(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl.
G06F 11/00 (2006.01)
G06F 15/16 (2006.01)

(45) 공고일자 2007년05월11일
(11) 등록번호 10-0717241
(24) 등록일자 2007년05월04일

(21) 출원번호 10-2005-0070726
(22) 출원일자 2005년08월02일
심사청구일자 2005년08월02일

(65) 공개번호 10-2007-0015999
(43) 공개일자 2007년02월07일

(73) 특허권자 엔에이치엔(주)
경기도 성남시 분당구 정자동 25-1 분당벤처타운

(72) 발명자 황민재
서울시 성북구 장위1동 231-287

박현우
서울시 강남구 대치4동 889-56번지 The Nine 1008호

(74) 대리인 천성진

(56) 선행기술조사문헌
KR1020010079612 A
KR1019990054594 A
KR1020030035181 A

심사관 : 안철용

전체 청구항 수 : 총 11 항

(54) 에러 관리 시스템 및 이를 이용한 에러 관리 방법

(57) 요약

본 발명은 에러 관리 시스템 및 방법에 관한 것으로, 특히 서버 어플리케이션과 클라이언트 어플리케이션으로 구성된 온라인 서비스 시스템에 있어서 클라이언트 어플리케이션에서 발생하는 에러를 수집하고 관리하는 에러 관리 시스템 및 방법에 관한 것이다. 본 발명에 따른 클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 시스템은, 상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하고, 수집한 에러 정보 및 크래쉬 덤프 파일을 네트워크를 통하여 소정의 포맷으로 송신하는 클라이언트 단말기 및 상기 클라이언트 어플리케이션에 대응한 서비스를 제공하는 서버 어플리케이션을 수행하는 온라인 서비스 서버와 독립되고, 상기 클라이언트 단말기로부터 송신된 에러 정보 및 크래쉬 덤프 파일을 수신하여 온라인 서비스 별로 저장하는 에러 수집 서버를 포함하는 것을 특징으로 한다. 따라서, 보다 효과적으로 클라이언트 단말기에서 발생하는 클라이언트 어플리케이션의 에러를 수집하고, 이를 관리할 수 있다.

대표도

도 2

특허청구의 범위

청구항 1.

클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 시스템에 있어서,

상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우, 운영체제에 의해 발생되는 이벤트에 대해 이벤트 핸들러에서 필요한 기능을 수행하여 에러 정보 및 크래쉬 덤프 파일을 수집하고, 수집한 에러 정보 및 크래쉬 덤프 파일을 네트워크를 통하여 소정의 포맷으로 송신하는 클라이언트 단말기;

상기 클라이언트 어플리케이션에 대응한 서비스를 제공하는 서버 어플리케이션을 수행하는 온라인 서비스 서버와 독립되고, 상기 클라이언트 단말기로부터 송신된 에러 정보 및 크래쉬 덤프 파일을 수신하여 온라인 서비스 별로 저장하는 에러 수집 서버; 및

상기 네트워크를 통하여 상기 에러 수집 서버에 접속하여 상기 에러 수집 서버에 저장된 에러 정보 및 크래쉬 덤프 파일을 제공 받는 에러 관리 단말기

를 포함하는 것을 특징으로 하는 에러 관리 시스템.

청구항 2.

삭제

청구항 3.

제1항에 있어서,

상기 클라이언트 단말기는

상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 상기 에러 정보 및 크래쉬 덤프 파일을 수집하여 상기 클라이언트 단말기 내에 저장하고, 상기 클라이언트 어플리케이션이 다시 시작되면 저장된 상기 에러 정보 및 크래쉬 덤프 파일을 상기 네트워크를 통하여 상기 소정의 포맷으로 송신하는 것을 특징으로 하는 에러 관리 시스템.

청구항 4.

제1항에 있어서,

상기 클라이언트 단말기는

상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 상기 에러 정보 및 크래쉬 덤프 파일을 수집하고 즉시 상기 에러 정보 및 크래쉬 덤프 파일을 상기 네트워크를 통하여 상기 소정의 포맷으로 송신하는 것을 특징으로 하는 에러 관리 시스템.

청구항 5.

제1항에 있어서,

상기 클라이언트 단말기는

HTTP 프로토콜을 이용하여 URL 형태로 상기 에러 정보 및 크래쉬 덤프 파일을 송신하는 것을 특징으로 하는 에러 관리 시스템.

청구항 6.

제1항에 있어서,

상기 에러 수집 서버는

상기 에러 정보에 포함된 상기 클라이언트 어플리케이션에서 발생한 에러의 어드레스를 수신하고, 상기 에러의 어드레스 별로 상기 에러 수집 서버에 저장된 에러 정보 및 상기 크래쉬 덤프 파일을 그룹화하여 각 그룹의 에러 발생횟수를 카운트하고, 상기 카운트 결과에 따라 상기 그룹을 순서대로 정렬하는 것을 특징으로 하는 에러 관리 시스템.

청구항 7.

제1항에 있어서,

상기 에러 수집 서버는

상기 에러 정보에 포함된 상기 클라이언트 어플리케이션에서 발생한 에러의 원인을 수신하고, 상기 에러의 원인 별로 상기 에러 수집 서버에 저장된 에러 정보 및 상기 크래쉬 덤프 파일을 그룹화하여 각 그룹의 에러 발생 횟수를 카운트하고, 상기 카운트 결과에 따라 상기 그룹을 순서대로 정렬하는 것을 특징으로 하는 에러 관리 시스템.

청구항 8.

제1항에 있어서,

상기 에러 정보는

상기 클라이언트 단말기의 클라이언트 환경 정보를 포함하는 것을 특징으로 하는 에러 관리 시스템.

청구항 9.

클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 방법에 있어서,

클라이언트 단말기에서 수행되는 클라이언트 어플리케이션이 온라인 서비스 서버에서 수행되는 서버 어플리케이션과 네트워크를 통하여 연동하여 동작하는 단계;

상기 클라이언트 단말기에서, 상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우, 운영체제에 의해 발생되는 이벤트에 대해 이벤트 핸들러에서 필요한 기능을 수행하여 에러 정보 및 크래쉬 덤프 파일을 수집하는 단계;

상기 클라이언트 단말기에서 상기 수집한 에러 정보 및 크래쉬 덤프 파일을 상기 네트워크를 통하여 소정의 포맷으로 에러 수집 서버로 송신하는 단계;

상기 에러 수집 서버에서, 상기 에러 정보 및 크래쉬 덤프 파일을 수신하여 저장하는 단계; 및

상기 네트워크를 통하여 상기 에러 수집 서버에 접속하는 에러 관리 단말기에서, 상기 에러 수집 서버에 저장된 에러 정보 및 크래쉬 덤프 파일을 제공 받는 단계

를 포함하는 것을 특징으로 하는 에러 관리 방법.

청구항 10.

삭제

청구항 11.

제9항에 있어서,

네트워크를 통하여 소정의 포맷으로 에러 수집 서버로 송신하는 상기 단계는,

상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 상기 이벤트를 가로채서 상기 에러 정보 및 크래쉬 덤프 파일을 수집하여 상기 클라이언트 단말기 내에 저장하는 단계; 및

상기 클라이언트 어플리케이션이 다시 시작되면 저장된 상기 에러 정보 및 크래쉬 덤프 파일을 상기 네트워크를 통하여 상기 소정의 포맷으로 송신하는 단계

를 포함하는 것을 특징으로 하는 에러 관리 방법.

청구항 12.

