멀티미디어 컴퓨터 지원 협력 작업에서 오류 공유 에이전트에 관한 연구

고 응 남
천안대학교 정보통신학부

요 약

본 논문은 ESA(Error Sharing Agent)의 설계와 구축을 설명한다. ESA는 멀티미디어 협동 작업 환경에서 소프트웨어 오류를 감지, 공유, 복구하기에 적합한 시스템이다. 이 시스템은 ED, ES, ER로 구성되어 있다. ED는 훅킹 기법으로 오류를 감지한다. ES는 멀티미디어 공동 작업 환경에서 오류를 공유하는 에이전트이다. 이 시스템에 의해서 오류를 공유할 수 있다. 멀티미디어 공동 작업 환경의 관점에서 오류 공유는 협동 작업에 참가하는 참가자에게 상호작용적으로 오류를 공유한다. 본 논문에서 멀티미디어 공동 작업 환경에서 하나의 오류 응용을 미디어로 취급하는 방법을 제안한다.

A Study on an Error Sharing Agent for Multimedia CSCW

Eung-Nam Ko

ABSTRACT

This paper explains the design and implementation of the ESA(Error Sharing Agent). ESA is a system that is suitable for detecting, sharing and recovering software error based on multimedia CSCW(Computer Supportes Cooperated Work). This system consists of an ED, ES, and ER. ED detects an error by hooking techniques. ES is an agent which is an error sharing system for distributed multimedia collaboration environment With error sharing system, a group cooperating users can share error applications. From the perspective of multimedia collaborative environment, an error application becomes another interactive presentation error is shared with participants engaged in a cooperative work. We propose how an error application can be treated as 'media' in a multimedia collaborative environment.

Ⅰ. 서 론

멀티미디어 공동 작업 서비스들과 응용들은 점점 더 중요하게 되고 있으며 복잡하고 실시간 멀티미디어 지원, 높은 대역폭 가용성 및 짧은 지연을 위한 요구가 증가하고 있다[1]. 사회가 복잡해지고 컴퓨터 네트워크가 발달함에 따라 다양한 종류의 상호 참여가 요구되어지고 있다. 이에 따라 상호 참여기능을 제공하기 위한 화상 회의 시스템의 개발이 활발해지고 있다[2]. 최근에 있었던 협동 작업 분야의 많은 연구와 발전에 힘입어 의료 및 교육을 포함하는 다양한 분야에서 컴퓨터를 이용하는 협동작업에 대한 요구가 날로 커지고 있다[3,4]. 협동작업은 전자메일을 통한 협력작업과 같이 한곳에서 생성한 결과를 나중에 다른 곳에서 이용하며 협력 작업이 진행되는 비동기형(asynchronous) 작업과 채팅, 화이트보드, 화상 회의와 같이 공동 작업을 위한 협동이 동시에 이루어지는 동기형(synchronous) 작업으로 구분할 수 있다[3,5]. 응용 공유 시스템은 공유되는 응용 프로그램의 출력을 공유 세션에 참여하는 사용자들에게 분배하며, 사용자들이 발생시킨 사건을 공유되는 응용으로 입력할 수 있도록 허가함으로써 단일 사용자용으로 개발된 윈도우 시스템 기반의 응용 프로그램을 멀티미디어 분산 공동 작업 환경에서 공동작업 도구로 사용할 수 있도록 지원한다[6,7]. 이러한 현재의 방향에도 불구하고 기존의 멀티미디어 컴퓨터 지원 협력 작업에서는 상호 작용하는 멀티미디어 환경의 구성 요소에서 충분한 신뢰성을 항상 보장하는 것은 아니다[8]. 인터넷 보급이 급격하게 증가함에 따라 인터넷을 활용한 다양한 솔루션들이 소개되고 있다. 최근 들어 초고속 통신망 구축이 현실화되면서 멀티미디어를 포함하는 컨텐츠가 더욱 다양화되어 가고 있다. 그 예로서, 인터넷을 활용한 전자상거래가 생활의 한 부분으로 사용되고 있으며, 교육에서도 웹기반 교육, 실시간/비실시간 원격교육과 같은 컨텐츠가 개발되고 있는 것이다.

