원격지 소프트웨어의 객체정보를 활용한 로컬 소프트웨어로의 복구 방법에 관한 연구

송호섭*
*포스코 ICT SW 융합기술팀
e-mail : webcus@poscoict.com

A Study On Method Of Local Software Restore Using Remote Software Object Information

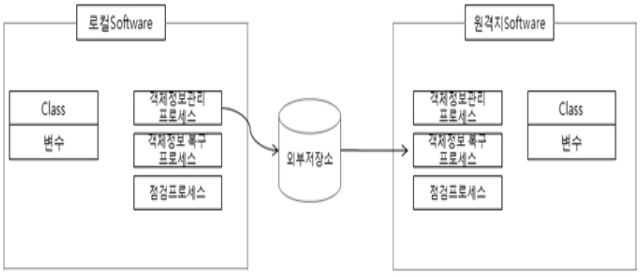
Ho-Seop Song*
*SW Convergence Technology Team POSCO ICT

요 약

수행중인 로컬 소프트웨어의 내부 실행상태를 실시간으로 원격지의 소프트웨어로 반영하여 예상치 못한 장애 상황에서 실행 중이던 소프트웨어를 끝까지 수행 할 수 있도록 복구 방법에 관한 연구 이다. 본 연구에서 수행한 방법은 수시로 소프트웨어 내부 상태 정보를 외부의 저장 서버로 전송하게 되며 원격에 존재하는 소프트웨어에서 그 저장 정보를 이용하여 실시간으로 최신까지의 상태를 업데이트 하여 원래의 소프트웨어 상태를 유지하게 된다. 갑작스런 하드웨어 장애 발생시 상태 점검 프로세스에 의해서 점검되며, 장시간에 걸쳐 수행되고 있는 소프트웨어를 다시 처음부터 수행하지 않고 실시간 정보를 통하여 계속적으로 수행을 완료 할 수 있게 된다. 장시간 동안 시뮬레이션 하는 시스템에서 적용 될 수 있으며 특히 OS 종류나 Server 종류에 종속 되지 않고 동작하며 소프트웨어 개발에 있어서도 프로그램 랭귀지에 상관 없이 구현 할 수 있다.

1. 서론

장시간 수행되는 시뮬레이션과 같은 시스템에선 로컬소프트웨어의 객체정보를 로컬저장소에 스트림 형식으로 저장을 한다. 이의 데이터는 OS 및 서버종류에 대하여 호환성이 없으며, 반드시 수행된 하드웨어와 똑 같은 사양의 시스템에서만 객체정보가 복구 되어진다. 하드웨어적으로 장애가 발생시에는 저장된 데이터를 활용할 수 없어 소프트웨어의 계속 수행이 어렵게 되어진다. 본 연구에서 수행한 시스템은 소프트웨어의 내부 객체 정보를 객체정보관리 프로세스에 의해서 외부 DB 로 관리 되며, 서버의 실행 상태를 점검하여 장애 발생시에 로컬 소프트웨어 및 타 지역의 소프트웨어로 실행상태를 복구 시켜 계속 수행을 할 수 있다.



<그림 1> 시스템 구성

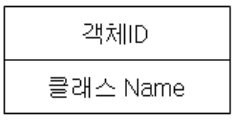
2. 시스템 구성

시스템의 구성은 <그림 1>과 같이 소프트웨어의 내부객체정보를 전송관리 하는 소프트웨어의 "객체 정보 관리 프로세스", 소프트웨어 객체정보를 복구하는 "복구 프로세스", 소프트웨어의 실행 상태를 점검하는 "점검 프로세스"의 세가지로 구성 되어 있다.

가) 객체정보 관리 프로세스

Class 에 정의된 객체정보를 관리하며 그 값들을 원격 DB 에 저장하는 구현 규칙이다.
이 구현 규칙에 의하여 소프트웨어의 내부 객체들을 관리하는 Class 를 구성 할 수 있다.

