

WEB GIS시스템을 통한 고해상도 영상지도의 속도향상을 위한 연구

정명훈¹ · 서용철^{2*}

A Study on Improving the Performance of High Resolution Image Web GIS System

Myeong-Hun JEONG¹ · Yong-Cheol SUH^{2*}

요 약

최근 실생활이 이루어지고 있는 공간을 모델링하여 컴퓨터를 통해 공간데이터를 구현하고자 하는 요구가 증가하면서 GIS의 중요성이 증대되고 있다. 여기에 인터넷 서비스가 제공되면서 GIS 분야에 이를 기반으로 한 Web GIS라는 새로운 개념이 나타나고 혁신적인 인터넷GIS 영역이 발달하게 되었다. 이러한 Web GIS 시스템을 통하여 고해상도의 영상지도서비스를 구현하기 위해서는 시스템의 성능 향상이 필수적이다.

본 논문은 Web GIS 시스템을 통하여 고해상도의 영상지도를 서비스하는 경우 속도향상을 위한 방안을 제시하는 것이다. Database 레벨에서 IBM AIX 운영시스템과 Oracle, ArcSDE에 대한 튜닝 작업을 수행하였으며, 응용 시스템 레벨에서는 Tomcat, ArcIMS에 대한 최적화된 설정과 성능향상을 위한 코딩을 수행하였다. 수행 결과 본 연구를 통하여 제안된 방법으로 고해상도 영상의 Web GIS 시스템의 높은 성능을 보장 하는 시스템을 구축할 수 있었다.

주요어 : Web GIS, AJAX, Tuning, 성능테스트

ABSTRACT

Nowdays the need of modeling a real world using computers leads to increase the importance of GIS. Moreover, the wide spread of the Internet service has brought about Web GIS Development. Especially, the tuning of Web GIS system has a crucial role in improving the performance of high resolution image Web GIS system.

Thus, this study provides a variety of ways to improve the performance of high resolution image Web GIS system. We have tuned IBM AIX operation system, Oracle, ArcSDE in the database level and Tomcat, ArcIMS in the application level. What is more, we propose ways of coding which can have a positive impact on performance. As a result, the proposed methods guarantee high resolution image Web GIS system to improve performance.

KEYWORDS : *Web GIS, AJAX, Tuning, Performance Test*

2008년 11월 12일 접수 Received on November. 12, 2008 / 2008년 12월 15일 수정 Revised on December 15, 2008 / 2008년 12월 16일 심사완료 Accepted on December 16, 2008

1 SK C&C Solution 사업팀 Solution Business Team, SK C&C

2 부경대학교 위성정보과학과 Department of Geoinformatic Engineering, Pukyong National University

* 연락처 E-mail: suh@pknu.ac.kr

서론

최근 실생활이 이루어지고 있는 공간을 모델링하여 컴퓨터를 통해 공간데이터를 구현하고자 하는 요구가 증가하면서 GIS(Geographic Information System)의 중요성이 증대되고 있다. 여기에 인터넷 서비스가 제공되면서 GIS 분야에 이를 기반으로 한 Web GIS라는 새로운 개념이 나타나고 혁신적인 인터넷 GIS 영역이 발달하게 되었다(오정현, 2007).

Web GIS는 Web을 기반으로 하였기 때문에 인터넷 서비스와 밀접한 관계가 있어 Web의 발달에 따른 그 서비스 제공 형태도 변화되어 왔다. 우선적으로 초기의 Web 1.0은 단순히 위치를 찾는 위치정보 조회지도, 사용자가 원하는 곳을 보여주는 사용자 요청지도, 공간 데이터 제공 등 GIS의 기본적 요소들을 다양한 형태로 인터넷에서 제공하는 것이었다. 현재는 Web 2.0이 대세로 인터넷 서비스 형태가 변화하면서 Web 1.0 시대의 일대다 관계에서 벗어나 다대다 관계로 확장되었다. 이로 인해 사용자들 스스로가 정보 제공에 대한 주체가 되면서 Web GIS 서비스의 형태가 쌍방향의 개방형으로 더욱 더 빠르게 변화하고 있으며 사용자 참여도의 폭이 넓어지고 이용도가 향상되고 있다(장진주 등, 2007). 이러한 Web GIS 시스템을 통하여 고해상도의 영상지도서비스를 구현하기 위해서는 시스템의 성능 향상이 필수적이다. 따라서 기존의 연구들이 응용 분야별로 Web GIS 시스템을 구축하는 것에 초점을 두었다면 본 연구에서는 Web GIS 시스템의 성능향상을 위한 방법들을 제안하였다. 따라서 본 연구에서는 용인시 지리정보 관련 데이터를 가지고 Database, 응용시스템에 대하여 동시 접속 사용자 50명을 가정하여 성능 테스트 소프트웨어인 LoadRunner를 이용하여 각 기능별 Response Time을 체크하면서 Database와 응용 시스템의 성능을 모니터링하여 시스템 속도 향상을 위한 방안을 제시하였다.

