(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2014-0004702
(43) 공개일자 2014년01월13일

(51) 국제특허분류(Int. Cl.)
G06F 9/48 (2006.01) *G06F 11/14* (2006.01)
(21) 출원번호 10-2013-7019989
(22) 출원일자(국제) 2012년02월16일
심사청구일자 없음
(85) 번역문제출일자 2013년07월26일
(86) 국제출원번호 PCT/US2012/025388
(87) 국제공개번호 WO 2012/112748
국제공개일자 2012년08월23일
(30) 우선권주장
13/030,998 2011년02월18일 미국(US)

(71) 출원인
아브 이니티오 테크놀로지 엘엘시
미국 02421 매사추세츠주 렉싱턴 스프링 스트리트 201
(72) 발명자
두로스 브라이언 필
미국 10701 매사추세츠주 프래밍험 레이크뷰 로드 92
홀리 요셉 스케핑튼 3
미국 02478 매사추세츠주 벨몬트 힐크레스트 로드 11
(74) 대리인
유미특허법인

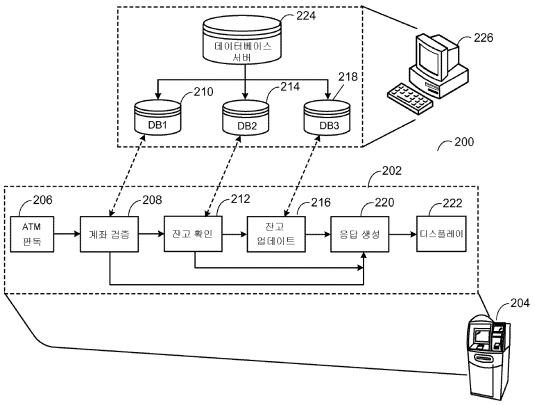
전체 청구항 수 : 총 23 항

(54) 발명의 명칭 **프로세스의 재기동**

(57) 요 약

컴퓨터에 의해 구현되는 방법을 포함하는 기술을 개시하며, 상기 방법은, 초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하는 단계(402)로서, 여기서 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 실행 단계의 실행의 완료 시에 실행 단계의 최종 상태를 나타내는 정보를 저장(404)하는 것을 포함하는, 저장하는 단계와, 사전에 정해진 이벤트에 응답하여 프로세스의 실행을 중단하는 단계(506)와, 프로세스를 셧다운할 필요 없이 저장된 초기 상태와 최종 상태 중의 하나에서부터 프로세스의 실행을 재개하는 단계(512)를 포함한다.

대 표 도



특허청구의 범위

청구항 1

컴퓨터에 의해 구현되는 방법에 있어서,

초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하는 단계로서, 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태(end state)를 나타내는 정보를 저장하는 것을 포함하는, 상기 저장하는 단계;

사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하는 단계; 및

상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하는 단계

를 포함하는 컴퓨터에 의해 구현되는 방법.

청구항 2

제1항에 있어서,

상기 사전에 정해진 이벤트는 외부 장치에 대한 접속의 상실을 나타내는, 컴퓨터에 의해 구현되는 방법.

청구항 3

제1항에 있어서,

상기 사전에 정해진 이벤트는 외부 장치와의 에러를 나타내는, 컴퓨터에 의해 구현되는 방법.

청구항 4

제2항에 있어서,

상기 프로세스의 실행은 상기 외부 장치에 대한 접속이 복구된 때에 재개되는, 컴퓨터에 의해 구현되는 방법.

청구항 5

제3항에 있어서,

상기 프로세스의 실행은 상기 외부 장치와의 에러가 해소된 때에 재개되는, 컴퓨터에 의해 구현되는 방법.

청구항 6

제1항에 있어서,

상기 프로세스의 실행은, 상기 사전에 정해진 이벤트가 발생한 실행 단계 이전에 저장된 최종 상태에서부터 재개되는, 컴퓨터에 의해 구현되는 방법.

청구항 7

제1항에 있어서,

상기 프로세스의 실행은, 실질적으로 상기 프로세스의 기동 직후의 실행 단계 동안에 상기 사전에 정해진 이벤트가 발생하는 경우에는 상기 초기 상태에서부터 재개되는, 컴퓨터에 의해 구현되는 방법.