제9항에 있어서,

네트워크를 통하여 소정의 포맷으로 에러 수집 서버로 송신하는 상기 단계는,

HTTP 프로토콜을 이용하여 URL 형태로 상기 에러 정보 및 크래쉬 덤프 파일을 송신하는 것을 특징으로 하는 에러 관리 방법.

청구항 13.

제9항, 제11항, 제12항 중 어느 한 항의 방법을 컴퓨터에서 실행하기 위한 프로그램을 기록하는 컴퓨터 판독 가능한 기록 매체.

명세서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 에러 관리 시스템 및 방법에 관한 것으로, 특히 서버 어플리케이션과 클라이언트 어플리케이션으로 구성된 온라인 서비스 시스템에 있어서 클라이언트 어플리케이션에서 발생하는 에러를 수집하고 관리하는 에러 관리 시스템 및 방법에 관한 것이다.

인터넷이 보편화 되어감에 따라 각각의 사용자 단말에 설치되는 클라이언트 어플리케이션과 서비스 서버에 설치되는 서버 어플리케이션이 네트워크를 통하여 연동되어 온라인 서비스를 제공하는 온라인 서비스 시스템이 보편화 되어가고 있다. 이러한 온라인 서비스의 대표적인 예로 '리니지', '카트라이더' 및 '당신은 골프왕' 등의 온라인 게임을 들 수 있다.

도 1은 종래 기술에 따른 온라인 서비스 시스템의 블록도이다.

도 1을 참조하면, 종래 기술에 따른 온라인 서비스 시스템은 클라이언트 단말기들(110-1, 110-2, ... , 110-N) 및 온라인 서비스 서버(120-1, 120-2, ... , 120-N)를 포함한다.

클라이언트 단말기들(110-1, 110-2, ... ,110-N) 각각은 온라인 서비스 서버들(120-1, 120-2, ... , 120-N) 각각과 인터넷 등의 네트워크를 통하여 연결된다. 이 때, 클라이언트 단말기들(110-1, 110-2, ... , 110-N) 각각에는 온라인 서비스 서버들(120-1, 120-2, ... , 120-N) 중 일부 또는 전부가 제공하는 온라인 서비스를 수행하기 위한 클라이언트 어플리케이션이 설치된다.

예를 들어, 온라인 서비스 서버(120-1)은 '리니지' 게임 서버일 수 있고, 온라인 서비스 서버(120-2)는 '당신은 골프왕' 게임 서버일 수 있다. 이 때, 클라이언트 단말기들(110-1, 110-2, ... , 110-N)에는 모두 '당신은 골프왕' 게임 서비스를 수행하기 위한 '당신은 골프왕' 게임 클라이언트 어플리케이션이 설치될 수 있다.

일반적으로, 클라이언트 단말기들(110-1, 110-2, ... , 110-N)은 그 하드웨어나 운영체제가 천차만별이고, 클라이언트 어플리케이션과 서버 어플리케이션이 연동하여 동작하는 온라인 서비스 시스템의 특성상 도 1에 도시된 온라인 서비스 시스템은 통상의 프로그램 디버깅에 비해서 복잡한 디버깅 작업을 필요로 한다. 즉, 어플리케이션을 개발하는 과정에서는 원활하게 동작하던 클라이언트 어플리케이션이 직접 클라이언트 단말기들(110-1, 110-2, ... , 110-N)로 다운로드 되어 실행되게 되면 예기치 못한 에러가 발생하는 경우가 많다. 또한, 클라이언트 단말기들(110-1, 110-2, ... , 110-N)에서 클라이언트 어플리케이션에 에러가 발생한다고 해도 클라이언트 단말기의 사용자들이 자신의 단말기에서 발생한 에러에 대해서 이를 프로그램 관리자 게시판에 올린다든가 아니면 관리자에게 전자메일을 통하여 통보하는 등의 방법으로 에러를 신고하지 않으면 프로그램 개발자 측에서 에러 발생 여부를 알기 어렵고, 에러 발생 여부를 안다고 하더라도 그 에러가 특정 클라이언트 단말기 환경에 기인한 것일 수 있어 그 에러를 정확히 재현해 내기 어렵다.

예를 들어, 온라인 서비스 서버(120-2)가 '당신은 골프왕' 게임 서버라고 할 때, 게임 프로그램 개발자는 가상의 클라이언트를 설정하고 온라인 서비스 서버(120-2)와 연동하여 동작시킴으로써 '당신은 골프왕' 게임 테스트를 수행한다. 다양한 클라이언트 환경을 고려한 가상의 클라이언트에서 테스트를 수행하여 모든 에러를 디버깅한 후에 비로소 프로그램 개발자는 클라이언트 단말기들(110-1, 110-2, ... , 110-N)이 게임 어플리케이션을 다운로드 하여 온라인 서비스 서버(120-2) 내의 서버 어플리케이션과 연동하여 동작하도록 한다. 그러나, 대부분의 경우에 테스트 시에 발생하지 아니하던 에러가 클라이언트 단말기들(110-1, 110-2, ... , 110-N)에서 발생하며, 클라이언트 단말기들(110-1, 110-2, ... , 110-N)의 사용자들의 불만은 쌓여만 가고, 온라인 서비스 서버(120-2) 측에서는 에러의 발생 유무와 에러의 발생 원인조차도 알기 어려우며, 에러 발생 유무를 안다고 하더라도 에러 발생시 클라이언트 단말기의 상태를 알 수 없어 디버깅에 어려움이 있다.

따라서, 보다 효과적으로 클라이언트 단말기에서 발생하는 클라이언트 어플리케이션의 에러를 수집하고, 이를 관리할 수 있는 에러 관리 시스템 및 방법의 필요성이 절실하게 대두된다.

발명이 이루고자 하는 기술적 과제

본 발명은 상술한 바와 같은 종래기술의 문제점을 해결하기 위해 안출된 것으로서, 클라이언트 어플리케이션에서 발생하는 에러를 효율적으로 수집하고, 이를 일목요연하게 관리할 수 있는 에러 관리 시스템 및 에러 관리 방법을 제공하는 것을 목적으로 한다.

또한, 본 발명은 클라이언트 어플리케이션에서 에러가 발생하는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하여, 이를 송신함으로써 별도의 에러 신고 절차 없이 편리하게 클라이언트 어플리케이션의 에러를 수집할 수 있는 에러 관리 시스템 및 에러 관리 방법을 제공하는 것을 목적으로 한다.

또한, 본 발명은 서로 다른 온라인 서비스를 제공하는 서비스 제공자들이 각각 소정의 포맷에 맞추어 클라이언트 어플리케이션의 에러를 에러 수집 서버로 전송하기만 하면 네트워크를 통하여 에러 수집 서버에 저장된 에러 정보 및 크래쉬 덤프 파일을 용이하게 제공 받을 수 있는 에러 관리 시스템 및 에러 관리 방법을 제공하는 것을 목적으로 한다.

또한, 본 발명은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 일차적으로 에러 정보 및 크래쉬 덤프 파일을 수집하여 클라이언트 단말기 내에 저장하고, 후에 클라이언트 어플리케이션이 다시 시작되면 저장된 에러 정보 및 크래쉬 덤프 파일을 네트워크를 통하여 송신함으로써 안정적으로 에러 정보 및 크래쉬 덤프 파일을 송신할 수 있는 에러 관리 방법을 제공하는 것을 목적으로 한다.

또한, 본 발명은 HTTP 프로토콜을 이용하여 에러 정보 및 크래쉬 덤프 파일을 송신함으로써 다양한 플랫폼 환경에서 누구나 용이하게 적용할 수 있고, 공유기나 방화벽에 의한 송신 차단을 방지할 수 있는 에러 관리 시스템 및 에러 관리 방법을 제공하는 것을 목적으로 한다.