시스템 아키텍처

본 연구에서 성능테스트를 위한 시스템 아키텍처는 그림 1과 같다. Database 레벨에서는 IBM p570장비에 Oracle9i, ArcSDE9.1을 사용하였으며, Application Server 레벨에서는 Windows 2003 Server에 Tomcat5.5, ArcIMS9.1로 구성되었고, Client 레벨에서는 MS 종속적인 기술인 Active X를 사용하지 않고 AJAX(Asynchronous JavaScript and XML)를 이용하여 다양하고 풍부한 유저 인터페이스를 제공하였으며, LoadRunner를 이용하여 성능테스트를 수행하였다.

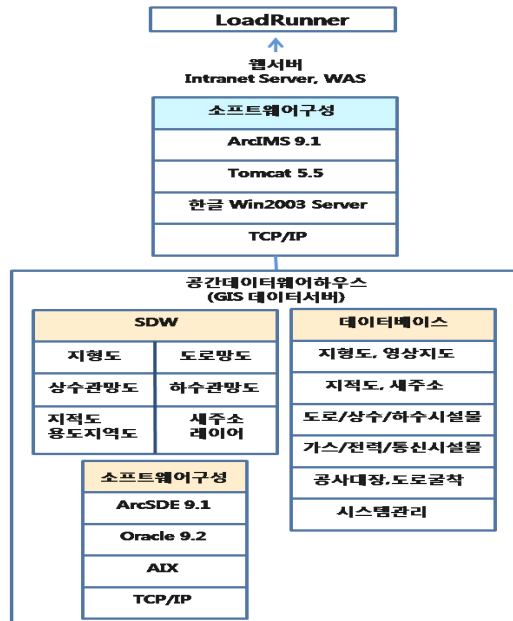
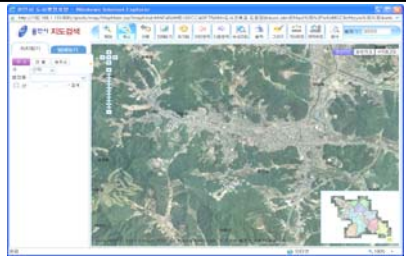



그림 1. H/W 및 S/W 총괄 구성도

1. 시스템 주요화면

본 연구를 통하여 개발한 지도화면은 WEB2.0의 대표적인 기술로서 AJAX 기능을 통한 비동기적 통신으로 속도 향상을 가져왔으며, 시스템의 주요 화면은 다음과 같다.

표 1. 시스템 주요화면

주요 화면	화면 설명
	지도관리창 화면(툴바, 지도창, 위치찾기, 범례보기, 네비게이션창)
	지번 위치 조회 후 지번의 국공유지 정보 조회 화면과 지번을 중심으로 반경 50m영역 거리제한 검색을 수행한 화면

성능 향상을 위한 방안

본 연구에서는 시스템의 성능테스트를 수행하기 전 Database, 응용 시스템에 대한 성능향상을 위한 사전 작업을 수행하였다.

1. Database 성능향상을 위한 제안

Database 레벨에서 성능 향상을 위하여 IBM 시스템의 AIX운영시스템과 Oracle9i, ArcSDE9.1에 대한 튜닝을 수행하였다.

1.1 IBM AIX 구성

AIX 시스템은 TCP/IP 네트워크에서 네트워크의 효율적인 사용을 위하여 몇 개의 메시지를 TCP/IP 패킷에 넣기 위하여 Nagle 알고리즘을 사용하고 있다. 이 알고리즘은 특히 고해상도의 영상데이터 전송시 과도한 부하를 증가시킨다. 따라서, AIX 시스템의 Nagle 알고리즘 사용여부를 담당하는 tcp_nodelay 파라미터 값을 비활성화 시킨다. 또한, AIX시스템에서 Oracle을 사용하는 경우 메모리 튜닝

을 다음과 같이 수행한다.

```
# vmo -p -o maxclient%=20 -o
maxperm%=20 -o minperm%=10 -o
lru_file_repage=0
```

1.2 Oracle 튜닝

본 연구에서 시스템의 데이터베이스는 Oracle을 사용하였다. 대용량의 영상데이터를 전송하기 위해서는 Oracle 튜닝이 필수적이다. 우선, Oracle 사용시 SGA메모리 할당 권고값인 서버의 물리적 메모리 66%~80%를 Oracle의 SGA에 할당하였다. 또한, Oracle의 다양한 파라미터들을 세팅하여 최적의 성능을 구현할 수 있도록 설정하였다.