청구항 8

제1항에 있어서,

상기 실행 단계의 실행은, 상기 실행 단계에 대응하는 출력 데이터를 발생하기 위해, 수신된 데이터 스트림에 대해 하나 이상에 처리 동작을 수행하는 것을 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 9

제8항에 있어서,

하나 이상의 실행 단계에 대응하는 출력 데이터를 저장하는 단계; 및

프로세스의 실행이 재개된 때에 상기 출력 데이터를 재생하는 단계

를 더 포함하는 컴퓨터에 의해 구현되는 방법.

청구항 10

제1항에 있어서,

상기 프로세스는 프로세스 스테이지의 일부분이며, 데이터 경로를 통해 상기 프로세스 스테이지의 상이한 제2 프로세스와 통신하는, 컴퓨터에 의해 구현되는 방법.

청구항 11

제10항에 있어서,

체크포인트 명령 메시지(checkpoint command message)를 상기 프로세스 스테이지의 각각의 프로세스에 통과시키는 단계; 및

각각의 프로세스에서, 상기 체크포인트 명령 메시지의 수신 시에 초기 상태 또는 최종 상태에 관련된 새로운 정보를 저장하는 단계로서, 프로세스의 작동을 유예시키는 단계와, 상기 새로운 정보를 저장 영역에 저장하는 단계를 포함하는, 상기 저장하는 단계

를 더 포함하는 컴퓨터에 의해 구현되는 방법.

청구항 12

제11항에 있어서,

이전의 저장된 초기 상태 또는 최종 상태를 새로운 초기 상태 또는 최종 상태로 오버라이트하는 단계를 더 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 13

제11항에 있어서,

각각의 프로세스는 프로세스를 위한 데이터를 수신하고 큐잉(queuing)하기 위해 데이터 큐(data queue)와 통신하는, 컴퓨터에 의해 구현되는 방법.

청구항 14

제11항에 있어서,

트리거 이벤트의 검출에 응답하여 상기 체크포인트 명령 메시지를 생성하는 단계를 더 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 15

제14항에 있어서,

상기 트리거 이벤트는 네트워크 이벤트에 대한 정보를 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 16

제11항에 있어서,

상기 체크포인트 명령 메시지를 주기적으로 생성하는 단계를 더 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 17

제11항에 있어서,

처리되고 있는 인입 데이터 레코드 내에 있거나 이 레코드로부터 구해지는 선택된 데이터 값의 발생에 응답하여 상기 체크포인트 명령 메시지를 생성하는 단계를 더 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 18

제11항에 있어서,

상기 사전에 정해진 이벤트에 응답하여 중단 명령 메시지(abort command message)를 생성하는 단계;

상기 중단 명령 메시지를 상기 프로세스 스테이지의 각각의 프로세스에 통과시키는 단계; 및

각각의 프로세스에서 상기 중단 명령 메시지를 수신할 때에, 프로세스의 실행을 중단하고, 상기 중단 명령 메시지를 다음 프로세스에 통과시키는 단계

를 더 포함하는 컴퓨터에 의해 구현되는 방법.

청구항 19

제1항에 있어서,

처리 재개 메시지(resume processing message)에 포함된 정보에 부분적으로 기초하여, 저장된 상기 초기 상태 또는 상기 최종 상태 중의 선택된 상태에서부터 프로세스의 실행을 재개하는 단계를 더 포함하는, 컴퓨터에 의해 구현되는 방법.

청구항 20

제17항에 있어서,

프로세스가 초기화되고, 프로세스의 초기 상태에 관련된 정보가 저장된 직후의 제1 실행 단계 동안 상기 중단 명령 메시지를 수신하는 단계; 및

프로세스를 셧다운하고 재기동할 필요 없이 저장된 상기 초기 상태에서부터 프로세스의 실행을 재개하는 단계

를 더 포함하는 컴퓨터에 의해 구현되는 방법.

청구항 21

명령어를 포함하는 컴퓨터 프로그램을 저장하는 컴퓨터 판독가능 저장 매체로서, 상기 명령어가, 컴퓨팅 시스템으로 하여금,

초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하도록 하며, 여기서 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태를 나타내는 정보를 저장하는 것을 포함하며,

사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하도록 하며,

상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하도록 하는,

컴퓨터 판독가능 저장 매체.

청구항 22

컴퓨팅 시스템에 있어서,

초기화될 시에 프로세스의 초기 상태에 관련된 정보를 수신하고 저장하도록 구성된 입력 장치 또는 입력 포트로서, 여기서 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태를 나타내는 정보를 저장하는 것을 포함하는, 상기 입력 장치 또는 입력 포트; 및

사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하고, 상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하도록 구성된 하나 이상의 프로세서

를 포함하는 컴퓨팅 시스템.