또한, 본 발명은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 에러 정보와 크래쉬 덤프 파일을 송신하도록 함으로써 클라이언트 단말기에서 발생한 에러를 효과적으로 재현할 수 있는 에러 관리 시스템 및 에러 관리 방법을 제공하는 것을 목적으로 한다.

나아가, 본 발명은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 클라이언트 단말기의 클라이언트 환경 정보를 송신함으로써 에러가 많이 발생하는 클라이언트 환경을 파악할 수 있도록 하는 것을 목적으로 한다.

발명의 구성

상기의 목적을 달성하고 종래기술의 문제점을 해결하기 위하여, 본 발명의 일 실시예에 따른 클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 시스템은, 상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하고, 수집한 에러 정보 및 크래쉬 덤프 파일을 네트워크를 통하여 소정의 포맷으로 송신하는 클라이언트 단말기 및 상기 클라이언트 어플리케이션에 대응한 서비스를 제공하는 서버 어플리케이션을 수행하는 온라인 서비스 서버와 독립되고, 상기 클라이언트 단말기로부터 송신된 에러 정보 및 크래쉬 덤프 파일을 수신하여 온라인 서비스 별로 저장하는 에러 수집 서버를 포함하는 것을 특징으로 한다.

또한, 본 발명의 다른 실시예에 따른 클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 방법은, 클라이언트 단말기에서 수행되는 클라이언트 어플리케이션과 온라인 서비스 서버에서 수행되는 서버 어플리케이션이 네트워크를 통하여 연동하여 동작하는 단계, 상기 클라이언트 단말기에 설치되는 에러 처리 모듈이, 상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하고 네트워크를 통하여 소정의 포맷으로 송신하는 단계 및 에러 수집 서버가, 상기 네트워크를 통하여 송신된 에러 정보 및 크래쉬 덤프 파일을 수신하여 저장하는 단계를 포함하는 것을 특징으로 한다.

또한, 본 발명의 또 다른 실시예에 따른 클라이언트 어플리케이션에서 발생하는 에러를 송신하는 에러 관리 방법은, 클라이언트 단말기에서 수행되는 클라이언트 어플리케이션과 온라인 서비스 서버에서 수행되는 서버 어플리케이션이 네트워크를 통하여 연동하여 동작하는 단계 및 상기 클라이언트 단말기에 설치되는 에러 처리 모듈이, 상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하고 네트워크를 통하여 소정의 포맷으로 송신하는 단계를 포함하는 것을 특징으로 한다.

이 때, 클라이언트 어플리케이션이 비정상적으로 종료되는 경우는 어플리케이션이 동작을 더 이상 지속할 수 없어 소정의 이벤트가 발생하는 것을 뜻한다. 이 때, 소정의 이벤트는 운영체제(Operating System; OS)에서 제공되는 것일 수 있다.

이 때, 이벤트를 가로챈다 함은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 발생하는 이벤트에 대한 이벤트 핸들러(event handler)에서 필요한 기능을 수행하도록 함을 뜻한다.

이 때, 에러 정보는 클라이언트 어플리케이션이 비정상적으로 종료되는 시점의 시간, 종료 이유, 메모리 어드레스, 콜스택(call stack), 관련 모듈 정보, 레지스터의 상태 및 해당 프로그램의 버전정보 등의 전부 또는 일부를 포함할 수 있다.

이하, 본 발명에 따른 바람직한 실시예를 첨부된 도면을 참조하여 상세하게 설명한다.

도 2는 본 발명의 일 실시예에 따른 에러 관리 시스템의 블록도이다.

도 2를 참조하면, 본 발명의 일 실시예에 따른 에러 관리 시스템은 클라이언트 단말기들(210-1, ... , 210-N), 온라인 서비스 서버들(220-1, ... , 220-N) 및 에러 수집 서버(230)를 포함한다.

클라이언트 단말기들은 컴퓨터, 포켓 PC, 노트북 컴퓨터, PDA, 휴대폰 및 게임기 등일 수 있다.

클라이언트 단말기들(210-1, ... , 210-N) 각각에는 클라이언트 어플리케이션이 설치된다. 각각의 클라이언트 단말기들(210-1, ... , 210-N)에는 둘 이상의 클라이언트 어플리케이션이 설치될 수 있다.

각각의 클라이언트 어플리케이션은 네트워크를 통하여 온라인 서비스 서버들(220-1, ... , 220-N)에 설치되는 서버 어플리케이션과 연동하여 동작한다. 예를 들어, 클라이언트 단말기(210-1)에는 '당신은 골프왕' 게임 서비스를 위한 게임 클라이언트 및 '리니지' 게임 서비스를 위한 게임 클라이언트가 설치될 수 있다. 이 때, '당신은 골프왕' 게임 서비스는 온라인 서비스 서버(220-1)에 설치된 서버 어플리케이션과 연동하여 동작하고, '리니지' 게임 서비스는 온라인 서비스 서버(220-2)에 설치된 서버 어플리케이션과 연동하여 동작할 수 있다.

온라인 서비스 서버들(220-1, ... , 220-N)은 각각 하나의 온라인 서비스를 위한 서버 어플리케이션을 수행할 수 있고, 두 개 이상의 온라인 서버들이 함께 하나의 온라인 서비스를 위한 서버 어플리케이션을 수행할 수도 있다. 예를 들어, 온라인 서비스 서버들(220-1, 220-2)은 모두 '당신은 골프왕' 게임 서비스를 위한 서버일 수 있다.

또한, 클라이언트 단말기들(210-1, ... , 210-N) 각각에는 에러 처리 모듈이 설치될 수 있다. 에러 처리 모듈은 클라이언트 단말기에서 수행되는 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하고, 수집한 에러 정보 및 크래쉬 덤프 파일을 네트워크를 통하여 소정의 포맷으로 송신한다.

이 때, 클라이언트 어플리케이션이 비정상적으로 종료된다 함은 클라이언트 어플리케이션이 더 이상 동작을 지속할 수 없어 운영체제(operating system; OS) 등이 설정한 소정의 이벤트(또는 exception)가 발생하는 것을 뜻한다. 이벤트(event)는 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 운영체제에 의하여 발생될 수 있다. 이 때, 운영체제는 마이크로소프트사의 윈도 계열, 애플사의 맥OS계열, 팜(Palm)의 PalmOS계열 또는 각종 실시간 운영체제(Real Time OS; RTOS)일 수 있다.

예를 들어, 에러 처리 모듈은 이벤트 핸들러(event handler)에서 소정의 동작을 수행하도록 작성된 컴퓨터 프로그램 코드일 수 있다.

에러 정보는 클라이언트 어플리케이션이 비정상적으로 종료된 시점의 시스템 시간, 에러가 발생한 어드레스 및 에러의 원인 등을 포함할 수 있다. 예를 들어, 에러의 원인은 "Access Violation"이나, "Divide By Zero" 등일 수 있다. 또한, 에러 정보는 콜스택(call stack), 레지스터 정보, 에러 발생 당시의 해당 프로그램 버전 정보 및 에러 발생 당시 해당 프로그램이 사용하고 있던 모듈 관련 정보를 포함할 수 있다.

콜스택(call stack)은 에러 발생 당시 해당 프로그램에서 함수들 사이의 호출 관계를 나타낸다. 예를 들어, 콜스택은 main 함수에서 Print함수를 호출하고, 다시 Print함수에서 GetData함수를 호출하는 관계일 수 있다. 이 때, 에러 정보에 포함되는 콜스택은 각 함수의 주소 값만을 전송할 수 있다. 에러 정보를 수신하는 수신측에서는 모듈 관련 정보 등을 이용하여 전송된 주소에 해당하는 함수를 알아낼 수 있다. 예를 들어, 콜스택으로 "Ox04001000 Ox04000700 Ox04000103"이 송신되면 수신측에서는 모듈 관련 정보 등을 이용하여 'Ox04001000'와 GetData함수, 'Ox04000700'과 Print함수, 'Ox04000103'과 main함수를 각각 대응시킬 수 있다.