1.3 ArcSDE 구성

본 연구의 시스템에서는 각종 공간데이터를 저장하기 위해서 ArcSDE를 사용하였다. 고해상도의 항공사진을 ArcSDE를 통하여 서비스하기 위해서 데이터 로딩시 관련 설정을 수정해야하며 영상데이터 로딩 후 DBMS Statistics를 수행하여 ArcSDE를 최적화 시켜야 한다.

표 2. Oracle 파라미터 튜닝

파라미터 명칭	설 명
log_buffer	SGA안에 존재하는 REDO LOG BUFFER에 할당된 BYTE수. Transaction이 길거나 양이 많을 경우 File I/O를 감소시키기 위해서 크게 설정하였다. CPU 개수 * 128KB
shared_pool_size	Shared Pool은 Shared Cursor와 Shared Procedure를 포함한다. 큰 값은 Multi User System에서 속도 향상을 가져오지만 메모리 낭비가 따른다. 16의 배수로 설정하였다.
db_cache_size	Buffer Pool의 메모리 사이즈를 지정한다. 아래의 공식을 이용하여 산정하였다. (memory available to SGA - (shared_pool_size + log_buffer)) * 0.9
workarea_size_policy	workarea_size_policy는 SQL work Area(sort, group-by, hash-join, bitmap merger, bitmap create)Size 정책을 Oracle system에 맞길 것인지 아니면 *Area_Size에 지정한 파라미터에 의해 운영할 것인지를 지정한다. Auto로 설정하여 시스템에서 사용되고 있는 PGA의 메모리에 따라 시스템에서 자동으로 조절하도록 하였다.
pga_aggregate_target	Instance내에서 모든 Server Process들이 사용하는 PGA의 총합을 지정한다. 다음의 공식을 적용하였다. total physical RAM * 0.16
db_block_size	Data Block의 크기. Block Size의 값은 DB Creation시에 결정되어지며, 임의로 바꿀 수가 없다. 16K로 세팅하였다.
open_cursor	Single User Process에 의해서 동시에 Open 되어지는 최대 Cursor 개수. 최소 2000으로 세팅하였다.
session_cached_cursor	Cache에 관리 될 session cursor cache의 수를 지정한다. 같은 SQL문장이 반복적으로 Parse Call이 반복될 경우 Session Cursor는 Session Cursor Cache로 이동되며, 이 cache수를 지정하는데 사용된다. Cache에 있는 cursor를 다시 call 했을 경우 cursor의 reopen이 발생되지 않으므로 성능의 개선효과가 있다. 본 연구에서는 50으로 세팅하였다. 그러나 1회성 SQL문장이 많은 시스템에서는 사용하지 않도록 한다.
cursor_sharing	1회성 SQL의 Overhead를 줄이기 위한 파라미터이다. EXACT로 세팅하였다.
cursor_space_for_time	Active Cursor가 SQL Area 메모리에 존재하여 속도가 빠르고 사용되어지는 동안에는 메모리에 계속적으로 상주되어 진다. True로 설정하였다. 비 공유 SQL문장 및 SQL의 종류가 집중적으로 많이 실행되는 곳은 사용하지 않도록 한다.
pre_page_sga	Instance가 처음 기동될 때 shared 메모리에 설정되는 모든 SGA를 접근함으로써 실제 메모리로 전환된다. 이렇게 함으로써 Page Fault를 다소간 방지할 수 있지만 기동시 약간의 시간이 더 걸릴 수 있다. True로 설정하였다.
log_checkpoint_interval	0으로 세팅하여 Log Switch 발생시 checkpoint 발생하게 하였다.
log_checkpoint_timeout	checkpoint가 발생 후 다음 번 checkpoint 발생시점까지의 대기시간으로 0으로 세팅하여 Time_Based checkpoint를 실행하지 않게 하였다.

본 연구에서 사용한 용인시 고해상도 영상 파일은 TIF파일로 하나의 파일 당 약 250메가로 전체 30Giga로 이루어져 있다. TIF파일을 ArcSDE에 로딩하기 위한 압축방식으로는 JPEG(50)을 사용하였다. 또한 영상처리 성능향상을 위하여 Bilinear Pyramid를 적용하였다. Pyramid적용은 전체 공간영역은 같으나 셀 사이즈를 다르게 하여 영상지도 조회 시 속도를 향상시킨다(정명훈, 2000).

