required level of application availability. In general, this key consideration is balanced against implementation cost. For most organizations, the cost of making a network completely fault tolerant is prohibitive. Determining the appropriate level of fault tolerance to be included in a network and where redundancy should be used is not trivial.

## 3.2 、Enterprise DBA ： Architecture and Administration

Database administrators are responsible for maintaining the Database Server so that the server can process user requests. An understanding of the Database architecture is necessary to maintain it effectively. The Database architecture and administrator tasks are as follows :

- Planning and creating databases
- Managing database availability
- Managing memory, physical and logical structures
- Managing users and privileges

### 3.2.1 、Planning and creating databases

The general purpose of a database is to store and retrieve related information. An Oracle database has a logical and a physical structure as shown in figure 2. The physical structure of a database is the set of operation system files in the database. An Oracle database consists of three file types such as :

- Data files containing the actual data in the database
- Redo logs containing a record of changes made to the database to enable recovery of the data in case of failures
- Control files containing information necessary to maintain and verify database integrity

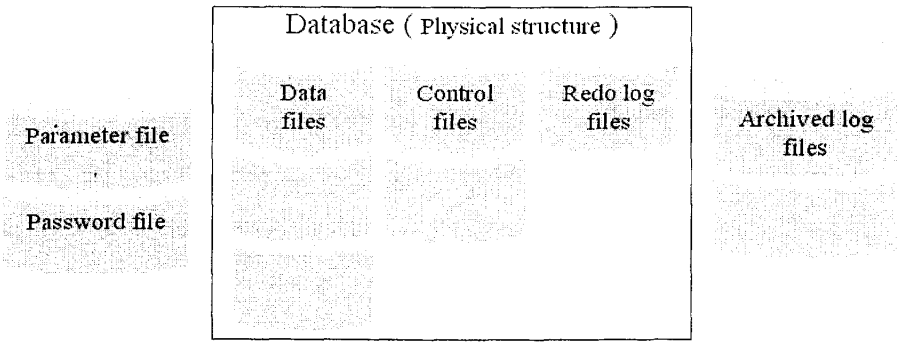


Figure 2. An Oracle Database

An important issue during installation and creation of a database is organizing the file system so that it is easy to administer growth by adding data into an existing database, adding users, creating new databases, adding hardware and distributing I/O load cross sufficiently many drives.

Depending on whether you want to administer your database locally on the same machine on which the database resides or to administer many different database servers from a single remote client, you can choose either operating system authentication or password files to authenticate data administrators.

Creating the database is the first step in managing and organizing a database system. Database creation is a task that prepares several operating system files and is needed only once no matter how many data files the database has. This is a very important task, because you must decide on database settings.

Before creating the database, calculate the necessary disk space for the database, including online redo log files, control files, and data files.

Plan how to protect the database, including the online redo log files, control files, archived redo log files, and provide a backup strategy. For the sake of safety, you should create a least two control files on two different disks. The online redo log files of a database should consist of multiplexed groups of online redo log files. A group of log files consists of identical copies, which should be located on different disks.

To minimize fragmentation of the database, you should separate database objects with different life spans, such as application data and temporary data, into different tablespaces. To ensure well-balanced I/O loads, you should separate objects with competing I/O requirements, such as tables and indexes, into different tablespaces.

After the database is created, the instance is left running and the database is open and available for normal use.

### 3.2.2 、Managing database availability

Availability is ensuring that the data is accessible when your user requests it. So there is no time for downtime. High availability means that the data stored in your database is available and accessible to your applications regardless of

events such as planned backups or disasters.

Providing a high availability database server is neither simple nor inexpensive. The challenge to meet very demanding availability requirements is to understand what may go wrong, to examine the expected costs downtime associated with these problems, and to compare those costs against the costs of various solutions to minimize downtime associated with these problems.

To achieve high availability, a system needs to have as few outages as possible and fast recovery when outages do occur. A highly available system typically has redundant hardware and software to avoid having single points of failure. The failover process moves the work performed by the failed component to a backup component. Damaged resources are repaired or replaced. Failed or partially failed transactions are recovered. The more transparent the failover is to users, the more high available is the system.

Downtime can be either planned or unplanned. They can be broadly classified as system faults, data and media errors, and site outages. Unplanned downtime is disruptive because of the lack of predictability of its timing. Plan downtime can be just as disruptive to operations, especially in global enterprises that support users in multiple time zones up to 24 hours a day.

The key to building a high availability solution is to consider all these causes of downtime, and design a solution that address them.

### 3.2.3 、Managing memory, physical and logical structures

- Managing memory and storage

  Memory is a critical system resource which has a significant impact on the overall performance of the database. Database administrators, therefore, closely monitor system memory utilization to ensure its most optimal use. Continuing its quest to make the management of the databases simple.

  Database administrators can change the size of the buffer cache and the shared pool without having to restart the instance.

  Managing tablespaces and physical datafiles enables administrators to always ensure sufficient space and that data can be committed or undone efficiently. At the same time, if recovery is required, it ensures appropriate and effective backup and recovery structures.

