외국기사소개

분산 시스템 구축상의 과제†

Hashimoto Shozo††

분산 시스템 기반 기술의 향상과 저가격화로 범용 호스트 규모에서부터 UNIX로 대표되는 분산 환경으로 이행, 또는 신시스템 구축을 분산 환경으로 하는 기업이 늘어나고 있다. 또한 분산 DB서버 환경을 활용한 시스템 구축이 급증하고 있다. 라이트 사이징, 다운 사이징, 멀티미디어, 윈도우, LAN 등의 분산 기술은 시스템 구축에 있어 좋은 점을 모두 가지고 있다.

과연, 현실에 있어서도 그럴까? 좋은 면만이 강조되고 있으며, 여러 문제(기술력 및 경험 부족에 기인하고 있는 것도 많지만)가 발생하고 있는 것도 사실이다.

이번에는 분산화의 문제점에 초점을 맞춰서, 분산 시스템에 대한 현장에서 얻은 지식을 토대로 주요한 과제를 열거해서 정리하기로 한다.

1. 분산 시스템 구축

먼저 나와 분산 시스템과의 관계 및 현재의 담당 업무내용에 대해서 설명해 두고자 한다. 그 이유는 나 자신이 분산 시스템의 전문가가 아니라, 이용자의 입장에서 시스템 구축 업무를 통해서, 체험과 매일 생각하고 있던 것이 배경이 되어 문제를 지적하게 되기 때문이다. 개개인의 놓여져 있는 입장과 상황에 따라 착안점은 다르고, 이번에 다룰 과제는 어디까지나 개인적인 사견임을 이해해 주기 바란다.

· 분산 시스템의 구축

1970년 후반에 가격이 저렴한 오피스 컴퓨터가 비지네스 쇼에서 각광을 받고 있을 당시, 80란 카드 베이스 분산 시스템을 본격적인 데이타 베이스화에 의한 온라인 시스템으로 재구축하는 프로젝트가 최초의 만남이었다.

수주한 곳이 외국임에도 불구하고, 일본에서 개발한 후 그것을 현지에 인스톨하는 방식을 채용했지만, 개발 환경과 운용 환경과의 미묘한 차이에 의한 문제에 직면하게 되고, 범용 시스템에서는 생각할 수 없는 대응을 강요받은 경험을 했다.

· End User Commputing 도입

다음에 착수하게 된 것은 항공기 부품 보급 계획 지원을 위한 End User Commputing의 도입이었다. 이 시스템의 특징은,

① 이용 부문이 주체가 된 일상 운용 관리.
② 이용자 자신이 간이 언어를 활용하여 프로그램을 개발.
③ 호스트 데이타의 오프 라인에 의한 월차 제공(데이타량이 많아서 온라인 전송은 불가).

을 들 수 있다. 월차 데이타 갱신 처리에서부터 화일 메인트넌스까지도 이용 부문에 위임하기 위해, 데이타 베이스 구조의 단순화와 데이타 베이스 정비 지원 기능에 충실화를 꾀했다. 이 프로젝트를 통해 얻은 교훈은 다음과 같다.

① 트라블 발생시의 대응 부하가 크다(이용 부문, 시스템 부문 모두)
② 선구적인 업무 추진에는 어려움이 따른다(EUC의 참고 사례가 적다)
③ 이용 부문 주체의 개발, 운용을 성공시키는데는 이용 부문에 강력한 추천력이 필요하다.

· 대규모 분산 시스템의 도입

그후, 여러 건의 분산 시스템에 관여했지만, 모

†Computer Today(日本사이언스사) 94.3월호 게재 기사임.
††일본항공(주) 시스템 개발부 근무

든 시스템이 호스트 환경에서는 코스트적으로 균형이 맞지않는 개별 업무 지원을 목적으로 한 것이 많았다. 그 중에서, 종래 호스트에 대응하는 규모의 안건을 기술과 코스트면에서 분산화하는 것이 적합하다고 판단해서, 운용면에 있어서 다소의 문제가 있었으나, 분산 시스템으로 대응한 적이 있었다. 그 시스템은 항공기의 운용 계획과 관리의 최적화를 지원하는 소위 말하는 현업 지원 온라인 시스템이었다. 대형 그래픽 디스플레이에 계획과 운용 상황의 변화를 즉시 표시하는 기능을 제공하는 시스템으로서, 시스템의 특징은 다음과 같다.