레지스터 정보는 에러가 발생할 당시 중앙처리장치(Central Processing Unit; CPU)의 레지스터에 저장되어있는 데이터들을 포함할 수 있다. 예를 들어, Intel사의 X86 CPU를 사용하는 경우에 레지스터 정보는, EAX, EBX, ECX, EDX, ESI, EDI, EBP, EIP 등의 레지스터에 저장되어 있는 데이터들을 포함할 수 있다. 레지스터 정보는 에러 발생 당시의 정황을 알아내는데 매우 유용하게 사용될 수 있다.

모듈 관련 정보는 모듈의 이름, 모듈의 체크섬 및 모듈이 로드 된 메모리 주소 등을 포함할 수 있다.

실시예에 따라, 에러 정보는 클라이언트 단말기의 클라이언트 환경 정보를 포함할 수 있다. 이 때, 클라이언트 환경 정보는 중앙처리장치(Central Processing Unit; CPU)나 그래픽 카드 등 하드웨어의 종류 및 하드웨어의 구동에 사용되는 디바이스 드라이버의 버전 등을 포함할 수 있다. 따라서, 에러 정보를 수신하는 수신측에서 어떤 하드웨어 조합에서 에러가 많이 발생하는지를 파악하여 하드웨어 등의 제조사 등과 협조하여 효과적으로 에러를 디버깅할 수 있다.

크래쉬 덤프 파일(crash dump file)은 어플리케이션이 동작하다가 에러가 발생하여 작동이 멈추는 경우에 메모리 안에 저장된 정보 등 에러가 발생한 상황을 그대로 파일로 저장한 것이다. 따라서, 시스템의 어떤 부분에서 어떤 이유로 에러가 발생하였는지를 알아내는 데에 크래쉬 덤프 파일(crash dump file)이 매우 유용하게 활용될 수 있다. 예를 들어, 크래쉬 덤프 파일은 마이크로소프트사의 운영체제에 의해서 생성되는 것일 수 있다.

에러 처리 모듈은 클라이언트 어플리케이션 별로 따로 구비될 수도 있고, 둘 이상의 클라이언트 어플리케이션이 하나의 에러 처리 모듈을 공유할 수도 있다. 예를 들어, '당신은 골프왕' 어플리케이션과 '리니지' 어플리케이션이 각각 상응하는 에러 처리 모듈을 구비할 수도 있고, '당신은 골프왕' 어플리케이션과 '리니지' 어플리케이션이 하나의 에러 처리 모듈을 공유할 수도 있다.

도 3은 도 2에 도시된 클라이언트 단말기의 일 예를 나타낸 블록도이다.

도 3을 참조하면, 도 2에 도시된 클라이언트 단말기(210-1)는 클라이언트 어플리케이션들(311, 321) 및 에러 처리 모듈들(312, 322)을 포함한다.

에러 처리 모듈(312)은 클라이언트 어플리케이션(311)에 대응되어 클라이언트 어플리케이션(311)에서 발생하는 에러를 처리하고, 에러 처리 모듈(322)은 클라이언트 어플리케이션(321)에 대응되어 클라이언트 어플리케이션(321)에서 발생하는 에러를 처리한다.

도 3에는 두 개의 클라이언트 어플리케이션들(311, 321)이 설치된 경우를 예로 들었으나, 클라이언트 단말기(210-1)는 세 개 이상의 클라이언트 어플리케이션들을 포함할 수도 있고, 단 하나의 클라이언트 어플리케이션만을 포함할 수도 있다. 도 3에 도시된 예에서 에러 처리 모듈은 클라이언트 어플리케이션마다 구비되므로 클라이언트 어플리케이션의 수만큼 구비된다.

도 4는 도 2에 도시된 클라이언트 단말기의 다른 예를 나타낸 블록도이다.

도 4를 참조하면, 도 2에 도시된 클라이언트 단말기(210-1)는 클라이언트 어플리케이션들(411, 421) 및 에러 처리 모듈(430)을 포함한다.

에러 처리 모듈(430)은 클라이언트 어플리케이션(411) 및 클라이언트 어플리케이션(421)에서 발생하는 에러를 처리한다. 즉, 클라이언트 어플리케이션(411) 및 클라이언트 어플리케이션(421)은 에러 처리 모듈(430)을 공유한다.

도 4에는 두 개의 클라이언트 어플리케이션들(411, 421)이 설치된 경우를 예로 들었으나, 클라이언트 단말기(210-1)는 세 개 이상의 클라이언트 어플리케이션들을 포함할 수도 있고, 단 하나의 클라이언트 어플리케이션만을 포함할 수도 있다. 또한, 세 개 이상의 클라이언트 어플리케이션이 하나의 에러 처리 모듈을 공유할 수도 있다.

이상에서 도 2에 도시된 클라이언트 단말기(210-1)를 예로 들어 설명하였으나, 도 2에 도시된 다른 클라이언트 단말기들(210-2, ... , 210-N)에도 클라이언트 단말기(210-1)에 대한 설명이 그대로 적용될 수 있다.

실시예에 따라, 에러 처리 모듈은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 일단 상기 에러 정보 및 크래쉬 덤프 파일을 수집하여 상기 클라이언트 단말기 내에 저장하고, 상기 클라이언트 어플리케이션이 다시 시작되면 저장된 상기 에러 정보 및 크래쉬 덤프 파일을 상기 네트워크를 통하여 상기 소정의 포맷으로 송신할 수 있다. 클라이언트 어플리케이션이 비정상적으로 종료되면 시스템이 불안정할 가능성이 높으므로 일단 에러 정보 및 크래쉬 덤프 파일을 수집하여 클라이언트 단말기 내에 저장하여 놓고, 후에 다시 클라이언트 어플리케이션이 시작되면 저장된 에러 정보 및 크래쉬 덤프 파일을 송신하여 안정적으로 에러 정보 및 크래쉬 덤프 파일을 송신할 수 있다.

또한, 에러 처리 모듈은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 에러 정보 및 크래쉬 덤프 파일을 수집하여 이를 즉시 네트워크를 통하여 소정의 포맷으로 송신할 수도 있다.

이 때, 네트워크는 유/무선 인터넷일 수 있다.

에러 처리 모듈은 HTTP(HyperText Transfer Protocol)을 이용하여 상기 에러 정보 및 크래쉬 덤프 파일을 전송할 수 있다. 이 때, 사용되는 포트는 80포트일 수 있다. 이와 같이, HTTP프로토콜 및 80포트를 이용하여 어떤 클라이언트 모델에서도 손쉽게 에러 정보를 보낼 수 있고, 공유기나 방화벽에 의하여 송신이 차단되는 것을 방지할 수 있다.

나아가, HTTP프로토콜을 이용하여 에러 정보 및 크래쉬 덤프 파일을 전송함으로써 통상적으로 사용되는 URL(Uniform Resource Locator) 형태로 에러 정보 및 크래쉬 덤프 파일을 전송할 수 있다.

다시 도 2를 참조하면, 에러 수집 서버(230)는 유/무선 인터넷 등의 네트워크를 통하여 에러 처리 모듈로부터 송신된 에러 정보 및 크래쉬 덤프 파일을 수신하여 저장한다.