따라서, 본 연구에서는 단일 사용자용으로 개발된 응용을, 이를 가지지 않은 다른 네트워크 사용자들이 같이 사용할 수 있도록 설계된 응용 공유 기능을 지닌 동기형 작업, 즉 멀티미디어 컴퓨터 지원 협력 작업에서의 오류 공유 에이전트를 제안한다. 본 논문의 구성은 2에서 기존 응용공유 시스템에 관련된 연구를 기술하고, 3에서는 제안하는 오류 공유 에이전트에 대해서 기술하고, 4에서는 시스템 평가, 5에서는 결론을 기술한다.

Ⅱ. 기존 응용 공유 시스템

본 절에서는 기존의 응용 공유 시스템의 종류, 응용 공유 방법 및 구조, 한계점 등에 대해서 기술한다.

2.1 기존 응용 공유 시스템

MERMAID[9]는 분산형 응용 공유 구조를 선택하면서, 공유 이벤트의 분배를 이벤트 발송 부분에서 처리함으로써 다양한 응용의 지원을 고려하고 있다. MMConf[10]는 분산형 응용 공유 구조를 선택하였으며, X-윈도우즈를 기반으로 설계되어 있다. Critique[11]은 복제형 응용 공유 구조를 선택하였으며, 여기에서 발생하는 일치화 문제를 해결하는데 중점을 두었다. EMX[12]은 X에 기반을 둔 이기종 컴퓨터 환경에서 응용을 공유할 수 있으며, 모든 사용자들이 공유되는 응용을 완전히 제어할 수 있도록 하는데 중점을 두었다. SCOOT[13]

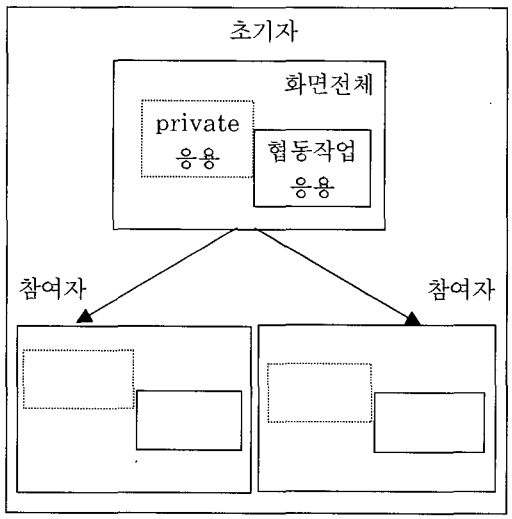
은 기존의 응용 프로그램을 최소한의 수정으로 공동작업에 적합한 응용으로 확장하는 방법에 대해 논의한다. Argo[14]은 프록시 서버를 통해서 기존의 X응용 프로그램을 공유하는데, 특정 응용들만 공유 가능한다. 또한, 여기에서는 윈도우 시스템 기반과 툴 킷 시스템 기반의 복제를 제안하였다. CECED[15]은 중앙 집중형 구조와 복제형 구조의 혼합 구조를 지원하며, 화면 공유 개념을 확장하였다. BERKOM[7]은 어떤 상황 하에서라도 새로운 참여자가 공유 환경에 참여할 수 있는 동적 공유 기능과 암시적 발언권 전달 정책을 사용하였다. XpleXer[16]은 X윈도우 시스템에서 응용 공유를 지원하는데, 선택적 윈도우 공유, 동적 공유 등을 지원한다. QuiX[6]은 중앙 집중형과 복제형 구조를 선택하였으며, 특히 매킨토시와 X 윈도우 시스템 등의 이기종으로 구성된 환경에서의 응용 공유 방법을 제안하였다.

2.2 응용 공유 방법

응용 공유는 공유되는 영역의 범위에 따라서 화면 공유와 윈도우 공유로 나누어지는데, 이는 공유 환경에 참여하는 사용자들이 접근할 수 있는 영역을 정의한다.

(1) 화면 공유

화면 전체를 공유하는 방식으로, 이는 초창기의 원격 응용을 사용하기 위한 방법으로 사용된 방법이다. 하지만, 화면 전체가 공유되기 때문에, 개인적인 부분까지 다른 사용자들이 보거나 수정 할 수 있다는 단점이 있으며, 보안에 있어서도 취약하므로 부적절한 방법이다. 이 방식을 이용한 기초적인 제품들은 다수가 상품화 되어 있으며 (Timbuktu, Carbon Copy, PlanetX, ·), 전형적으로 전체 화면을 공유한다[6]. 그림 1은 화면 공유 방식을 나타낸다. 여기에서 초기자의 화면 전체가 참여자에게도 그대로 공유 가능하기 때문에, 초기자의 개인적인 작업 내용까지 참여자가 접근 가능하다.