1. Class 는 Class Name 을 관리 하는 변수가 있어야 한다. 또한 객체의 ID 를 관리하는 변수가 있어야 한다.
<그림 2>의 객체정보를 저장 한다.



<그림 2> 객체정보

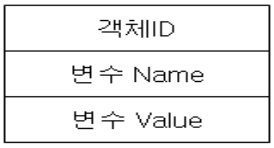
2. 객체를 생성 할 때에는 고유의 ID 를 발생하게 하여야 하며 그 ID 를 관리하여야 한다.

3. 소프트웨어 구현시에 동적 생성되는 모든 객체의 주소(포인터)를 관리하는 배열이 있어야 하며 아래와 같은 규칙이 있다.

a) 동적 생성할 때마다 이 배열에 주소 값을 넣어서 관리 하여야 한다.
b) 객체가 삭제될 때는 이 배열에서 주소 값을 삭제하여야 한다.

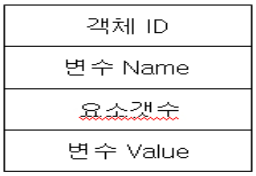
4. 각각의 Class 의 저장기능을 담당하는 멤버함수는 아래의 규칙에 의하여 구현 되어져야 한다.

a) Class 선언부에서 선언 되어진 변수들의 값을 순차적으로 처리한다. <그림 3>



<그림 3> 단순변수정보

b) 단순 변수가 아닌 배열이나 구조체 또는 템플릿형 변수들은 그 내부의 값을 모조리 하나하나 끄집어 내어 순차적으로 저장을 한다.
<그림 4>의 복합변수 정보를 저장한다.



<그림 4> 복합변수정보

5. 실행중인 소프트웨어의 객체 정보를 저장하기위한 명령을 내리는 루틴에서는 아래와 같은 규칙으로 구현 되어져야 한다.

이 루틴에서 각각의 Class 에 존재하는
“ 저장멤버함수”를 호출하게 된다.

a) 위 3 번에서 기술된 배열에 저장된 포인터 들을 차례로 꺼내어 그 객체들의 “ 저장멤버함수” 들을 차례로 실행한다.
b) 클래스에 존재하지 않는 변수들 중에 저장하여야 하는 항목들을 저장한다.

나) 객체정보 복구 프로세스
저장된 데이터를 이용하여 객체들을 복구하며 그 객체들의 멤버변수들에 값들을 원복시키는 방법이다. 다음의 구현 규칙이 있다.

1. Software 수행후에 복구를 수행하기 위하여 복구명령을 내리는 루틴이 있어야 하며 이 루틴에서 우선적으로 주소(포인터)를 관리하는 배열에 객체정보들을 복구시켜야 한다.
아래와 같은 규칙을 따른다.

a) 객체들의 포인터를 관리하는 배열을 초기화 한다.
b) 저장 데이타로부터 객체들의 상태값들을 저장할때의 순서와 동일한 순서로 복구한다.
c) 데이터에 저장된 Class 이름을 이용하여 객체를 생성한 후 배열에 포인터를 넣어준다.

2. 주소(포인터)를 관리하는 배열에 포함된 객체 주소를 순차적으로 꺼내어 그 객체의 “ 복구멤버함수” 를 수행하여 객체의 멤버변수들의 값을 복원한다.

3. 각각의 Class 의 복구기능을 하는 멤버함수는 아래의 규칙에 의하여 구현 되어져야 한다.

a) “ 저장멤버함수” 에서 저장되는 순서 그대로 복구순서를 정하여 구현한다.
복구하여야 할 객체 구분은 Object ID 를 참조한다.
b) 단순 변수가 아닌 배열이나 구조체 또는 템플릿형 변수들은 복구하면서 그 값을 넣어준다. 복구하여야 할 객체 구분은 Object ID 를 참조한다.

다) 점검 프로세스
점검 프로세스는 소프트웨어의 실행 상태를 모니터링하며, 복구 요청시에 복구 수행루틴을 구동 시킨다. 로컬 소프트웨어의 실행여부를 원격지의 소프트웨어로 주기적으로 전송하며, 원격지에서 모니터링 결과 장애 발생하였다면 복구 프로세스를 수행한다.
복구 프로세스를 수행시에 DB 에 저장된 객체정보를 가져와 그 값으로 소프트웨어의 모든 상태를 복구하게 된다.