이 때, 에러 수집 서버(230)는 오라클 데이터베이스 등의 별도의 데이터베이스에 네트워크를 통하여 송신된 에러 정보 및 크래쉬 덤프 파일을 저장할 수 있다.

이 때, 에러 수집 서버(230)는 클라이언트 어플리케이션에서 발생한 에러의 어드레스 별로 에러 수집 서버(230)에 저장된 에러 정보 및 크래쉬 덤프 파일을 정렬할 수도 있고, 클라이언트 어플리케이션에서 발생한 에러의 원인 별로 에러 수집 서버(230)에 저장된 에러 정보 및 크래쉬 덤프 파일을 정렬할 수도 있다.

이 때, 에러 수집 서버(230)는 상기 에러 정보에 포함된 클라이언트 어플리케이션에서 발생한 에러의 어드레스를 수신하고, 상기 에러의 어드레스 별로 에러 수집 서버(230)에 저장된 에러 정보 및 크래쉬 덤프 파일을 그룹화하여 각 그룹의 에러 발생횟수를 카운트하고, 상기 카운트 결과에 따라 상기 그룹을 순서대로 정렬할 수 있다.

이 때, 에러 수집 서버(230)는 상기 에러 정보에 포함된 클라이언트 어플리케이션에서 발생한 에러의 원인을 수신하고, 상기 에러의 원인 별로 에러 수집 서버(230)에 저장된 에러 정보 및 크래쉬 덤프 파일을 그룹화하여 각 그룹의 에러 발생 횟수를 카운트하고, 상기 카운트 결과에 따라 상기 그룹을 순서대로 정렬할 수 있다.

따라서, 클라이언트 단말기들(210-1, ... , 210-N)에 포함된 클라이언트 어플리케이션에서 에러가 발생하여 클라이언트 어플리케이션이 비정상적으로 종료될 때마다 에러 처리 모듈로부터 자동적으로 에러 정보 및 크래쉬 덤프 파일이 전송되어 에러 수집 서버(230)에서 수집된다.

이 때, 에러 관리 시스템은 도 2에는 도시되지 아니하였으나 네트워크를 통하여 상기 에러 수집 서버(230)에 접속하여 에러 수집 서버(230)에 저장된 에러 정보 및 크래쉬 덤프 파일을 제공 받는 에러 관리 단말기를 더 포함할 수 있다. 이 때, 에러 관리 단말기는 컴퓨터, 포켓 PC, 노트북 컴퓨터, PDA, 휴대폰 및 게임기 등일 수 있다. 따라서, 다양한 온라인 서비스에 대한 각종 에러를 에러 수집 서버(230)에서 통일적으로 수집/관리하고, 인터넷 등을 통하여 관리자가 용이하게 누적된 에러 정보 및 크래쉬 덤프 파일에 접근할 수 있다.

도 5는 에러 수집 서버에 저장된 에러 정보를 어드레스 별로 분류한 화면을 나타낸 도면이다.

도 5를 참조하면, 에러 수집 서버에 저장된 에러 정보가 에러가 많이 발생한 어드레스부터 순서대로 정렬되어 있는 것을 알 수 있다. 도 5에 도시된 바와 같이, 에러가 많이 발생한 순서대로 에러가 발생한 어드레스(510), 에러의 원인(520), 콜스택(530), 레지스터 정보(540), 프로그램 버전 정보(550), 모듈 관련 정보(560), 크래쉬 덤프 파일 링크(570) 및 에러 발생 횟수(580)가 표시된다.

에러가 발생한 어드레스(510)는 "10018BB9", "1000E119" 등과 같이 에러가 발생한 메모리 어드레스를 나타낸다.

에러의 원인(520)은 "EXCEPTION_FLT_DIVIDE_BY_ZERO", "EXCEPTION_ACCESS_VIOLATION" 등과 같이 에러가 발생한 이유 내지 에러의 종류를 나타낸다. 에러의 원인(520)은 클라이언트 컴퓨터에 설치된 운영체제에 따라 달라질 수 있다.

콜스택(call stack)(530)은 에러 발생 당시 해당 프로그램에서 함수들 사이의 호출 관계를 나타낸다. 도 5에 도시된 예에서 콜스택은 각 함수의 주소 값을 나타낸다.

레지스터 정보(540)는 에러 발생 당시에 클라이언트 컴퓨터의 레지스터 내에 저장된 데이터들을 나타낸다. 예를 들어, 레지스터 정보(540)는 "EAX:FFFFFFE2 EBX:029D2420 ECX:00000000 EDX:0000001E ESI:021EC1D8 EDI:0000000F EBP:00000000 EIP:00000000 SegCs:0000001B EFlags:00010286 Esp:00000008 SegGS:00000000" 등과 같이 표시될 수 있다.

프로그램 버전 정보(550)는 에러가 발생할 당시의 해당 프로그램의 버전 정보를 나타낸다.

모듈 관련 정보(560)는 에러 발생 당시의 해당 프로그램이 사용하고 있던 모듈 관련 정보를 나타낸다. 모듈 관련 정보(560)는 모듈의 이름, 모듈의 체크섬, 모듈이 로드 된 메모리 주소 등을 포함할 수 있다.

크래쉬 덤프 파일 링크(570)는 해당하는 크래쉬 덤프 파일을 다운로드 받을 수 있는 링크일 수 있다.

이와 같이, 사용자가 간단하게 에러 수집 서버에 접속하여 원하는 에러 정보에 접근하여 원하는 형태로 정렬할 수 있어 클라이언트 어플리케이션에서 발생하는 에러를 일목요연하게 관리할 수 있다. 나아가, 프로그래머는 발생 빈도가 높은 에러의 순으로 디버깅할 수 있어 효율적으로 클라이언트 어플리케이션의 에러를 디버깅할 수 있다.

도 6은 본 발명의 일 실시예에 따른 에러 관리 방법을 나타낸 동작 흐름도이다.

도 6을 참조하면, 본 발명의 일 실시예에 따른 에러 관리 방법은 클라이언트 단말기에서 수행되는 클라이언트 어플리케이션이 온라인 서비스 서버에서 수행되는 서버 어플리케이션과 네트워크를 통하여 연동하여 동작한다(S610).

이 때, 클라이언트 어플리케이션이 설치된 클라이언트 단말기가 두 개 이상일 수 있고, 각기 다른 서버 어플리케이션을 제공하는 온라인 서비스 서버들이 두 개 이상일 수 있다. 또한, 하나의 클라이언트 단말기에 설치되는 클라이언트 어플리케이션도 두 개 이상일 수 있다.

클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 방법은 상기 클라이언트 단말기에 설치되는 에러 처리 모듈이, 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하고 네트워크를 통하여 소정의 포맷으로 송신한다(S620).

이 때, 에러 처리 모듈은 각각의 클라이언트 어플리케이션에 대응되도록 설정될 수도 있고, 두 개 이상의 클라이언트 어플리케이션이 하나의 에러 처리 모듈을 공유할 수도 있다.

이 때, 이벤트를 가로챈다 함은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 발생하는 이벤트에 대한 이벤트 핸들러(event handler)에서 필요한 기능을 수행하도록 함을 뜻한다. 즉, 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 발생하는 이벤트에 대한 이벤트 핸들러에는, 에러 정보 및 크래쉬 덤프 파일을 수집하고 네트워크를 통하여 소정의 포맷으로 송신하는 기능이 부가될 수 있다.