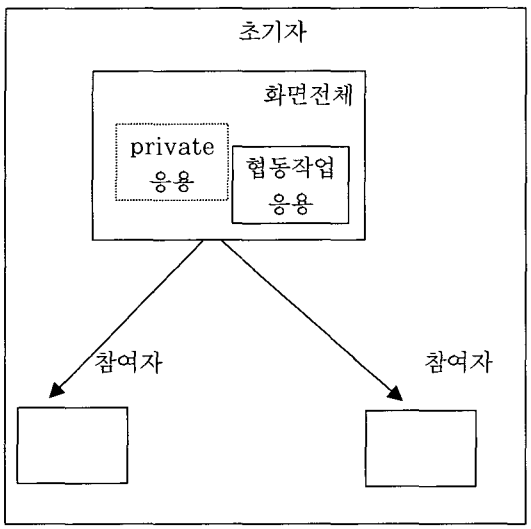


(그림 1) 화면 공유

(2) 윈도우 공유

공유 윈도우만 관리하는 방식은 공동 작업 환경에 참여하는 사용자들에게 각자의 웍스테이션에서 독립적인 작업을 할 수 있도록 보장한다[10].

이는 공유 영역을 윈도우 단위로 설정할 수 있도록 하므로, 각자의 개인적인 공간을 유지 할 수 있으며, 공유가 설정된 윈도우만 다른 참여자들이 접근 할 수 있으므로 보안성의 측면에서도 유리하다. 그림 2는 윈도우 공유 방식을 나타낸다. 여기에서는 초기자가 공유 설정을 한 응용의 윈도우만 참여자들이 접근 가능하므로, 초기자의 개인적인 영역은 그대로 유지된다.



(그림 2) 윈도우 공유

2.3 응용 공유 구조 및 기존 결함 허용 기법의 한계점

응용 공유는 구조에 따라 집중형(Centralized), 분산형(Distributed), 복제형(Replicated)으로 구분할 수 있다. 집중형 구조는 모드 구성 요소가 하나의 웍스테이션에서만 실행되는 구조이다. 이는 모든 요소가 하나의 웍스테이션에서만 실행되기 때문에 가장 간단한 방법이다. 분산형 구조는 구성 요소가 분산되어 있으며, 이들은 여러 웍스테이션에 걸쳐서 존재하는 구조이다. 복제형 구조는 분산형 구조의 변형된 형태로서, 대응하는 구성 요소가 아주 동일하거나 복제된다. 이는 각자의 웍스테이션에 실행에 필요한 모든 요소가 존재하는 구조이다.

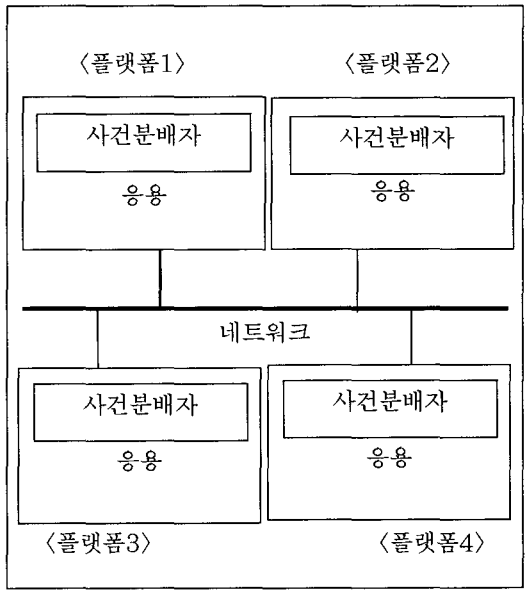
기존 결함 허용 기법은 멀티미디어 컴퓨터 지원 공동 작업 환경을 위한 응용 공유 구조에서의 오류 감지, 전달 등의 방법이 지원되지 않고 있다.

Ⅲ. 오류 공유 에이전트

멀티미디어 컴퓨터 지원 협력 작업 환경을 위한 응용 공유 구조에는 중앙 집중형 구조와 복제형 구조 2가지가 있다. 본 논문에서는 복제형 구조 상에서 오류를 검출하여 빠르게 전달하는 시스템인 ESA(An Error Sharing Agent for Multimedia Computer Supported Cooperative Work)를 제안한다. 오류 검출 시 훅(hook) 기술을 이용하고 오류 전달 시에 오류 공유시스템을 이용하여 오류를 신속히 전달한 후 그 오류를 복구한다.