청구항 23

컴퓨팅 시스템에 있어서,

초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하는 수단으로서, 여기서 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태를 나타내는 정보를 저장하는 것을 포함하는, 상기 저장하는 수단; 및

상기 프로세스의 실행을 제어하는 수단으로서, 상기 제어가, 사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하는 것과, 상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하는 것을 포함하는, 상기 제어하는 수단

을 포함하는 컴퓨팅 시스템.

명 세 서

기 술 분 야

[0001] 관련 출원에 대한 상호 참조

[0002] 본 출원은 "Restarting Processes"를 발명의 명칭으로 하여 2011년 2월 18일자로 출원된 미국 특허 출원 번호 13/030,998을 우선권으로 주장하며, 상기 특허 출원은 그 전체 내용이 참조에 의해 본 명세서에 원용된다.

[0003] 본 발명은 프로세스의 재기동에 관한 것이다.

배 경 기 술

[0004] 단일 프로세서 컴퓨터에 의해 제공되는 컴퓨터 처리 속도는 지난 수 십년에 걸쳐 엄청나게 진보되었다. 그러나, 이러한 프로세서에 의해 실행되는 다수의 애플리케이션은 가장 빠른 단일 프로세서 컴퓨터를 넘는 컴퓨터 처리 능력을 요구할 수도 있다. 예컨대, 항공기 예약 시스템과 같은 트랜잭션 시스템에서, 복수의 사용자가 컴퓨터 자원을 동시에 액세스할 수도 있다. 이들 사용자는 통상적으로 짧은 응답 시간을 기대한다. 단일 프로세스 컴퓨터는 이러한 수요를 따르지 못할 수도 있다. 성능을 향상시키기 위해 이러한 애플리케이션을 핸들링하도록 병렬 처리 시스템과 같은 다양한 아키텍처가 개발되었다. 일반적으로, 병렬 처리 시스템은 단일 지점에 위치되거나 또는 원격으로 분포될 수 있는 복수의 프로세서를 이용한다. 이들 프로세서의 처리 능력 때문에, 일부 경우에는 근본적으로 연속적이고 거의 실시간의 처리를 포함할 수 있는 대량의 데이터를 처리하는 애플리케이션에 대해서는 이러한 병렬 처리 시스템에 의존하게 되었다. 이러한 처리 능력은 시스템 장애에 대해 견고하면서 저항성, 즉 내고장성(fault tolerance)을 갖게 되도록 기대한다. 이들 능력은 대규모 인터넷 기반 데이터 처리에서부터 사설 네트워크와 통신 시스템(예컨대, 기업내 "인트라넷" 등)까지의 범위의 모든 유형 및 크기의 컴퓨터 네트워크에 유용하다.

발명의 내용

과제의 해결 수단

[0005] 일특징에서, 전반적으로, 컴퓨터에 의해 구현되는 방법은, 초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하는 단계로서, 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태(end state)를 나타내는 정보를 저장하는 것을 포함하는, 상기 저장하는 단계; 사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하는 단계; 및 상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하는 단계를 포함한다.

[0006] 본 발명의 특징은 이하의 것들 중의 하나 이상을 포함할 수 있다.

[0007] 상기 사전에 정해진 이벤트는 외부 장치에 대한 접속의 상실을 나타낼 수 있다. 상기 사전에 정해진 이벤트는 외부 장치와의 에러를 나타내를 수 있다. 상기 프로세스의 실행은 상기 외부 장치에 대한 접속이 복구된 때에 재개될 수 있다. 상기 프로세스의 실행은 상기 외부 장치와의 에러가 해소된 때에 재개될 수 있다. 상기 프로

세스의 실행은, 상기 사전에 정해진 이벤트가 발생한 실행 단계 이전에 저장된 최종 상태에서부터 재개될 수 있다. 상기 프로세스의 실행은, 실질적으로 상기 프로세스의 기동(startup) 직후의 실행 단계 동안에 상기 사전에 정해진 이벤트가 발생하는 경우에는 상기 초기 상태에서부터 재개될 수 있다. 상기 실행 단계의 실행은, 상기 실행 단계에 대응하는 출력 데이터를 발생하기 위해, 수신된 데이터 스트림에 대해 하나 이상에 처리 동작을 수행하는 것을 포함할 수 있다.