이 때, 단계(S620)는 상기 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 상기 이벤트를 가로채서 상기 에러 정보 및 크래쉬 덤프 파일을 수집하여 상기 클라이언트 단말기 내에 저장하는 단계 및 상기 클라이언트 어플리케이션이 다시 시작되면 저장된 상기 에러 정보 및 크래쉬 덤프 파일을 상기 네트워크를 통하여 상기 소정의 포맷으로 송신하는 단계를 포함하여 단계적으로 수행될 수도 있다.

또한, 네트워크를 통한 에러 정보 및 크래쉬 덤프 파일의 송신은 HTTP(HyperText Transfer Protocol)이용하여 80포트로 송신될 수 있다. 따라서, 어떤 클라이언트 모델에서도 손쉽게 에러 정보를 보낼 수 있고, 공유기나 방화벽에 의하여 송신이 차단되는 것을 방지할 수 있고, 통상적으로 사용되는 URL(Uniform Resource Locator) 형태로 에러 정보 및 크래쉬 덤프 파일을 전송할 수 있다.

에러 정보 및 크래쉬 덤프 파일이 송신되면 클라이언트 어플리케이션에서 발생하는 에러를 관리하는 에러 관리 방법은 에러 수집 서버에서 상기 네트워크를 통하여 송신된 에러 정보 및 크래쉬 덤프 파일을 수신하여 저장한다(S630).

이 때, 클라이언트 어플리케이션에서 발생한 에러의 어드레스 별로 상기 에러 수집 서버에 저장된 에러 정보 및 크래쉬 덤프 파일이 정렬될 수도 있고, 클라이언트 어플리케이션에서 발생한 에러의 원인 별로 상기 에러 수집 서버(230)에 저장된 에러 정보 및 크래쉬 덤프 파일이 정렬될 수도 있다.

도 6에 도시된 각 단계는 도 6에 도시된 순서, 그 역순 또는 동시에 수행될 수 있다.

도 6을 통하여 설명한 방법과 관련하여 설명하지 아니한 내용은 앞서 에러 관리 시스템에 관한 실시예들에서 그대로 적용 가능하므로 이하 생략한다.

본 발명에 따른 에러 관리 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 상기 매체는 프로그램 명령, 데이터 구조 등을 지정하는 신호를 전송하는 반송파를 포함하는 광 또는 금속선, 도파관 등의 전송 매체일 수도 있다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기된 하드웨어 장치는 본 발명의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

도 7는 본 발명에 따른 에러 관리 방법을 수행하는 데 채용될 수 있는 범용 컴퓨터 장치의 내부 블록도이다.

컴퓨터 장치(700)는 램(RAM: Random Access Memory)(720)과 롬(ROM: Read Only Memory)(730)을 포함하는 주기억장치와 연결되는 하나 이상의 프로세서(710)를 포함한다. 프로세서(710)는 중앙처리장치(CPU)로 불리기도 한다. 본 기술분야에서 널리 알려져 있는 바와 같이, 롬(730)은 데이터(data)와 명령(instruction)을 단방향성으로 CPU에 전송하는 역할을 하며, 램(720)은 통상적으로 데이터와 명령을 양방향성으로 전송하는 데 사용된다. 램(720) 및 롬(730)은 컴퓨터 판독 가능 매체의 어떠한 적절한 형태를 포함할 수 있다. 대용량 기억장치(Mass Storage)(740)는 양방향성으로 프로세서(710)와 연결되어 추가적인 데이터 저장 능력을 제공하며, 상기된 컴퓨터 판독 가능 기록 매체 중 어떠한 것일 수 있다. 대용량 기억장치(740)는 프로그램, 데이터 등을 저장하는데 사용되며, 통상적으로 주기억장치보다 속도가 느린 하드 디스크와 같은 보조기억장치이다. CD 롬(760)과 같은 특정 대용량 기억장치가 사용될 수도 있다. 프로세서(710)는 비디오 모니터, 트랙볼, 마우스, 키보드, 마이크로폰, 터치스크린 형 디스플레이, 카드 판독기, 자기 또는 종이 테이프 판독기, 음성 또는 필기 인식기, 조이스틱, 또는 기타 공지된 컴퓨터 입출력장치와 같은 하나 이상의 입출력 인터페이스(750)와 연결된다. 마지막으로, 프로세서(710)는 네트워크 인터페이스(770)를 통하여 유선 또는 무선 통신 네트워크에 연결될 수 있다. 이러한 네트워크 연결을 통하여 상기된 방법의 절차를 수행할 수 있다. 상기된 장치 및 도구는 컴퓨터 하드웨어 및 소프트웨어 기술 분야의 당업자에게 잘 알려져 있다. 한편, 상기된 하드웨어 장치는 본 발명의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있다.

이상과 같이 본 발명은 비록 한정된 실시예와 도면에 의해 설명되었으나, 본 발명은 상기의 실시예에 한정되는 것은 아니며, 본 발명이 속하는 분야에서 통상의 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다.

그러므로, 본 발명의 범위는 설명된 실시예에 국한되어 정해져서는 아니 되며, 후술하는 특허청구범위뿐 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

발명의 효과

본 발명의 에러 관리 시스템 및 에러 관리 방법은 클라이언트 어플리케이션에서 발생하는 에러를 효율적으로 수집하고, 이를 일목요연하게 관리할 수 있다.

또한, 본 발명은 클라이언트 어플리케이션에서 에러가 발생하는 경우에 이벤트를 가로채서 에러 정보 및 크래쉬 덤프 파일을 수집하여, 이를 송신함으로써 별도의 에러 신고 절차 없이 편리하게 클라이언트 어플리케이션의 에러를 수집할 수 있다.

또한, 본 발명은 서로 다른 온라인 서비스를 제공하는 서비스 제공자들이 각각 소정의 포맷에 맞추어 클라이언트 어플리케이션의 에러를 에러 수집 서버로 전송하기만 하면 네트워크를 통하여 에러 수집 서버에 저장된 에러 정보 및 크래쉬 덤프 파일을 용이하게 제공 받을 수 있다.

또한, 본 발명은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 일차적으로 에러 정보 및 크래쉬 덤프 파일을 수집하여 클라이언트 단말기 내에 저장하고, 후에 클라이언트 어플리케이션이 다시 시작되면 저장된 에러 정보 및 크래쉬 덤프 파일을 네트워크를 통하여 송신함으로써 안정적으로 에러 정보 및 크래쉬 덤프 파일을 송신할 수 있다.

또한, 본 발명은 HTTP 프로토콜을 이용하여 에러 정보 및 크래쉬 덤프 파일을 송신함으로써 다양한 플랫폼 환경에서 누구나 용이하게 적용할 수 있고, 공유기나 방화벽에 의한 송신 차단을 방지할 수 있다.

또한, 본 발명은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 에러 정보와 크래쉬 덤프 파일을 송신하도록 함으로써 클라이언트 단말기에서 발생한 에러를 효과적으로 재현할 수 있다.

나아가, 본 발명은 클라이언트 어플리케이션이 비정상적으로 종료되는 경우에 클라이언트 단말기의 클라이언트 환경 정보를 송신함으로써 에러가 많이 발생하는 클라이언트 환경을 파악할 수 있다.

도면의 간단한 설명

도 1은 종래 기술에 따른 온라인 서비스 시스템의 블록도이다.

도 2는 본 발명의 일 실시예에 따른 에러 관리 시스템의 블록도이다.

도 3은 도 2에 도시된 클라이언트 단말기의 일 예를 나타낸 블록도이다.

도 4는 도 2에 도시된 클라이언트 단말기의 다른 예를 나타낸 블록도이다.

도 5는 에러 수집 서버에 저장된 에러 정보를 어드레스 별로 분류한 화면을 나타낸 도면이다.

도 6은 본 발명의 일 실시예에 따른 에러 관리 방법을 나타낸 동작 흐름도이다.