3.1 ESA의 구조

본 시스템은 응용 프로그램의 복제 본이 모든 사용자들의 워크스테이션에 그림 3과 같이 존재한다.



(그림 3) ESA의 복제형 구조

응용 공유는 응용 프로그램의 재사용을 통해서 기존의 응용을 공동 작업 환경에서 수정 없이 사용하고, 응용 프로그램을 공동 작업 환경에 참여한 사용자들 사이에 공유하는 것을 그 목적으로 한다.

3.2 정의 및 표기

ESA에 대한 설명과 분석을 위해서 필요한 정의 및 표기는 다음과 같다.

(정의 1)

멀티미디어 공동 작업 환경에서 오류 공유 시스템을 ESA라고 표시하면

ESA = 〈 P, L, M, S〉 이다.

여기서 P ={p1, p2, . . . , pn } 이며 프로세스(process)들의 유한 집합(finite set)이다. L ⊆ P2 이며 채널(channel)들의 부분 집합이다.

L = {〈pi,pj〉 | pi :메시지 보내는 프로세스,
pj :메시지 받는 프로세스}

M은 메시지들의 유한 집합이다.

M = {m〈pi,pj〉 | pi :메시지 보내는 프로세스, pj : 메시지 받는 프로세스}

(정의 2)

S는오류 공유 시스템에 관련된 집합이다.

S = {AP, A}이고

AP: 응용 공유 들의 집합,

A: 응용 프로그램 들의 집합이라고 정의 한다.

(정의3)

본 논문에서 오류 감지 및 복구 시스템에 관련되어 있는 에이전트들의 집합은 다음과 같다.

세션이 개설되어 있을 때 여러 플랫폼(platform) 중에서 i번째 플랫폼에 실행하는 오류 감지 및 복구 프로세스들을 ESAi라고 정의한다. 정의된 오류 감지 및 오류 복구에이전트들 ESAi, EDi, ESi 및 ERi 사이의 관계는 다음과 같다. 분할 πESAi = {EDi, ESi, ERi}이고

ESAi = EDi ∪ ESi ∪ ERi (i ∈ N)이다.

(정의 4)

Si(j)는 프로세스 pi가 실행하고 있을 때 그 프로세스 pi에서 j번째 발견되는 공유된 오류(error)들의 집합으로 정의한다.

즉, Si(j)= {si(j) | i ∈ N, j ∈ N }이다.

3.3 ESA의 알고리즘

본 논문에서 제안하는 ESA는 여러 기능의 에이전트가 존재하며 원활한 오류 감지 및 복구 기법을 수행하는 멀티 에이전트 시스템이다. ESA를 구성하는 구성 모듈로는 ED(Error Detection)와 ES(Error Sharing) 및 ER(Error Recovery)이다.

ED는 오류를 감지하는 핵심 에이전트로 고장 감지 정보 흐름은 윈도우의 훅킹(hooking) 방법을 이용하여 그 상태를 분석하여 오류의 발생 여부를 감지한다. 이 과정에서 오류를 감지한 내용, 즉, 포인팅 하는 함수를 가로채서 전달하는 방식이다. ES는 ED로부터 전달 받은 오류를 공유하여 신속하게 전달한다. ER은 ES로부터 전달 받은 오류 정보를 바탕으로 오류를 복구하는 모듈이 실행된다. 이 때 검사점 설정까지 설정된 지점까지 롤백(rollback)하여 복구된다.

본 논문의 범위는 주로 ES에 대하여 기술한다. 오류 전달 방법에 대한 알고리즘은 다음과 같다.