3. 시스템 동작 및 특징

"객체정보 관리프로세스"에 의하여 내부 객체 상태가 주기적 또는 비주기적으로 원격지의 데이타 저장소로 저장 되어진다.

"점검 프로세스"에 의하여 원격지의 소프트웨어가 계속적으로 실행상태에 있는지 점검을 하게 되며, 점검결과 하드웨어 장애라 판단되면 원격지 소프트웨어의 "객체정보 복구프로세스"를 수행시켜 원격지로 소프트웨어 상태가 전이 되어 계속적으로 수행 할 수 있다.

복구시스템이 적용된 소프트웨어는 로컬과 리모트 양쪽에 존재하게 되며, DB 에 의해서 객체정보를 공유하게 된다. 로컬서버에 존재하는 소프트웨어가 수행을 담당하며, 주기적으로 DB 에 객체정보를 전송하게 된다.
또한 실행 상태를 원격지 소프트웨어로 주기적으로 알리며, 원격지에 존재하는 소프트웨어는 실행상태에 대한 정보만 보고 받게 된다. 즉 원격지 소프트웨어는 소프트웨어 수행은 하지 않고 상대방 소프트웨어의 실행 상태만 감시 하게 된다.
감시 결과, 일정 시간이 지나도 실행상태에 대한 보고가 이루어 지지 않을경우 그 원격지 소프트웨어는 복구 프로세스가 동작하여 DB 로부터 객체정보를 가져와 내부 상태정보를 복구하여 프로그램 수행을 이어서 진행 하게 된다.

로컬서버의 하드웨어적 고장시에는 저장된 스트림형 데이타가 소용없게 된다. 이런점을 개선하기 위하여 원격지 DB 에 저장을 하며, 이때 데이타는 하드웨어 종류나 OS 종류 프로그램언어와는 상관없이 호환성을 갖게 된다. 그러므로 원격지 복구서버는 OS,서버종류와 상관없이 구성을 할 수 있다.

복구되어 실행되는 서버의 위치는 네트워크환경이 제공되는 어디서나 실행가능하며 장시간 수행되는 시뮬레이션 소프트웨어처럼 중단없이 수행되어져야 하는 소프트웨어에 적용하면, 장애발생시에 수행시간을 절약 할 수 있게 된다.

4. 결론

본 연구에서 수행된 시스템은 장시간 수행되는 소프트웨어에 적용하여, 중단시점부터 이어서 계속 수행 할 수 있게 하는 시스템으로써, 소프트웨어 구현시에 제시된 방법에 의하여 구현을 하게 된다. 그러므로, 목적하는 소프트웨어에 제시된 저장 및 복구 방법이 추가되며, 하드웨어적으로 이중화하여 원격 DB 와 함께 전체 시스템이 구성된다.
소프트웨어 객체정보의 저장 방법은 여러 방법이 이미 알려져 있으나, 로컬시스템에서 복구를 하여야 하며, 스트림으로 네트웍을 통하여 전송하여도, 전송중에 끊어진다면 복구 할 수 있는 방법이 없다.
이는 객체정보의 DB 공유를 통하여 해결되며, DB 로의 객체정보를 정확하게 전송하는 방법과, DB 에서 꺼내온 객체정보를 정확하게 복구하는 방법이 본 시스템의 핵심이다.
복구 시점을 주기적으로 체크하여 자동으로 복구 하는 것이 기본 방법이지만, 이를 자동으로 하지 않고 알람 기능을 추가하여 관리자에 의하여 수동으로 복구 할 수 있게 구현을 할 수도 있다.
소프트웨어의 수행을 안정적으로 보장하는 방안의 하나로, 제시된 연구의 결론을 짓는다.

참고문헌

[1] 이창섭, DEVS 기반의 실시간 워게임 시뮬레이션을 위한 저장 및 복구 방법론, KAIST, 2006.

[2] 윤종현, 객체기반 저장시스템을 위한 백업시스템 설계 및 구현, 한국정보처리학회, 2006

[3] 김명식, 객체 직렬화를 이용한 시스템 설계 및 구축 , 한남대학교, 2005