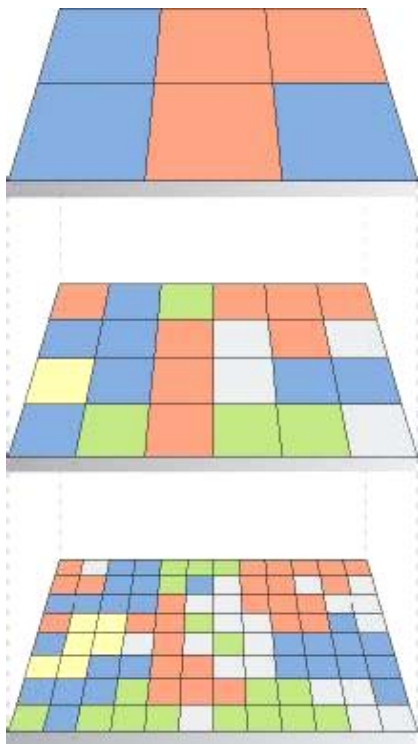


그림 2. Pyramid 생성

마지막으로, ArcSDE에서 영상데이터를 로딩시 관련 테이블이 7개가 생성된다. 각각의 테이블 정보를 저장시 Raster block을 담당하는 테이블 (BKL_STROAGE, BLK_INDEX_COMPOSITE)은 Oracle에서 별도의 테이블스페이스를 만들어서 분리하여 저장해야 한다.(ESRI Technical Team, 2005) 이와 같은 세팅은 SDE.DBTUNE안에 파라미터를 세팅하여 수행하였다.

```

#RASTER
B_STORAGE *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
B_INDEX_ROWID *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
FIAS_STORAGE *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
FIAS_INDEX_ID *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
BND_STORAGE *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
BND_INDEX_COMPOSITE *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
BND_INDEX_ID *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
AUX_STORAGE *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
AUX_INDEX_COMPOSITE *PCTFREE 0 INITRANS 4 TABLESPACE GDB*
BLK_STORAGE *PCTFREE 0 INITRANS 4 TABLESPACE GDB_BLOCKS*
BLK_INDEX_COMPOSITE *PCTFREE 0 INITRANS 4 TABLESPACE GDB_BLOCKS*
END
    
```

그림 3. 영상데이터 설정방법

2. 응용시스템 성능향상을 위한 제안

본 연구에서는 ArcIMS9.1과 Tomcat5.5를 사용하여 Application Server를 구성하였으며, Java와 AJAX등 Web2.0기술을 이용하여 응용시스템을 개발하였다.

2.1 Tomcat 세팅

Tomcat 설정에서 performance에 관련을 갖는 parameter들은 JVM heap size, connection pool, enableLookups, reloadable 등이 있으며, 각 설정 값의 상세 내용은 아래와 같다(표 3).

2.2 ArcIMS 세팅

ArcIMS는 확장 용이한 인터넷 맵 서버로써 웹에서 많은 사용자들에게 맵, 데이터, 메타데이터를 제공하기 위한 웹 배포용 GIS에 널리 사용되고 있다.

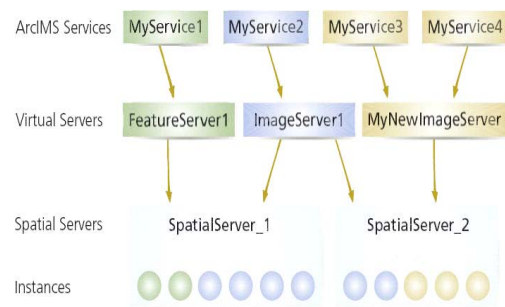


그림 4. ArcIMS 사용자 요청 처리 구조

표 3. Tomcat 성능관련 요소

파라미터	설 명
enableLookups	DNS lookup을 활성화 시킬 것인지 여부를 정한다. DNS lookup이 활성화된 경우(true인 경우), request.getRemoteHost()을 호출하여 클라이언트의 host name을 리턴해 주는 DNS lookup을 실행하게 된다. “alse”로 설정한 경우, DNS lookup을 건너뛰고 대신 String으로 된 IP 주소를 리턴하게 된다. DNS lookup을 활성화하는 것은 성능에 악영향을 미칠 수 있으므로 false로 설정하였다.
reloadable	“true”로 설정된 경우, tomcat은 WEB-INF/classes와 WEB-INF/lib 디렉토리 내 class의 변경 사항을 체크하게 된다. 변경 사항이 있게 되면 이러한 class를 가진 application이 자동으로 reload된다. 이러한 설정은 심각한 성능 저하를 가져올 수 있으므로 “false”로 설정하였다.
JVM heap size	catalina.bat파일 안에 JAVA_OPTS의 세팅을 수정한다. Default 세팅으로 다중 사용자 접속 사용시 Tomcat이 Down되는 현상이 발생된다. 따라서, Java runtime시 메모리 세팅 값을 변경하였다. set JAVA_OPTS=%JAVA_OPTS% -Xms1024m -Xmx1024m
connection pool	컨텍스트 xml파일안에 Resource세팅시 maxActive, maxIdle세팅하여 커넥션 풀을 미리 생성하여 데이터베이스 커넥션시 속도 향상을 가져왔다. <Resource name=“jdbc/UPEXDS” auth=“Container” type=“javax.sql.DataSource” factory=“org.objectweb.jndi.DataSourceFactory” driverClassName=“oracle.jdbc.driver.OracleDriver” username=“***” password=“***” url=“jdbc:oracle:thin:@888.88.88.88:1521:test” maxActive=“50” maxIdle=“50”/>