Set of Error Sharing = {Set of fault,
Set of error sharing}

여기서,

Set of fault= {F, S}

- F: 오류의 원인이 되는 고장(fault)
- S: 오류의 공유 여부

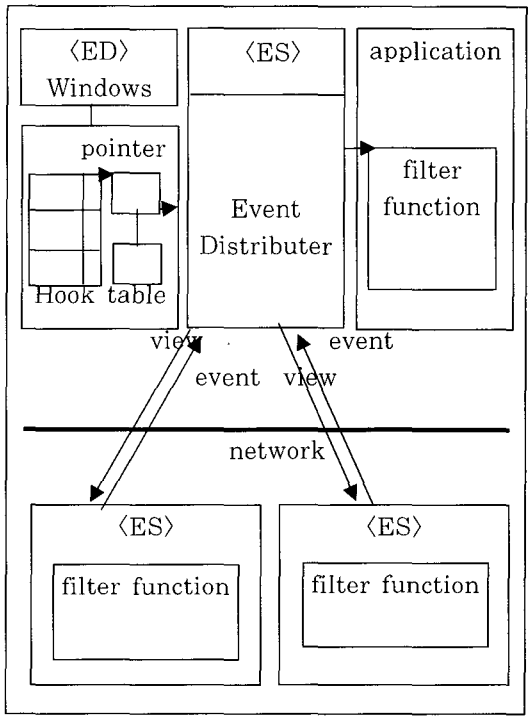
Set of error sharing = {Addr_ES, Method_ES, Func_ES}

- Addr_ES: ES의 주소 정보, 즉 Ei(j) 및 Si(j)에 대한 정보
- Method_ES: 오류 공유 방식으로 오류를 전달
- Func_ES: ES의 기능(function)은 세 집합 P, Ei(j), Si(j)에서 R1을 집합 P에서 Ei(j)로의 관계(relation)라 하고, R2를 집합 Ei(j)에서 Si(j)로의 관계(relation)라 하면, 집합 P에서 Si(j)로의 합성관계 R1R2는 다음과 같이 정의된다.

$$R1R2 = \{(pi,si(j)) \mid pi \in P,\ si(j) \in Si(j),\ (pi,ei(j)) \in R1,\ (ei(j), si(j)) \in R2\}$$

관계 R1에서는 오류를 감지한 내용, 즉, 포인팅하는 함수를 가로채서 전달하는 방식이다. ED는 저널 레코드 또는 쉘 훅 등의 훅 킹 메시지를 사용하여 오류를 감지한다.

관계 R2에서 발생된 오류가 감지되면 오류 공유가 발생한다. 그 사건은 윈도우 메시지 형태로 사건 분배기로 재 지향 되고, 이는 다시 다른 사용자들의 사건 분배기로 네트웍을 통해서 전송된다. 다른 사용자들의 사건 분배기는 수신한 사건을 다시 공유되는 응용 프로그램으로 재지향 한다. 즉, 한 사용자에서 발생된 오류가 다른 사용자들의 공유 응용까지 오류를 전달하는 경로이다. 다른 사용자들의 응용은 각자 전달 받은 오류 사건을 수행해서 수행 결과인 뷰(view)를 화면상에 생성하며, 이를 각자의 화면으로 출력한다. 이렇게 해서 사용자들은 각자의 웍스테이션에 응용 프로그램을 가지고 각자 발생한 오류 사건을 사건 분배기를 통해서 분배하는 방법으로 각자의 응용 프로그램을 가지고 공동 작업을 수행하거나 오류를 인식한다. 오류 공유 과정은 그림 4와 같다.



(그림 4) ESA의 오류 공유 과정

오류 공유에서는 발생된 오류의 메시지 큐의 함수와 포인터를 이벤트 필터가 가로채 실제 응용이 있는 곳으로 전송한다. 응용 프로그램은 이를 처리하여 생성된 결과인 뷰를 생성하게 되며 뷰 변화 감시기는 변화를 감지하여 원격지 윈도우에는 뷰를 압축해서 전달하고, 지역 윈도우에는 그대로 화면에 보낸다.

이런 오류 공유가 가능하게 하기 위해서는 훅 서버, 오류 공유 서버, 오류 공유 인스턴스, 오류 공유 윈도우 관리기, 네트워크 인터페이스, 세션 관리기 인터페이스로 이루어진다.

오류 공유 서버는 오류 공유 에이전트를 설치 및 운용하며, 오류 공유 서버 인스턴스를 생성하고 관리한다. 오류 공유 서버는 오류 공유 에이전트를 생성 종료 시키는 인스턴스 관리기와 데몬과 통신 선로를 개설하는 기능을 가지고 있다. 인스턴스 관리기는 세션 관리기의 요구에 따라서 오류 공유 서버 인스턴스를 생성, 종료시킨다. 데몬 인터페이스는 데몬과의 접속을 담당하는 역할을 하며, 오류 공유 서버와의 정보 교환을 담당한다. 또한, 데몬에서 요청하는 기능을 수행한다. 또한 오류 공유 시스템의 초기화 역할도 담당한다.