도 7은 본 발명에 따른 에러 관리 방법을 수행하는 데 채용될 수 있는 범용 컴퓨터 장치의 내부 블록도이다.

<도면의 주요 부분에 대한 부호의 설명>

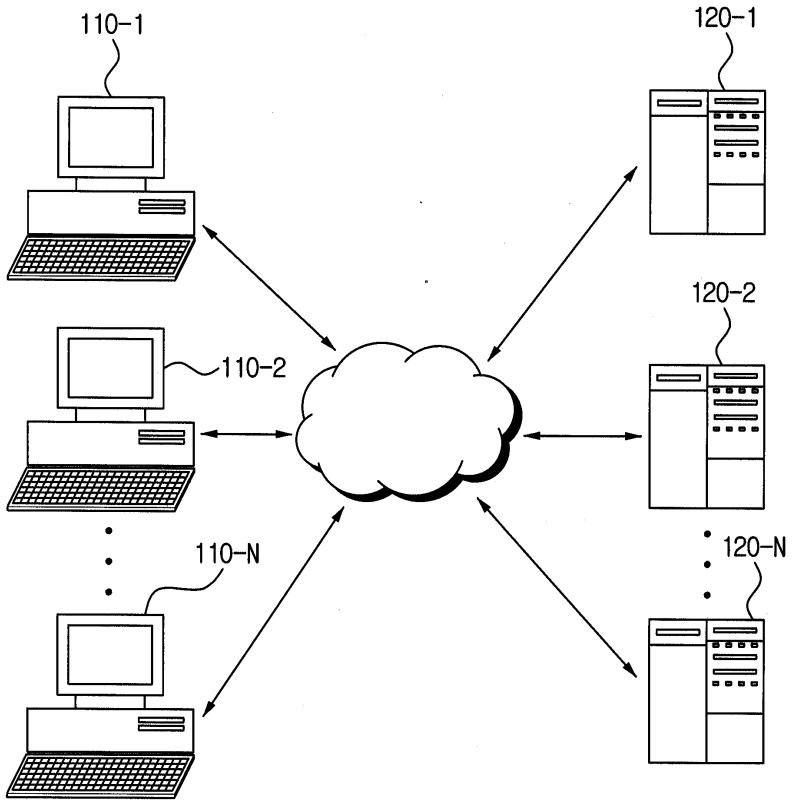
210-1 ~ 210-N: 클라이언트 단말기들

220-1 ~ 220-N: 온라인 서비스 서버들

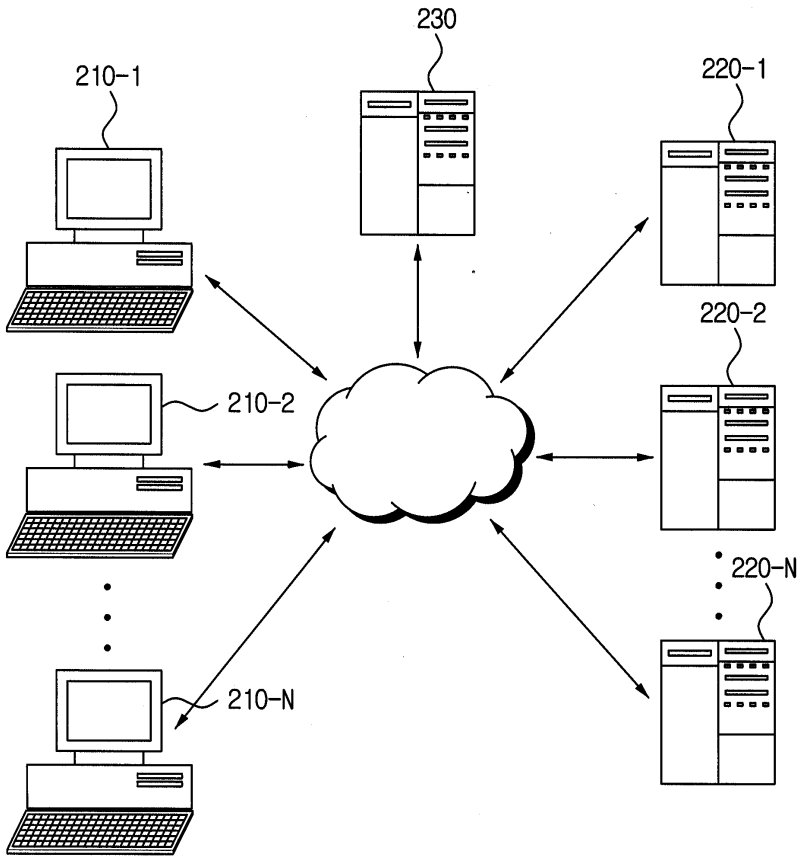
230: 에러 수집 서버

도면

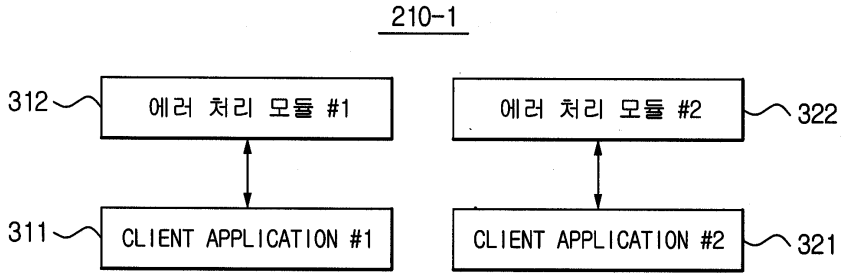
도면1


110-1
110-2
110-N
120-1
120-2
120-N

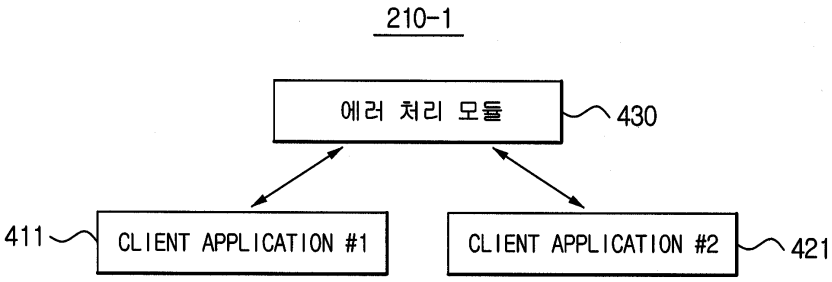
도면2


230
210-1
210-2
210-N
220-1
220-2
220-N

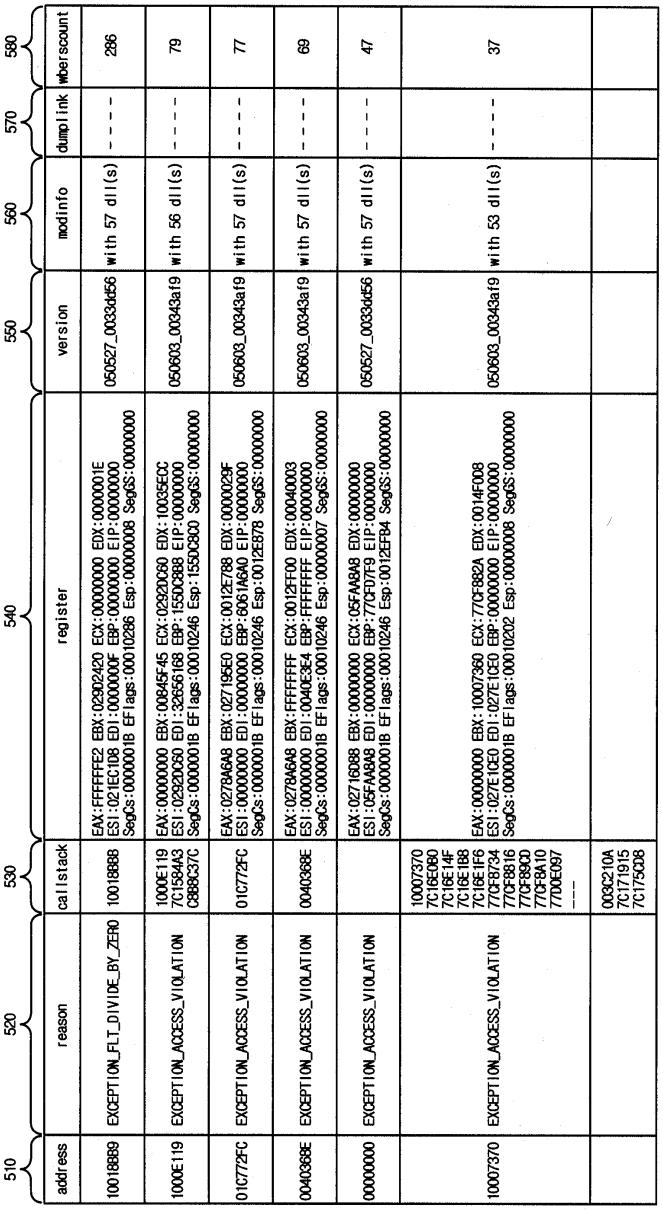
도면3


210-1
312
에러 처리 모듈 #1
에러 처리 모듈 #2
322
311
CLIENT APPLICATION #1
CLIENT APPLICATION #2
321

도면4

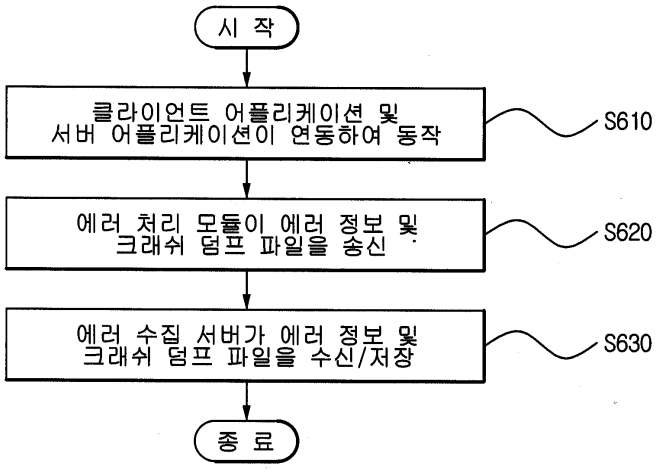

210-1
에러 처리 모듈
430
411
CLIENT APPLICATION #1
CLIENT APPLICATION #2
421

도면5

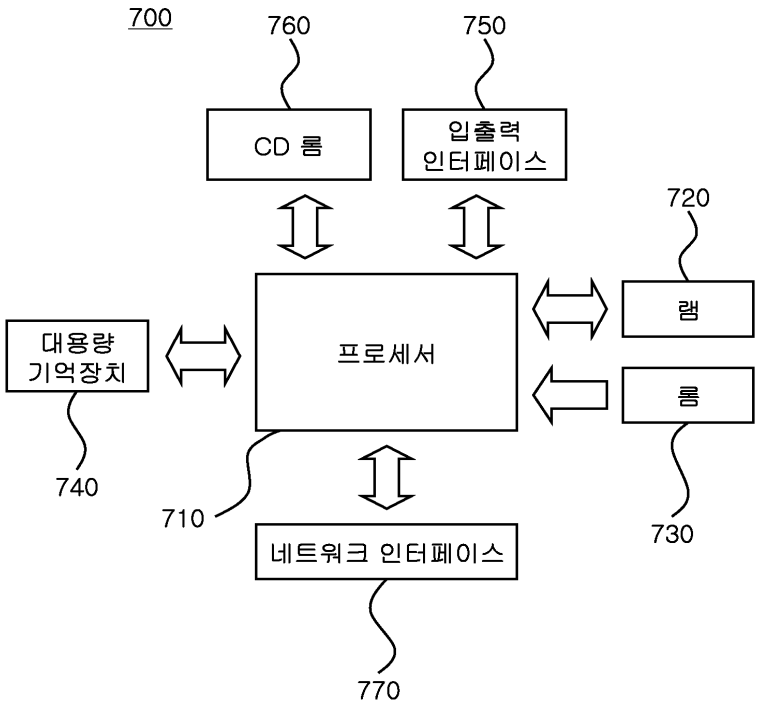


address	reason	callstack	register	version	modinfo	dumplink	wberscount
10018889	EXCEPTION_FLT_DIVIDE_BY_ZERO	10018888	EAX:FFFFFFE2 EBX:02902420 ECX:00000000 EDX:0000001E ESI:021EC108 EDI:0000000F EBP:00000000 EIP:00000000 SegCs:0000001B EFlags:00010286 Esp:00000008 SegSS:00000000	050527_0033dd56	with 57 dll(s)	- - - -	286
1000E119	EXCEPTION_ACCESS_VIOLATION	1000E119 7C15844A3 C888C37C	EAX:00000000 EBX:00845F45 ECX:02920C60 EDX:10035EOC ESI:02920C50 EDI:32656168 EBP:1550C888 EIP:00000000 SegCs:0000001B EFlags:00010246 Esp:1550C8C0 SegSS:00000000	050603_00343af9	with 56 dll(s)	- - - -	79