ArcIMS가 서비스로부터 직접 요청된 작업을 받으면, Virtual Server가 그 요청된 작업을 처리한다. Instance 들과 함께 Virtual Server는 Application Server의 대기열에 들어간다. 이 instance들은 Saptial Server 내에 존재한다. 요청된 작업이 도착하면 서비스에서 사용 가능해 질 때까지 대기 열에서 기다리게 된다. 왜냐하면 Virtual Server의 각각의 instance들은 동일한 서비스를 제공해 주며, 요청된 작업은 round-robin 형식으로 처리된다. 다중 instance를 사용하면 작업 처리량과 응답 시간은 개선되지만, 그렇다고 무제한으로 instance를 증가 시킬 수 는 없다. 모든 instance는 추가적인 메모리를 필요로 하며, 운영 시스템에 있어서 thread로 관리된다. 이것은 실제로 작업을 처리중인 instance와 함께

추가적인 부하로 작용한다. 최적의 instance 숫자는 시스템 사양과 서비스 내용에 따라 달라진다. 본 연구에서는 CPU당 2개의 instance로 세팅하였다.

2.3 응용시스템 개발

본 연구의 응용시스템은 JAVA 언어와 ArcIMS의 JAVA Connector를 사용하여 응용 시스템을 개발하였으며, JSP와 AJAX를 사용하여 사용자 화면을 구성하였다. 응용시스템 개발시 디스크 I/O의 발생 빈도를 낮추고, 메모리 과다 사용을 막기 위한 코딩을 수행하였다. 또한 Client 측면 개발시 AJAX 기술을 사용함으로써 사용자에게 불필요하 Plug-in 설치를 제거하였고 전체화면 Refresh없이 비동기적 통신을 통한 지도화면 구성을 수행하였다.

<p>반복문안에 변수선언 제거</p> <pre> FeatureLayer featureLayer = null; for(int i= imsMap.getLayers().getCount()-1; i>= 0; i--){ if(FEATURE.equals(imsMap.getLayers().item(i).getType())){ // 기존의 HighlightLayer 지운다. featureLayer = (FeatureLayer)imsMap.getLayers().item(i); if(HIGHLIGHTLAYER.equals(featureLayer.getName())){ imsMap.getLayers().remove(i); } } } </pre>	<p>StringBuffer를 사용한 메모리 최적화</p> <pre> //strQuery = sField + " = &apos;" + sQuery + "&apos;"; StringBuffer sb = new StringBuffer(100); sb.append(sField); sb.append(" = &apos;"); sb.append(sQuery); sb.append("&apos;"); strQuery = sb.toString(); </pre>
---	---

그림 5. 메모리 사용 감소를 위한 코딩 방법 예시

성능 테스트 비교 및 평가

본 연구에서는 총 4가지의 테스트 시나리오를 가지고 성능테스트 소프트웨어인 LoadRunner를 이용하여 성능테스트를 수행하였다. 각 테스트 시나리오의 정의는 다음과 같다.

또한, Database는 오라클에서 제공하는 DB 모니터링 리포트 툴인 STATSPACK와 응용시스템은 제니퍼를 사용하여 모니터링 하였다. 성능테스트의 목표성능은 각 기능들에 대하여 서버평균응답시간은 5초로 하였으며, TPS (Transaction Per Second)는 기존 사용시스템의 Web Access Log를 분석하여 1.63TPS로 하였다.