오류 공유 에이전트는 이벤트 처리기, 이벤트 재생기 사건 여과기로 구성되어 있다. 오류 공유 에이전트는 윈도우와 응용 사이의 이벤트 큐에 이벤트 처리기와 이벤트 재지향기, 사건 여과기를 설치한다. 이벤트 처리기는 공유된 윈도우에서 사건의 발생 중 오류를 검출한다. 오류가 발생하면 윈도우에서는 훅킹 함수가 발생하게 되는데 이 훅킹 함수는 메시지 큐에 옮겨진다. 메시지 큐로 옮겨진 훅킹 함수들을 이벤트 필터가 읽어내어 사건 재지향기를 통하여 응용 프로그램으로 전달한다. 즉, 정상적인 윈도우에서 오류가 발생하면 훅킹 함수가 발생하고 훅킹 함수를 응용 프로그램에 전달하는데, 전달되기 전에 사건 여과기가 함수를 가로채어 오류 공유 에이전트가 실제 오류가 있는 시스템으로 함수를 전달하여 처리하는 것이다. 오류 공유 에이전트는 참여자의 오류 공유 요청을 받아 사건 처리기, 사건 재지향기 및 이벤트 필터를 실행시킨다.

Ⅳ. 시스템 평가

제안된 시스템은 Visual C++로 설계 및 구축하였다. 오류 감지 및 전달 시에 제안된 방법의 나은 점을 시뮬레이션을 통하여 비교하였다. 본 논문에서는 전체 응용의 가로채는 방법에 대해서 기존 방식 중의 하나인 가로채기(snatch) 방식을 사용하는 방식과 본 논문에서 제안한 훅킹 방법(hooking)을 사용하여 오류 전달 시간을 줄이는 방식 2가지를 비교하여 효율성 검토를 한다. 오류 감지에 대한 효율성 비교는 표 1과 같다.

여기서 훅킹 방법 중에서 쉘 훅을 이용하는 방법이 GDI 가로채기 방식보다 데이터 양이 현저히 적음을 알 수 있고 명령어 사용에도 간단하다.

(표 1) 훅킹 방법과 가로채기 방법과의 비교

	MS 워드		파워포인트	
	이벤트 훅킹	GDI 가로채기	이벤트 훅킹	GDI 가로채기
함수 call		115,541		42,728
쉘 훅	1,124		1,229	
함수 갯수	1,124	115,541	1,229	42,728

표 2는 응용 프로그램 개수와 오류 감지 수행 시간과의 관계를 나타낸 것이다. 만일 프로세스 간의 메시지가 전달될 때 걸리는 시간을 0.01초라고 하면 한번 폴링하는 시간은 0.02초가 된다. 응용 프로그램의 개수를 10, 20, 30, 40개라고 하면 기존 방법의 오류 감지 수행 시간은 각각 0.2, 0.4, 0.6, 0.8이 된다. 훅 킹을 사용한 제안된 방법은 프로그램 개수에 관계없이 각각 0.01씩 된다. 프로세스의 수가 많아질수록 제안된 방식이 더 효율적임을 알 수 있다.

(표 2) 응용 프로그램의 개수와 오류 감지 수행 시간과의 관계

응용프로그램갯수	기존방식	제안된 방식
10	0.19초	0.01초
20	0.41초	0.01초
30	0.58초	0.01초
40	0.79초	0.01초

기존의 시스템과 기능적인 측면을 비교하면 표 3과 같다.

(표 3) 오류 공유 기능 유무 비교

	Shastra	MER-MAID	MMConf	CECED	제안된 논문
오류 공유	지원 안됨	지원 안됨	지원 안됨	지원 안됨	지원
응용 공유	지원 안됨	지원 안됨	지원 안됨	지원	지원