① 개발 기간이 1년이상과 대규모라는 점.
② 이용 부문은 成田공항 정비 기구.
③ 시스템 운용 정지시 업무에 대한 영향이 크다.
④ 응답 성능 요구가 엄격하다.
⑤ 대량의 데이타를 취급한다(데이타 베이스, 단말간 데이타 전송).
⑥ 리얼타임 온라인 처리.
⑦ 그래피컬 기능의 한계 활용.
⑧ 시스템의 일상 운용 관리는 이용 부문(분산으로 했기 때문에).
⑨ 보수 유지 관리는 시스템 부문.

상기 시스템은 시스템 가동 본체 기기를 분산화(설치 장소)했을 뿐이므로, 이러한 시스템을 일반적인 분산 시스템의 범위에 포함시켜서는 안되고, 코호스트 시스템이라고 하는 것이 타당할 것이다.

· OA붐의 시작

'80년대에 들어서, 전세계에서 OA붐을 일으킨 주역인 퍼스널 컴퓨터(이후 PC라고 한다)가 등장했다. 나는 '83에서부터 약 4년간 OA 추진 담당으로서 다음과 같은 업무를 담당했다.

① 각 지점에서 공통적인 업무 지원 패키지의 개발, 유지와 적용의 추진.
② 각 사업소 특유의 OA화 지원.
③ 호스트 데이타를 PC로 다운로드함으로써 데이타 활용 추진.

이들 업무를 통해서, 가장 큰 문제로 지적된 것은 PC상에서 간이 언어로 개개인이 작성한 프로그램(자동 실행 마크로의 집합)이 계승 이용되지 않았다는 점이다. 그 주요 원인은 다음과 같다고 생각한다.

① 프로그램의 기능 사양서가 작성되지 않고, 본인밖에 내용을 이해할 수 없다.
② 이용 부문 OA담당자간에 PC활용 기술에 큰 격차가 있다.
③ 이용 부문이 조직적인 체제를 확립하지 않고, 개인에게 위임했다.
④ 정보 시스템 부문의 지원하는 한계가 있다.

· PC에 의한 호스트 데이타 활용 추진

당시, 호스트 데이타를 PC에 다운로드해서 이용부문 데이타 활용을 추진하려고 노력했지만, 시스템 담당자로부터 적극적인 지원을 받지 못했다. 그 주요한 이유는 다음과 같다고 생각한다.

① 호스트 시스템의 백로그를 포함해서 투입할 매닝을 검출하는 것이 곤란.
② PC의 활용 지원까지 시스템 부문이 관여하는 것을 꺼려한다.
③ 이용 부문이 요구하는 제공 데이타에 가공과 생성을 하기 위한 호스트측의 대응 부하가 크다.
④ 시스템 부문 담당자의 PC에 대한 지식이 적다.

등이다.

· 업무 횡단적인 분산 공용 데이타베이스의 구축 필요성 대두

이즈음, 데이타베이스 관리 시스템에 혁명이 일어났다. RDBMS(릴레이셔널 데이타베이스 관리 시스템)의 등장이다. 당시, OA추진을 담당하는 한편, 공항 발착 관리에 관한 관련 부문의 공통 정보를 분산 공용하는 데이타베이스 구축 프로젝트에 참가했다. 이 시스템은 여러 개의 기간 업무 시스템과 리얼타임 접속에 의해 필요한 데이타를 얻어서 공용 데이타 베이스에 축적하고, 그것을 토대로 정보 검색 및 OA지원 정보를 제공하는 것으로서, 그 특징은,

① 분산 시스템 환경에 의한 공용 데이타베이스의 출현.
② 개발 및 유지 보수의 효율화와 개발 기간의 단축화.
③ 다른 프로토콜에 의한 호스트 시스템간 연결.
④ 현업 지원을 통해 업무 횡단적인 공용 데이타베이스의 구축과 제공 지원 환경의 정비.
⑤ 현업 스텝 부문에 대한 OA화 지원 정보 제공.
⑥ PC에 의한 단말 기능과 OA화 지원 기능의 공용으로, 그를 위해 다음의 기반 환경에 대한 대응을 했다.