표 4. 성능 테스트 시나리오

테스트 시나리오	상세 설명
시나리오 1	구축된 시스템의 기본환경에서 부하레벨 30명을 기준으로(Think Time 5초) 지도 확대 기능에 대한 응답시간
시나리오 2	구축된 시스템의 기본환경에서 ArcIMS 세팅만 수정한 후 부하레벨 30명을 기준으로(Think Time 5초) 지도확대 기능에 대한 응답시간
시나리오 3	구축된 시스템의 기본환경에서 본 연구에서 제안한 방법으로 데이터베이스, 응용시스템을 구축 후 부하레벨 50명을 기준으로(Think Time 10초) 지도 확대, 주소검색, 국공유지정보조회, 거리제한 검색, 출력, 건물명검색, 토지이용주제도조회, 인허가민원파일저장 기능에 대한응답시간
시나리오 4	구축된 시스템의 기본환경에서 본 연구에서 제안한 방법으로 데이터베이스, 응용시스템을 구축 후 부하레벨 50명을 기준으로(Think Time 5초) 지도 확대, 주소검색, 국공유지정보조회, 거리제한 검색, 출력, 건물명검색, 토지이용주제도조회, 인허가민원파일저장 기능에 대한응답시간

1. 시나리오 1 결과

시나리오 1은 시스템 구축 후 기본 환경에서 지도확대 기능에 대하여 부하레벨 30명 수준의 상황(Think Time 5초)에서 부하테스트를 수행한 결과, Transaction 평균 응답시간이 4.203초 수준이었다. 성능테스트 수행 시 Application 서버의 CPU자원 사용률은 약 94%로 높은 사용률을 보였으며, GIS DB서버의 자원 사용률은 약 8% 양호한 것으로 판단되었다. Application 서버의 성능 저하로 GIS DB의 사용률이 낮은 것으로 판단되었다.

표 5. 자원사용률 추이(시나리오 1)

구 분	사 용 률
Application서버 CPU	94%
Application서버 메모리	15%
GIS DB서버 CPU	5%
GIS DB서버 메모리	1%

표 6. 자원사용률 추이(시나리오 2)

구 분	사 용 률
Application서버 CPU	91.75%
Application서버 메모리	19.7%
GIS DB서버 CPU	26%
GIS DB서버 메모리	3%

2. 시나리오 2 결과

시나리오 2는 시스템 구축 후 기본 환경에서 ArcIMS 관련 세팅만 변경한 후 지도확대 기능에 대하여 부하레벨 30명 수준의 상황(Think Time 5초)에서 부하테스트를 수행한 결과, Transaction 평균 응답시간이 2.817초 수준이었다. 성능테스트 수행 시 Application 서버의 CPU자원 사용률은 약 91.75%로 높은 사용률은 보였으며, GIS DB서버의 자원 사용률은 약 26%로 양호한 것으로 판단되었다. 시나리오 2의 결과도 Application 서버의 과도한 자원 사용과 늦은 응답시간의 문제가 발생하였다.

3. 시나리오 3 결과

시나리오 3은 본 연구에서 제안한 방법을 적용한 후 시스템의 성능테스트 대상 업무를 통하여 부하레벨 50명 수준의 상황(Think Time 10초)에서 부하테스트를 수행한 결과, 전체 대상 업무의 주요 Transaction 평균 응답시간이 0.1~4.2초 수준으로 대체로 양호하였으며, 10분간 Business Transaction 처리건수는 2,592건(4.32 TPS)로 기준 부하 대비 약 2.4배 부하 수준이었다. 성능테스트 수행 시 CPU자원 사용률은 Application Server 및 GIS DB서버 공히 양호하였으며, 서비스다운 등의 비정상적인 현상이 발생하지 않았다.

테스트 결과 시스템의 모든 지도관련 기능

시스템	대상 업무	주요 Transaction	성능목표			수행 결과				
			서버응답시간 (평균,초)	처리건수 (10분)	TPS	복합성능테스트(최대) - 부하레벨50명,ThinkTime10초				
						부하레벨	서버응답시간 (평균,초)	처리건수 (10분)	TPS	CPU%
테스트 시스템	지도확대	지도확대	5.0	486	0.81	10	0.6	545	0.91	Application Server AVG: 28.5% MAX: 39.5% GIS DB AVG: 13.1% MAX: 20.1%
	주소검색	주소검색	5.0	243	0.41	8	0.2	432	0.72	
		주소선택	5.0		0.5					
	국공유지정보조회	국공유지선택	5.0	47	0.08	5	0.6	257	0.43	
	거리제한검색	거리제한검색	5.0	47	0.08	7	0.5	361	0.60	
	출력	간략기능확대 (1:5000)	5.0	42	0.07	5	0.4	263	0.44	
		미리보기	5.0		0.5					
	건물명검색	Ajax(제시여기능)	5.0	113	0.19	5	0.1	270	0.45	
		건물명검색	5.0		0.5					
	인허가민원파일추가	저장	5.0	49	0.08	5	4.2	190	0.32	
토지이용주제도조회	확대	5.0	49	0.08	5	0.6	274	0.46		
총계			-	1,076	1.79	50	-	2,592	4.32	