V. 결 론

본 논문에서는 오류 감지, 오류 유형 분류, 전달, 복구 기능 중에서 오류 감지 후에 자동적으로 신속하게 오류를 전달하는 기능을 갖고 있는 에이전트인 ESA를 제안하였다. ESA를 구성하는 구성 모듈로는 ED, ES 및 ER이다. ED는 오류를 감지하는 핵심 에이전트로 고장 감지 정보 흐름은 윈도우의 훅킹(hooking) 방법을 이용하여 그 상태를 분석하여 오류의 발생 여부를 감지하였고 ES는 ED로부터 전달 받은 오류를 공유하여 신속하게 전달하였다. ER은 ES로부터 전달 받은 오류 정보를 바탕으로 오류를 복구하는 모듈이 실행되었다. 본 논문에서 제안한 훅킹 방법(hooking)을 사용하여 효율성 분석을 하였고, 응용 프로그램 개수와 오류 감지 수행 시간과의 관계를 나타내어서 효율성 비교를 하였다. 향 후 연구 과제는 다중 세션이 활성화되어 있는 경우, 네스티드 세션, 웹 환경에서의 오류 감지 및 복구 시스템에 대한 연구 등이다.

참고문헌

[1] Kevin H.Liu and Vassilios Th. Tsaousidis, Efficient Network Management for Collaborative Services and Application Development, 2000 IEEE International Conference on Multimedia and Expo ICME2000, 30 July 2 August 2000 New York, NY USA. pp.53-56.

[2] 전준걸, 황대준, "상호 참여를 위한 탁상회의 시스템의 구현", 95년 한국정보과학회 가을학술 발표논문집 vol.22, No.2, 1995, pp.1041.

[3] 김문석, 성미영, 동기적 웹 브라우저 공유를 지원하는 협동작업 시스템, 한국정보처리학회 정보처리학회논문지B, 제8-B권 제3호, 2001년 6월, pp.283-288.

[4] 궁상환,황승구, Collaborative Computing의 기술 응용, 한국정보과학회 정보과학회지 제 16권 제7호, 1998, pp.5-14.

[5] 이재호, 협력작업을 위한 에이전트 기반 소프트웨어, 한국정보과학회 정보과학회지 제16권 제7 호,1998, pp.24-30.

[6] Klaus H. Wolf and Peter Schulthess, Multimedia Application Sharing in a Heterogeneous Environment, ACM Multimedia95, November 5-9,1995.

[7] Michael Altenhofen and Thomas Steinig, The BERKOM Multimedia Collaboration Service, Proceedings ACM Multimedia 93, August 1-6 1993.

[8] Hiroaki Higaki, Kenji Shima, Takayuki Tachikawa, Makoto Takizawa, Checkpoint and Rollback in Asynchronous Distributed Systems, IEEE INFOCOM97, Proceedings Volume 3.

[9] T. Ohmori and K. Watabe, Distributed Cooperative Control for Application Sharing Based on Multiparty and Multimedia Desktop Conferencing Systems:MERMAID, 4th IEEE ComSoc International Workshop on Multimedia Communications, April 1-4, 1992.

[10] Torrence Crowley and Raymond Tomlinson, MMConf: An Infrastructure for Building Shared Multimedia Applications, CSCW 90 Proceedings, October 1990.

[11] J. Chris Lauwers and Allyn L. Romanow, Replicated Architectures for Shared WindowSystems: A Critique, Proceedings of the Conference on Office Information Systems, March 1990.

[12] Vincent Phuah and Steve Gutfreund, Developing Distributed Multimedia Applications, 4th, IEEE ComSoc International Workshop on Multimedia Communications, April 1-4, 1992.

[13] Earl Craighill and Kathryn Gruenefeldt, SCOOT: An Object-Oriented Toolkit for Multimedia Collaboration, Proceedings ACM Multimedia 94, October 15-20 1994.

[14] Hania Gajewska and David D. Redell, Argo: A System for Distributed Collaboration, Proceedings ACM Multimedia 94, October 15-20 1994.

[15] Earl Craighill and Keith Skinner, CECED: A System For Informal Multimedia Collaboration, Proceedings ACM Multimedia 93, August 1-6 1993.

고 응 남

1984년 연세대 수학과(이학사)
1991년 숭실대 정보과학 대학원 전산공학과 (공학석사)
2000년 성균관대 대학원 정보공학과(공학박사)
1983년-1993년 대우통신 컴퓨터개발부 선임연구원
1993년-1997년 동우대학 전자계산과교수
1997년-2001년 신성대학 컴퓨터계열 교수
2001년-현재 천안대학교 정보통신학부 교수
관심분야 : 인터넷, 멀티미디어, CSCW, 결함허용, 에이전트 및 게임 등