- RDBM의 채용(SQL/DS)
- COBOL 소스 생성 언어의 채용(CSP)
- 호스트간 데이타 링크(LUO) 환경의 개발
- IWS 툴 킷의 채용

위에서 기술한 기능에 대한 기대는 오늘날의 분산 시스템에 있어서 DB서버 및 W/S에서 필요로 하는 것과 같다. '86년 현재의 직장으로 이동해서, 다음의 업무를 주로 담당하고 있다.

① 전사 IRM/DB(정보 자원 관리를 위한 DB) 구축과 그 지원 기반 환경 정비.
② 데이타 분석과 데이타베이스 설계에 관한 기술 지원.
③ DOA(데이타 중심 어플로치)에 의한 시스템의 구축 추진과 관련 기법의 확립, 그리고 기술 및 지도.

오늘날, 분산 DB서버에 의한 시스템 구축이 늘어나게 되어, 처리 성능과 데이타 베이스에 관련된 기술 지원 및 상담이 늘어나고 있다. 과거의 경험 등을 토대로 하여 분산 시스템 구축에 있어서의 과제를 정리하면 다음과 같다.

2. 분산 시스템 구축에 있어서의 주요 과제

· 처리 성능의 확보

하드웨어 성능과 DBMS기능 및 성능이 향상되고 있는데도 불구하고, 당초 생각했던대로 처리 성능이 올라가지 않는다는 문제가 발생하여, 개발 지연의 주된 이유가 되고 있다. 처리 성능은 시스템 전체에 관련된 것과 여러 요인이 복잡하게 관련된 것이 많아서, 그 원인의 조사와 개선 대응을 곤란하게 하고 있다. 현재의 상태에서, 분산 시스템의 처리 성능의 확실한 사전 예측을 어렵게 하고 있는 이유는 다음과 같다.

① 벤더 제공의 벤치마크는 어디까지나 표준일 뿐, 그대로 인용할 수 없다.
② 제품의 조합이 다종다양하므로 인해, 모든 조합을 카버하는 예측 방식을 확립하는 것이 곤란하다.
③ 데이타 처리 특성 및 운용 형태에 따라 처리 성능에 큰 굴곡이 생기기 쉽다.
④ 하드웨어와 소프트웨어, 모두 버젼에 의한 처리 성능이 크게 다르다.
⑤ 분산에 관한 경험과 지식이 적다.
⑥ 한정된 개발 기간으로는 사전에 실제 환경에 의한 실측 검증을 실시하는 것이 곤란하다.

가까운 장래에는 확실히 예측할 수 있게 하는 방식과 산정 기준값이 제공되겠지만, 현재로는 실제 환경에서 계측하는 것이외에는 방법이 없다. 이 과제에 대한 사전으로서는 다음의 어프로치로 대응해야 한다고 생각한다.

(1) 본격적인 개발에 착수하기 전에, 기능 요건을 토대로 하여
- 논리 데이타베이스 구조 분석과 설계.
- 주요 기능의 논리 데이타베이스를 토대로 데이타 처리 분석과 정리.
- 전체 처리 부하의 예측.
- 벤더와 공동으로 가상 환경에서의 벤치마크에 의한 사전 검증.

(2) 어플리케이션 설계 도중 또는 종료시에 상기와 마찬가지로 다시 가상 환경에서 벤치마크에 의한 검증 확인

상기 대응은 개발 부하라는 점에서는 반드시 최

선책이라고는 생각하지 않지만, 프로토타입 기법의 효과적인 활용에 의해 부하가 늘어나는 것을 막을 수 있다고 생각한다.

· MIPS라는 함정

처리 성능을 평가하는 기준으로서, MIPS값을 사용한다. 분명히 CPU성능면(클럭)에 있어서, 그 만큼의 명령을 실행하는 것이 가능하다. 단, 그 수치를 범용시스템(MVS등)과 대등하게 비교, 평가해서는 안된다. 분명히, 범용 시스템에 있어서는 MIPS에 비례한 처리 성능을 향상시키는 기구가 있지만(이로인해 고가이다), 분산 시스템의 경우에는 CPU성능이 그대로 성능에 비례하지 않는 경우가 많다. 그 이유는 시스템 전체 구성에 의해 처리 성능이 정해지고, 특히, DB서버, LAN 컨트롤러의 기능과 성능이 크게 영향을 미치는 경우가 많다. MIPS가 호스트와 동등하다고 해서 너무 많은 기대를 하지 말기를 권하고 싶다.