Storage management also includes redo logs, rollback segments or archived logs details and options to create or modify their properties.

- Managing physical and logical structures

  As shown in figure 2, the physical structure of a database includes the control files, redo log files, and the data files that make up the database.

  The control file is a small binary file necessary for the database to start and operate successfully. Each control file is associated with only one database. Before a database is opened, the control file is read to determine if the database is in a valid state to use. A control file is updated continuously by the server during database use, so it must be available for writing whenever the database is open. The information in the control file can be modified only by the server, no database administrator or end user can edit the control file. If for some reason the control file is not accessible, the database does not function properly. If all copies of a database's control files are lost, the database must be recovered before it can be opened.

  Redo log files provide the means to redo transactions in the event of a database failure. Every transaction is written synchronously to the redo log files in order to provide a recovery mechanism in case of media failure. This includes transactions that have not yet been committed, undo segment information, and schema and object management statements. Redo log files are used in a situation such as an instance failure to recover committed data that has not been written to the data files. The redo log files are used only for recovery.

  A small database might need only the SYSTEM tablespace. Each tablespace consists of one or more operating system files, which are called data files. It is recommends that you create additional tablespaces to store user data, user indexes, undo segments, and temporary segments separate from data dictionary. This gives you more flexibility in various database administration operations and reduces contention among dictionary objects and schema objects for the same data files.

  The DBA can create new tablespaces, resize data files, add datafiles to tablespaces, set and alter default segment storage settings for segments created in a tablespace, make a tablespace read-only or read-write, make a tablespace temporary or permanent, and drop tablespaces.

### 3.2.4 、Managing users and privileges

The database administrator defines the names of the users allowed to access a database. A security domain defines the settings that apply to the user.

A user who needs to access to the database can be authenticated by one of the following :

- Data Dictionary
- Operating system
- Network

The means of authentication is specified at the time the user is defined in the database and can be altered later.

A privilege is a right to execute a particular type of SQL statement or to access another user's object. These include the right to :

- Connect to a database
- Create a table
- Select rows from another user's table
- Execute another user's stored procedure

Each system privilege allows a user to perform a particular database operation or class of database operations. For example, the privilege to create tablespaces is a system privilege.

Each object privilege allows a user to perform a particular action on a specific object, such as a table, view, sequence, procedure, function, or package.

A DBA's control of privilege includes :

- Providing a user the right to perform a type of operation
- Granting and revoking access to perform system functions
- Granting privileges directly to users or to roles
- Granting privileges to all users (PUBLIC)

### 3.3、Enterprise DBA： Performance and Tuning

The best practice of tuning is careful design of systems and applications, and the majority of performance gains are realized by tuning the application.

It is much likely to run into performance problems if :

- The hardware can meet user demands
- The database was carefully designed
- The application developers write efficient SQL programs

If wrong decision were made early, or if users expect much more from the system now than they did previously, it should seriously consider further tuning to improve performance. The database should be monitored on a regular basis to look for bottlenecks that affect performance.

There are basically two forms of tuning :

- Speed : short response time
- High throughput scalability : higher load at a comparable response time or throughput.

The result of tuning should be visible to users, either as a decrease in the time it takes to perform a task, or as an increase in the number of concurrent sessions.

Tuning is performed because either a problem already exists or the DBA wishes to prevent problems from occurring. Some examples of items to be monitored are critical table growth, changes in the statements users excute, and I/O distribution across devices.

The course discusses methods to determine where waits and bottlenecks exist, and how to resolve these problems.

#### 3.3.1、Tuning Phases

Tuning 可分為以下幾各階段( Phase )：

- Application design and programming
  Whenever possible, tuning should start at this level. With a good

design, many tuning problems do not occur. For example, although it is normally good practice to fully normalized tables to reduce redundancy, this may result in a high number of table joins. By de-normalizing the performance of the application may improve dramatically.

- Database configuration
  It is important to monitor hot spots, even on fast disks, it should plan the data configuration in order to enable faster recovery times, and faster data access.

- Adding a new application
  When adding a new application to an existing system, the workload changes. Any major change in the workload should be accompanied by performance monitoring.

- Troubleshooting and tuning
  This is the methodology recommended for production database. It consists of looking for bottlenecks, and resolving them. Use tools to identify performance problems, By examining this data form a hypothesis of what is causing the bottleneck. From the hypothesis develop a solution, and implement it. Run a test load against the database to determine if the performance problem has been resolved.

### 3.3.2、Tuning Goals

Tuning Goals可分以下幾項：

- Reducing or eliminating waits
- Accessing the least number of blocks
- Caching blocks in memory
- Response time
- Throughput
- Load
- Recovery time

The goal in tuning a database is to make sure that users get responses to their statements as quickly as possible. Because "as quickly as possible"

is an ambiguous term, the time measure must be quantified in some manner. The goal is usually measured in terms of response time, throughput, load, or recovery time.

Tuning goals can arise due to a Service Level Agreement. For example, Process A must complete within a specified time period, or a certain number of transactions per second have to be processed.