그림 6. 응답시간 결과(시나리오 3)

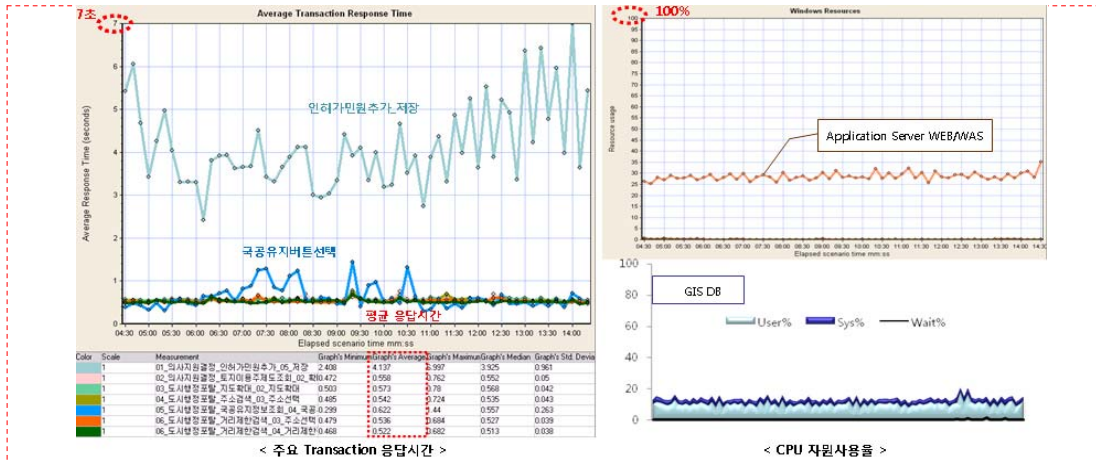


그림 7. 응답시간 및 CPU 자원사용률 추이(Think Time 10초)

서브시스템	대상 업무	주요 Transaction	성능목표			수행 결과				
			서버응답시간 (평균,초)	처리건수 (10분)	TPS	복합성능테스트(최대) - 부하레벨50명,ThinkTime5초				
						부하레벨	서버응답시간 (평균,초)	처리건수 (10분)	TPS	CPU%
테스트 시스템	지도확대	지도확대	5.0	486	0.81	11	0.7	1,042	1.74	Application Server WEB/WAS AVG: 44.2% MAX: 51.6% GIS DB AVG: 21.1% MAX: 23.3%
	주소검색	주소검색	5.0	243	0.41	9	0.5	794	1.32	
		주소선택	5.0	47	0.08	5	1.7	354	0.59	
	국공유지정보조회	국공유지선택	5.0	47	0.08	7	0.7	558	0.93	
	거리제한검색	거리제한검색	5.0	42	0.07	5	0.5	432	0.72	
		간략기능확대 (1:5000)	5.0	113	0.19	6	0.4	512	0.85	
	출력	미리보기	5.0	49	0.08	2	4.1	110	0.18	
	건물명검색	Ajax(제시어가능)	5.0	49	0.08	5	0.7	480	0.80	
	인허가민원파일추가	저장	5.0	49	0.08	5	0.7	480	0.80	
토지이용주제도조회	확대	5.0	49	0.08	5	0.7	480	0.80		
총계			-	1,076	1.79	50	-	4,282	7.14	

그림 8. 응답시간 결과(시나리오 4)

들은 평균 응답시간이 1초 이내로 들어왔다. 또한 Application Server의 CPU사용률은 약 28.5%로 안정적인 상태를 유지하였다. 하지만, 부가적으로 추가한 기능인 인허가민원파일추가 기능은 약 6메가 파일의 로딩 시간으로 4초가량의 응답시간이 걸렸다.

4. 시나리오 4 결과

시나리오 4는 본 연구에서 제안한 방법을 적용한 후 성능테스트 대상 업무를 통하여 부하레벨 50명 수준의 상황(Think Time 5초)에서 복합 부하테스트를 수행한 결과, 전체 대상 업무의 주요

Transaction 평균 응답시간이 0.4~4.1초 수준으로 대체로 양호하였으며, 10분간 Business Transaction 처리건수는 4,282건(7.14 TPS)로 기준 부하 대비 약 4.0배 부하 수준이다. Think Time을 5초로 줄임으로써 처리건수가 늘어난 것을 알 수 있었으며 CPU자원 사용률은 Application Server 및 GIS DB서버 공히 양호하였고, 서비스다운 등의 비정상적인 현상이 발생하지 않았다.

Think Time 5초 성능테스트 수행시도 시스템의 모든 지도관련 기능들은 평균 응답시간이 1초 이내로 들어왔으며 Application Server의 CPU사용률도 44.2%로 안정적이었다.