· 분산 데이타베이스의 보전 대책

분산화에서 가장 고려해야 할 내용이 데이타베이스의 보전이다. 그 이유는 하드웨어 및 소프트웨어(어플리케이션 포함)의 장해 발생으로 인한 데이타 파괴의 위험이 항상 뒤따르기 때문이다. DBMS자체의 보전 기능(배타제어, 기능제어, 장해 발생 검지와 복구 등)이 준비되어 있는 것만으로는 부족하다. 호스트 컴퓨터에서는 데이타베이스의 복구에 관한 장치와 절차가 전임 오퍼레이터를 위해 준비되어 있는데, 이것이 중요한 포인트이다. 장해에 대한 복구는 전문 교육을 받은 전임 오퍼레이터가 실시하고 있다. 장해 상황에 따라서는 시스템 엔지니어까지 참가하는 체제가 확립되어 있다.

그러면, 분산 시스템의 경우는 어떤가? 시스템의 활용 목적과 용도에 따라서는 호스트와 동등한 대응을 필요로 하지 않는 경우도 있지만, 일반적으로 데이타 보전을 위한 대응은 부족하다.
기본적으로는,

① 운용 관리는 통상 이용 부문이 담당하고 있고, 일상 운용에 필요한 교육을 받지만 장해 복구를 위한 지식이 적다.
② 데이타 파괴에 영향을 미치는 큰 장해는 거의 생기지 않으므로 교육을 받아도 실제로는 대응할 수 없다.
③ 현재, 제공되고 있는 체계 그 자체가 전문 지식을 갖고 있다는 것을 전제로 하고 있고, 이용 부문 담당자에게 너무 어렵다.
④ 데이타 파괴가 발생하고 있다는 것을 검지하는 것조차 어렵다(검지를 위한 체계를 준비하게 되면 개발 부하가 팽대해진다).
⑤ 데이타 복구의 자동화을 위한 체계를 준비하는 데는 고도의 기술과 많은 투자가 필요하며, 가볍고, 빠르며, 싸다는 분산화의 효과가 줄어들게 되므로 대응할 수 없다.

라는 등, 열거하면 끝이 없다.

데이타 파괴가 발생했을 때는 개발 담당 또는 협력 벤더에게 지원을 요청하는 것이 대부분의 현상이다. 그러나, 시스템 정지가 업무에 중요한 영향을 주는 경우에는 「발생하고 나면 너무 늦지」않도록, 그 나름대로의 대책과 준비를 해야할 필요가 있다. 그 대응책의 한 예를 소개한다.

분산 데이타베이스를 이중화하는 방법이다. 분산 기기 그 자체는 가격이 저렴하다는 점에서 기기에 대한 투자는 소프트웨어 환경에 비해 적다. 데이타베이스를 다른 분산 DB환경에 밀러 데이타베이스화하면, 간단한 분할 조작으로 복구시킬 수 있게 된다. 우리 회사도 이 방법을 일부 채용하고 있다.

· 시스템 장해의 구분

시스템에 장해가 발생했을 경우, 먼저 장해 상황을 파악하고, 원인 조사 대상을 판단한 후, 관련 협력 회사(벤더, 개발 위탁처)에게 조사를 의뢰하는 것이 일반적인 순서이다. 그러나, 분산 시스템이라고 하더라도 그 구성 기기와 기능은 복잡해서 초보자는 대응할 수 없으며, 이용 부문은 발생과 동시에 개발 담당 부문에게 연락하게 된다. 그러

나, 장해라고 하더라도 조작 미스와 기능의 사용 방법을 몰라서 생기는 경우가 많은데, 그때마다 호출한다는 것은 곤란하다. 장해인가 아닌가의 구분은 확실히 대응할 필요가 있지만, 현실적으로 그것이 가능한 것인가? 시스템 전체에 관한 기본적인 지식을 가진 전임 담당자를 이용 부문에 배치하는 것밖에 해결책이 없는 것이 현실이다. 모든 분산 시스템의 운용 관리를 시스템 부문에 집중시켜, 장해에의 신속하고 확실한 대응을 실시하면 해결된다고 생각할 수 있으나, 현실적으로는 불가능하다. 현실적인 대응으로서는,

① 조작 미스, 기능 이해의 미숙으로 인해 생기는 트라블 정도의 대응 가능한 담당자를 이용 부문에 파견 또는 육성.
② 분산 장해 접수 창구를 회사내에 설정하든지 회사밖으로 위탁화.