01C772FC	EXCEPTION_ACCESS_VIOLATION	01C772FC	EAX:0278A6A8 EBX:027195E0 ECX:0012E788 EDX:0000029F ESI:00000000 EDI:00000000 EBP:8061A6A0 EIP:00000000 SegCs:0000001B EFlags:00010246 Esp:0012E878 SegSS:00000000	050603_00343af9	with 57 dll(s)	- - - -	77
0040368E	EXCEPTION_ACCESS_VIOLATION	0040368E	EAX:0278A6A8 EBX:FFFFFFFF ECX:0012FF00 EDX:00040003 ESI:00000000 EDI:0040E3C4 EBP:FFFFFFFF EIP:00000000 SegCs:0000001B EFlags:00010246 Esp:00000007 SegSS:00000000	050603_00343af9	with 57 dll(s)	- - - -	69
00000000	EXCEPTION_ACCESS_VIOLATION		EAX:02716D88 EBX:00000000 ECX:05FAA9A8 EDX:00000000 ESI:05FAA9A8 EDI:00000000 EBP:77CFD7F9 EIP:00000000 SegCs:0000001B EFlags:00010246 Esp:0012EFB4 SegSS:00000000	050527_0033dd56	with 57 dll(s)	- - - -	47
10007370	EXCEPTION_ACCESS_VIOLATION	10007370 7C16E0B0 7C16E14F 7C16E188 7C16E1F6 77CF8734 77CF8816 77CF89D0 77CF8A10 7700E097 ---	EAX:00000000 EBX:10007360 ECX:77CF882A EDX:0014F008 ESI:027E1CE0 EDI:027E1CE0 EBP:00000000 EIP:00000000 SegCs:0000001B EFlags:00010202 Esp:00000008 SegSS:00000000	050603_00343af9	with 53 dll(s)	- - - -	37
		0036210A 7C171915 7C175C08					

도면6



도면7


700
760
750
CD 롬
입출력
인터페이스
720
램
대용량
기억장치
프로세서
롬
740
710
730
네트워크 인터페이스
770