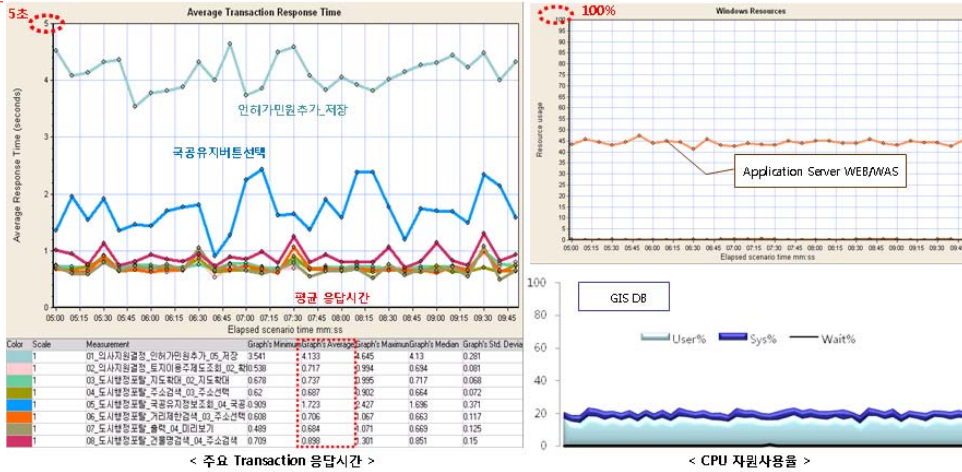


그림 9. 응답시간 및 CPU 자원사용을 추이(Think Time 5초)

결론

본 연구는 WEB GIS시스템에서 고해상도 영상의 지도서비스 시 성능 향상을 위한 방법들을 제안하였다. 시스템 구축 후 기본환경에서 지도확대기능에 대한 응답시간을 본 연구에서 제안한 방법과 비교한 결과 약 6배 이상의 성능 효과를 볼 수 있었으며, Application Server의 CPU사용율도 2배 이상의 효과를 볼 수 있었다. 또한, Database측면과 응용시스템 측면에 대한 다양한 환경설정과 개발을 수행한 후 성능테스트 결과 영상 지도서비스를 수행하는 각 지도 기능들에 대한 평균 응답시간은 모두 1초 이내로 평균 목표 응답시간을 만족하였다. 이와 같이 Database와 응용 시스템에 대한 튜닝 작업과 개발을 통하여 시스템 성능 향상을 위한 별도의 상업 엔진 등을 구입하지 않고도 사용자들에게 충분한 속도향상 효과를 나타낼 수 있었다. 차후, 영상지도와 같이 일정기간 데이터의 변경 없이 사용하는 기본 데이터에 대하여 Tiled Map Service를 구축하여 지도서비스를 하면 시스템의 성능이 더욱 향상 될 것으로 판단된다.

감사의 글

본 연구는 국토해양부 첨단도시기술개발사업-지능형국토정보기술혁신사업과제의 연구비지원(07 국토정보C02)에 의해 수행되었습니다. **KAGIS**

참고 문헌

박현철, 김형섭, 조명희. 2005. Web GIS를 이용한 연안위험취약지역 정보시스템 구축. 한국지리정보학회지 8(4):155-164.

오정연. 2007. 새로운 GIS 패러다임 Where 2.0에 주목하라. 정보사회 현안 분석. 한국정보사회진흥원. 15-28쪽.

장진주, 박세호, 이보미, 최원근, 류민주, 염유영. 2007. Google Earth 와 Sketch Up을 이용하여 지역 개발과정에서 Public Participation을 가능하게 하는 Web Portal Site의 발전모델 연구. GIS KOREA 2007 발표집. 227-241쪽.

조명희, 김준범, 김현식, 조운원. 2002. 웹지리정보시스템 기술을 이용한 산발 현황정보 관리 시스템 개발. 한국지리정보학회지 5(4):93-105.

정명훈, 윤홍식, 위광재. 2000. 수치 사진측량에 있어서 정합 강도 측정에 의한 불량 정합점 제거에 관한 연구. 한국측량학회지 18(2):191-198.

Seokchan Yun. 2007. The User-Participated Geospatial Web as Open Platform. Paper presented at the Proceeding of the 11th International Seminar on GIS.Scotland, pp.33-48.

ESRI Technical Team. 2005. Managing a Raster Database. An ESRI Technical Paper. pp.4-28.

ESRI Technical Team 2005. Raster Data in ArcSDE 9.1. An ESRI White Paper. pp.2-11.

[http://tomcat.apache.org /index. html](http://tomcat.apache.org/index.html) 