하는 것이 적당하다고 생각한다. 우리 회사의 경우, 시스템의 개발과 유지보수를 본업으로 하는 관련 회사에 위탁하는 경우가 많다.

· 멀티벤더화에 대해

오픈화에 의한 가격 경쟁 원리의 도입, 뛰어난 활용이라는 장점을 얻기위해, 멀티 벤더화를 지향하는 기업이 늘어나고 있다. 이것은 분명히 유효한 수단이지만, 모든 면에서 뛰어나다고는 할 수 없다. 멀티 벤더화를 추진하는 경우에는 다음의 점에 대해서 고려할 필요가 있다.

① 시스템 전체에 관한 종합적인 기술 지원을 기대할 수 없다.
② 장해가 발생했을 때의 원인 조사와 개선 대응에 시간이 걸린다.
③ 개개의 제품을 조합했을 경우의 최적화에 관한 지원을 받기 어렵다.
④ 장해 발생시의 원인의 일차 구분이 곤란하다.
⑤ 버전 관리의 부담이 증가한다.

특히, 처리 성능에 문제가 발생했을 때 위와 같은 상황이 현저하게 나타난다. 처리 성능은 시스템 가동 환경 성능(하드웨어, 소프트웨어, 네트워크)과 데이타베이스 및 어플리케이션 처리 구조가 복잡하게 얽히는 경우가 많다. 벤더는 어플리케이션의 문제, 이플리케이션 개발측은 벤더 제공품의 성능 문제로서, 취해야할 자세를 전면으로 나타내기 때문에(문제 해결보다도 책임 회피 우선), 원인 조사와 개선 대응에 많은 시간을 요하는 상황에 자주 직면했다.

종합적인 기술 지원을 얻기 위해서는 분산 기술을 폭넓고 세부에 걸쳐 알고 있는 시스템엔지니어를 많이 고용하고 있는 소프트 하우스 또는 벤더에 의뢰하던지 외부에 의뢰하지 않고 자기 회사내에 그 요원을 확보해야 하지만, 현재는 외부에 기대하는 것이 좋은 방책이라고 말할 수 있다. 그 이유는,

① 이용자는 분산의 유효적인 활용에 중점을 두고, 각종 제품의 상세한 기술을 숙지하는데 시간을 낭비해서는 안된다.
② 기술의 진보가 빠르고, 그에 관련된 한계가 있다.

또, 멀티 벤더화에 의한 위험을 줄임과 동시에, 회사내에 분산 기술을 조기에 축적하는 것을 꾀하는데도 다음의 대응이 유효하다.

① 무리하게 다종다양한 제품을 도입 활용하지 않는다.
② 분산화의 영역을 설정(그룹핑)해서, 그 단위에서의 표준적인 제품 조합을 확정 한다.

· 버전 관리

호스트와 달리, 제품 개발과 개선이 용이하다고 해서 버전업이 단기적으로 행해지고 있다. 그러나, 벤더간에 있어서 그 대응이 개개로 실시되기 때문에 버전간의 충돌이 생긴다. 기능이 단기간에 풍부해지고, 충실해지는 것은 이용자가 환영할 만한 것이지만, 버전 관리와 갱신 작업 등, 분산시스템의 유지보수에 관한 부하가 증대하고 있다. 버전에 관한 주요 작업 항목은,

① 신버전 기능의 조사와 파악.

② 제품간 버젼 충돌 관리.
③ 버전업에 따른 영향 대상 범위 조사.
④ 버전업에 따른, 어플리케이션 프로그램 및 데이타베이스의 재생성.

등이다. 이들 작업을 지원하는 환경의 정비와 관리, 유지 보수 체제가 확립되어 있지 않으면 부하의 증대화를 불러일으켜서, 분산화의 장점이 감소한다.

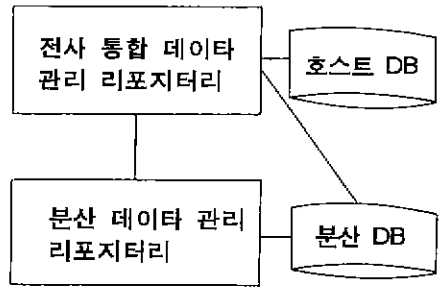
앞으로, 분산화의 확대와 오픈화에 의한 멀티벤더 지향으로 인해, 이 문제는 클로즈업될 것이다. 이제부터 실제 시스템에 대한 자원 관리의 일환으로서 버전 관리 기법을 위한 관리 환경과 체제에 대해 설명하기로 한다.