[0008] 상기 컴퓨터에 의해 구현되는 방법은, 하나 이상의 실행 단계에 대응하는 출력 데이터를 저장하는 단계와, 프로세스의 실행이 재개된 때에 상기 출력 데이터를 재생하는 단계를 더 포함할 수 있다. 상기 프로세스는 프로세스 스테이지의 일부분이며, 데이터 경로를 통해 상기 프로세스 스테이지의 상이한 제2 프로세스와 통신할 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 체크포인트 명령 메시지(checkpoint command message)를 상기 프로세스 스테이지의 각각의 프로세스에 통과시키는 단계와; 각각의 프로세스에서, 상기 체크포인트 명령 메시지의 수신 시에 초기 상태 또는 최종 상태에 관련된 새로운 정보를 저장하는 단계로서, 프로세스의 작동을 유예시키는 단계와, 상기 새로운 정보를 저장 영역에 저장하는 단계를 포함하는, 상기 저장하는 단계를 더 포함할 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 이전의 저장된 초기 상태 또는 최종 상태를 새로운 초기 상태 또는 최종 상태로 오버라이트하는 단계를 더 포함할 수 있다.

[0009] 각각의 프로세스는 프로세스를 위한 데이터를 수신하고 큐잉(queuing)하기 위해 데이터 큐(data queue)와 통신하게 될 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 트리거 이벤트의 검출에 응답하여 상기 체크포인트 명령 메시지를 생성하는 단계를 더 포함할 수 있다. 상기 트리거 이벤트는 네트워크 이벤트에 대한 정보를 포함할 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 상기 체크포인트 명령 메시지를 주기적으로 생성하는 단계를 더 포함할 수 있다.

[0010] 상기 컴퓨터에 의해 구현되는 방법은, 처리되고 있는 인입 데이터 레코드 내에 있거나 이 레코드로부터 구해지는 선택된 데이터 값의 발생에 응답하여 상기 체크포인트 명령 메시지를 생성하는 단계를 더 포함할 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 상기 사전에 정해진 이벤트에 응답하여 중단 명령 메시지(abort command message)를 생성하는 단계와; 상기 중단 명령 메시지를 상기 프로세스 스테이지의 각각의 프로세스에 통과시키는 단계와; 각각의 프로세스에서 상기 중단 명령 메시지를 수신할 때에, 프로세스의 실행을 중단하고, 상기 중단 명령 메시지를 다음 프로세스에 통과시키는 단계를 더 포함할 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 처리 재개 메시지(resume processing message)에 포함된 정보에 부분적으로 기초하여, 저장된 상기 초기 상태 또는 상기 최종 상태 중의 선택된 상태에서부터 프로세스의 실행을 재개하는 단계를 더 포함할 수 있다. 상기 컴퓨터에 의해 구현되는 방법은, 프로세스가 초기화되고, 프로세스의 초기 상태에 관련된 정보가 저장된 직후의, 제1 실행 단계 동안 상기 중단 명령 메시지를 수신하는 단계와; 프로세스를 셧다운하고 재기동할 필요 없이 저장된 상기 초기 상태에서부터 프로세스의 실행을 재개하는 단계를 더 포함할 수 있다.

[0011] 또 다른 특징에서, 일반적으로, 명령어를 포함하는 컴퓨터 프로그램을 저장하는 컴퓨터 판독가능 저장 매체는, 상기 명령어가, 컴퓨팅 시스템으로 하여금, 초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하도록 하며, 여기서 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태를 나타내는 정보를 저장하는 것을 포함하며; 사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하도록 하며; 상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하도록 한다.

[0012] 또 다른 특징에서, 일반적으로, 컴퓨팅 시스템은, 초기화될 시에 프로세스의 초기 상태에 관련된 정보를 수신하고 저장하도록 구성된 입력 장치 또는 입력 포트로서, 여기서 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태를 나타내는 정보를 저장하는 것을 포함하는, 상기 입력 장치 또는 입력 포트와; 사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하고, 상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하도록 구성된 하나 이상의 프로세서를 포함한다.

[0013] 또 다른 특징에서, 컴퓨팅 시스템은, 초기화될 시에 프로세스의 초기 상태에 관련된 정보를 저장하는 수단으로서, 여기서 상기 프로세스의 실행이, 하나 이상의 실행 단계를 실행하는 것과, 상기 실행 단계의 실행의 완료 시에 상기 실행 단계의 최종 상태를 나타내는 정보를 저장하는 것을 포함하는, 상기 저장하는 수단과; 상기 프로세스의 실행을 제어하는 수단으로서, 상기 제어가, 사전에 정해진 이벤트에 응답하여 상기 프로세스의 실행을 중단하는 것과, 상기 프로세스를 셧다운할 필요 없이 저장된 상기 초기 상태와 상기 최종 상태 중의 하나에서부터 상기 프로세스의 실행을 재개하는 것을 포함하는, 상기 제어하는 수단을 포함한다.