### 3.3.3、Examples of Measurable Tuning Goals

Examples of Measurable Tuning Goals可分以下幾項：

- Improved response time
- Improved database availability
- Improved database hit percentages
- Improved memory utilization
- Fewer waits

When tuning a database environment, the DBA should establish measurable tuning goals. Without them, it will be difficult to determine when enough tuning has been performed.

Response time is how long it takes a user to receive data from a request ( for example, the result set of a query ), or update a table, or generate a report.

Database availability is also a food tuning goal. Availability can be impacted due to backup and recovery, or from shutting down and starting the instance to tune parameters.

Database hit percentages provide a good baseline for determining if performance is increasing or decreasing over time.

Memory utilization is also a valid measure, because excessive paging and swapping can impact database and operating system performance. Memory utilization may also impact data hit percentages.

Checking for waits and bottlenecks is another good method for determining whether performance can be improved.

### 3.3.4 、Common Tuning Problems

- Bad session management ( usually related to middleware )
  An example of this is a Web page which continually logs on and off a database. This costs the end user time while the logon procedures are followed.
- Bad cursor management ( usually resulting from programmer error )
  For example, an application that does not make use of bind variables in the where clause. If CURSOR_SHARING is set to SIMILAR or FORCE, then
  - select * from hr.employees where employee_id = 7900; and
  - select * from hr.employees where employee_id = 7369;

  will both parse to a single cursor,
  select * from hr.employees where employee_id = :SYS_B_0;
  even though the SQL text is different.
- Bad relational designs ( usually resulting from over normalization )
  For example, collecting the wrong columns into tables would require many joins in order to produce output that could have been obtained from a single table.

### 3.3.5 、Results of Common Tuning Problems

(1) Bad session management
- Limits scalability to a point you cannot exceed
- Makes the system one or two orders of magnitude slower than it should be

(2) Bad cursor management makes scalability more limited ( not as bad as session management )

(3) Bad relational design
- Unnecessary table joins performed
- Usually a result of trying to build an object interface of relational storage

### 3.3.6 、Proactive Tuning Considerations During Development

During the development of a new system, the following order of tuning implementation is recommended :

- Tune the design
- Tune the application
- Tune memory
- Tune I/O
- Tune contention
- Tune the operating system

Repeat the process if your goals have not yet been achieved.

The rationale for this structure is that improvements early in the sequence can save from having to deal with problems later.

For example, if the applications use many full table scans, this may show up as excessive I/O. However, there is no point in resizing the buffer cache or redistributing disk files, if you can rewrite the queries so that they access only four blocks instead of four thousand.

The first two steps are typically the responsibility of the system architects and application developers; however, the DBA may also be involved in application tuning.

### 3.3.7 、Tuning steps During Production

The Tuning methodology for production systems works by resolving problems when occur :

- Locate a bottleneck by using tools, such as STATSPACK, UTLBSTAT and UTLESTAT, or Oracle Enterprise Manager.
- The bottleneck usually manifests itself as a wait event, Determine the reason for the wait event.
- Resolve the cause of the wait. This could mean changing the size of a member of the System Global Area.

- Check that the change has produced a positive effect on the system by running the application again, and then using the tools used in step (1).
- Repeat the process if your goals have not yet been achieved.

The rationale for this structure is that redoing the same tuning methodology that was used for a development system wastes time. If the system requires a major overhaul, then using the development system methodology is beneficial.

### 3.3.8 、Performance versus Safety Trade-Offs

Factors that affect performance：

- Multiple control files
- Multiple redo log members in a group
- Frequent check pointing
- Backing up data files
- Performing archiving
- Block check numbers
- Number of concurrent users and transactions

Performance Trade-Offs：

There is always a trade-off with performance, In order to increase performance, something else is necessarily affected adversely. Often, recovery time is what suffers. The safer the database administrator makes the database, the slower it runs.

The decision has to be regarding just how safe to make the database, and what level of performance is required; how many concurrent users must have access to the database, how many transactions per second must take place, and so on.

## 4、心得

藉由此次國外訓練機會可以了解Internetwork的概念以及網路的設計。

參加「Enterprise DBA：Architecture and Administration」課程所獲知識如下：

- Learn to use an administration tool to startup and shutdown a database, create a database, manage file and database storage, and manage users and their privileges
- Learn to organize a database and to move data into and between databases, under different environments

參加「Enterprise DBA：Performance and Tuning」課程所獲知識為：

- Create a good initial design
- Define a tuning methodology
- Perform production tuning
- Establish quantifiable goals
- Lists tuning problems
- Decide between performance and safety

## 5、建議

針對此次國外為期十二天實習期間，所學習之內容相當有限，僅能獲得概略性知識，希望承商多提供一些補充資料，可提供職等在日後研究Wireless IN+IP技術發展與規劃之參考，以達成技術自主之目標。

## 6、參考資料

【1】Internetworking Design Guide

【2】Orcle9i DBA Fundamentals Student Guide, Volume 1

【3】Orcle9i DBA Fundamentals Student Guide, Volume 2

【4】Orcle9i Performance Tuning Student Guide, Volume 1

【5】Orcle9i Performance Tuning Student Guide, Volume 2