· 분산화와 데이타의 관리와 통제

데이타베이스를 분산 시스템상에 전개, 그것을 유효하게 활용함으로써, 이용 부문의 데이타 활용의 활성화, 데이타의 이차 가공 작업 부하의 경감화와 용이화를 꾀할 수 있다는 것은 의심할 여지가 없다. 그러나, 한편에서는 데이타의 표준화, 데이타 중복성의 배제라는 면에 있어서의 관리와 통제가 곤란하게 된다. 데이타 관리 입장에서 이 문제를 무시할 수는 없다. 데이타에 관한 표준화와 충돌 방지를 어느 범위 또는 어느 정도로 할 것인가는 각 기업에 따라 상황은 다르겠지만, 아뭏든 이용 부문이 주체성을 가지고, 분산 환경을 활용한 시스템화가 실시된다면 앞으로 대책을 세울 필요가 있다. 시스템 부문의 데이타 관리자가 모든 분산 시스템의 구축에 관여하는 것이 가능하다면 이 문제는 발생하지 않을 것이다. 이를 위해서는 분산 데이타 베이스를 포함한 기업 전체의 데이타 관리를 가능하게 하는 지원 환경이 정비되어야 한다는 것이 전체가 된다. 이용 부문 주체에 의한 분산 시스템이 데이타 관리면에 있어서 마이너스가 되는 것은 다음의 이유때문이다.

① 데이타의 표준화와 충돌 방지라는 관점을 유지해 나갈 수가 없다.
② 데이타의 정보를 구별할 수가 없다.
③ 시스템 대상 업무 범위에서의 표준화와 충돌 방지가 한계이다.
④ 데이타베이스 설계는 개발 위탁처에 맡기기 위해서, 데이타 본래의 의미까지 언급한 분석이 되기 어렵다.

데이타 관리자가 모든 분산 데이타 베이스에 참가 또는 검증, 평가하는데는 리포지터리를 중핵으로 하는 데이타 관리를 위한 데이타 베이스 구축과 그 지원 환경을 정비하는 것이 필요하다. 우리 회사에서는 현재 전사적인 규모로 경영 활동의 중추 기능이 활용하는 데이타를 개념 데이타 모델로서 관리하는 프로젝트를 수행하고 있고, 앞으로 분산 데이타 베이스의 중추 데이타도 대상을 확대하려 하고 있다. 이러한 환경이 갖추어지면, 분산을 포함한 통합적인 데이타 관리를 추진할 수 있게 된다.



개개의 분산 DB를 대상으로 한 분산 리포지터리 환경을 준비할 것인가, 통합 리포지터리로 모든 것을 관리할 것인가는 데이타 관리 지원 기반 환경에 따라 정하지만, 현 단계에서는 통합 리포지터리에 의한 집중 관리가 관리의 용이성, 환경 정비 코스트의 억제하는 관점에서 좋은 방법이라고 생각된다.

현상의 분산 시스템 구축에 있어서는 여러 과제를 갖고 있지만, 그 기대는 크고 착실히 발전, 확대되고 있다고 확신한다. 마지막으로 시스템의 분산화를 추진할 때의 기본적인 어프로치에 관해서

사견을 기술하면 다음과 같다.

① 시스템 체계(데이타베이스, 어플리케이션, 시스템 기반, 네트워크)를 확립한 후, 실현 기술을 토대로 한 계획 수립과 그 계획을 토대로 한 착실한 전개.

② 분산 시스템 구축 기술 지원 체제의 강화와 충실.

③ 비지네스 그 자체의 수정(집중 관리방식으로부터 분리 분산 독립형으로 탈피).

◈ 제 1회 부산국제컴퓨터/소프트웨어 전시회 ◈

다음과 같이 소프트웨어 전시회를 개최하오니 많은 참석 바랍니다.

1. 일시 : 1994. 9.9(금)~9.13(화)
2. 장소 : 부산 무역전시장
3. 주관 : 체신부
4. 주최 : 한국정보처리전문가협회, 전자신문사
5. 후원 : 한국통신