발명의 효과

[0014] 본 발명의 특징은 이하의 장점 중의 하나 이상을 포함할 수 있다.

[0015] 멀티-프로세스 처리 시스템의 프로세스는 별개의 실행 단계로 실행될 수 있다. 시스템 장애의 이벤트에서, 처리 시스템을 중단시키고 가장 최근에 완료된 체크포인트에서부터의 재기동은 과도한 양의 처리 시간 및 자원을 소모할 수 있다. 처리 시스템이 예외 조건(exception condition)에 응답하여 자신의 활동(activity)을 종료한 후, 처리 시스템은 이러한 시스템에 경험이 있는 정보 기술 전문가에 의해 수동으로 다시 초기화될 필요가 있을 수도 있다. 이것은 커다란 시스템 비작동 시간을 초래할 수 있다. 일부 예에서, 별도의 프로세스가 시스템 장애를 검지하고 전문가에게 알려주도록 설계될 필요가 있을 수도 있다. 이로써, 효율을 향상시키고 처리 자원 소모를 감소시키기 위해, 처리 시스템 내의 프로세스는, 프로세스에 대한 장애 발생 접속(failed connection to a process)이 복구될 시에, 전체 시스템을 재기동하는 대신 자신의 마지막 기록된 체크포인트에서부터 실행될 수 있다. 일구현예에서는, 전체 시스템을 종료하고 재기동하지 않고, 시스템의 개별 프로세스가 장애 발생 접속이 복구될 때까지 처리를 유예하도록 통지될 수 있다.

[0016] 본 발명의 기타 특징 및 장점은 이하의 상세한 설명 및 청구범위로부터 명백하게 될 것이다.

도면의 간단한 설명

[0017] 도 1은 멀티-프로세스 데이터 처리 시스템의 블록도이다.

도 2 및 도 3은 일례의 멀티-프로세스 데이터 처리 시스템을 예시하고 있다.

도 4는 일례의 체크포인팅 프로세스(checkpointing process)를 예시하는 흐름도이다.

도 5 및 도 6은 일례의 복구 메카니즘의 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0018] 도 1을 참조하면, 데이터 처리 시스템(100)은 데이터를 처리하기 위해 능률적인 방식(streamlined manner)으로 배열되는 복수의 프로세스를 제공한다. 일례의 시스템(100) 내에서, 데이터는 데이터 소스(102)(예컨대, 웹 서버로서 기능하는 서버(104) 상에서 실행되는 애플리케이션)로부터 수신되고, 컴퓨터 시스템(108) 상에서 실행되거나 또는 분산 방식으로(예컨대, 2개 이상의 네트워크 연결된 컴퓨터 단말기를 이용하여) 실행되는 멀티-프로세스 데이터 처리 모듈(106)에 통신된다. 데이터 처리 모듈(106)은 데이터 처리 시스템(100)의 데이터 처리 특징을 모니터하고, 제어하고, 수행한다. 이러한 처리를 제공하기 위해, 데이터 처리 모듈(106)은 하나 이상의 프로세스(116, 118)에 의해 처리될 데이터를 저장할 수 있는 하나 이상의 큐(queue)(110, 112, 114)를 포함한다. 이 경우, 도시된 바와 같이, 데이터 소스(102)로부터 수신된 데이터는 초기 데이터 큐(110)에 저장되고, 초기 프로세스(116)에 추기적으로 제공된다. 초기 프로세스(116)는 데이터를 처리(예컨대, 컨텐츠를 변환하고, 필터링하고, 확인하는 등)하고, 처리된 데이터를 하나 이상의 하류측 데이터 큐(112, 112')에 제공한다. 후속 프로세스(118, 118')는 큐(112, 112')로부터의 데이터가 제공될 수 있으며, 기타(또는 유사한) 처리를 수행한 후에 그 결과를 다른 하류측 데이터 큐(114, 114')에 전달한다. 데이터 처리 모듈(106)의 예시된 큐 및 프로세스 레이아웃은 이용될 수 있는 다수의 가능한 처리 체계 중의 하나이다. 예컨대, 데이터 처리 모듈(106)은 도시된 프로세스의 상류측에 위치되거나, 하류측에 위치되거나, 또는 도시된 프로세스에 대해 독립적으로 위치될 수 있는 추가의 프로세스(예컨대, 병렬 또는 직렬 실행을 위한)를 포함할 수 있다. 일부 예에서, 마지막 세트의 큐(예컨대, 큐 114 및 114')로부터의 데이터는 관계형 데이터베이스 관리 시스템(RDBMS)과 같은 수신지 애플리케이션(120)(또는 복수의 애플리케이션)에 출력될 수 있다.

[0019] 데이터 처리 모듈(106)에 포함된 프로세스는 외부 장치 및/또는 다른 처리 시스템(예컨대, 컴퓨터 시스템(122))과 통신할 수도 있다. 예컨대, 프로세스는 메시지를 다른 시스템으로부터의 프로세스에 제공하는 자바 메시지 서비스(Java Message Service, JMS) 큐과 통신하게 될 수도 있다. 일부 경우에, 데이터 처리 모듈(106) 내의 프로세스는 하나 이상의 데이터베이스(예컨대, 외부 시스템(122)에 위치된)와 통신하게 될 수도 있다. 예컨대, 데이터 처리 모듈(106)은 하나 이상의 자동화 기기(ATM)에서 해당 고객 세션(customer session)으로부터 수신된 정보에 기초하여 은행 데이터베이스 내의 그 고객의 금융 계좌에 대한 업데이트를 수행할 수 있다.

[0020] 예로써, 도 2는 중앙 위치에서 처리하기 위한 데이터를 제공하기 위해 이용되는 원격으로 실행되는 처리 모듈(202)(ATM(204)에 의해 실행되는)을 갖는 처리 시스템(200)을 예시하고 있다. 예시된 예에서, 초기 프로세스, 예컨대 ATM 판독 프로세스(206)는 ATM으로부터 고객 계좌 데이터(예컨대, 트랜잭션에 연관된)를 수신할 수

있고, 그 데이터를 계좌 세부내용을 인증하기 위한 계좌 검증 프로세스(208)에 보낼 수 있다. 이 경우, 계좌 검증 프로세스(208)는 고객에 의해 입력된 개인 식별 번호(PIN)를 PIN 데이터베이스(210)에 대하여 확인할 수 있다. 고객의 신원이 인증된 후, 추가의 데이터 레코드가 하류측으로 잔고 확인 프로세스(212)에 통신될 수 있으며, 잔고 확인 프로세스가 예컨대 식별된 고객 계좌의 잔고를 확인하기 위한 상이한 제2 데이터베이스(214)와 통신할 수 있다. 그 이상의 트랜잭션이 완료된 후, 추가의 데이터가 하류측으로 잔고 업데이트 프로세스(216)에 보내질 수 있으며, 잔고 업데이트 프로세스가 예컨대 고객 계좌에 연관된 잔고 정보를 업데이트하기 위한 제3 데이터베이스(218)와 통신할 수 있다. 응답 생성 프로세스(220)가 트랜잭션의 개요 출력(output summary)을 작성할 수 있으며, 이 개요 출력이 출력 디스플레이 프로세스(222)(예컨대, ATM(204)을 통해 고객에게 디스플레이하기 위한)에 제공될 수 있다. 시스템-레벨 모니터링(예컨대, 시스템 품질 보증) 또는 다른 애플리케이션을 위해, 데이터베이스(210, 214, 218)는 마스터 데이터 서버(224)와 통신하게 될 수 있다. 일부 구현예에서, 데이터베이스는 예컨대 독립형 컴퓨터 시스템(226)에 의해 실행될 수도 있다.

[0021] 이러한 멀티-프로세스 처리 시스템의 프로세스는 별개의 실행 단계로 실행될 수도 있다. 프로세스의 실행은 프로세스 내의 하나 이상의 태스크의 실행을 별개의 실행 단계로 포함할 수 있다. 실행을 이러한 단계로 단편화함으로써, 별개의 실행 단계들은 예컨대 데이터 처리에서의 복수의 논리적 종점 또는 중단점(multiple logical endpoints or breakpoint)에 의해 종료될 수도 있다. 각각의 실행 단계는 그 실행 단계의 목적을 달성하기 위해 하나 이상의 처리 동작을 가질 수 있다. 일례로서, 계좌 검증 프로세스(208)는 별개의 실행 단계에서 하나 이상의 방식으로 실행될 수 있다. 예컨대, 제1 실행 단계로서, 계좌 검증 프로세스(208)는 처음에 고객으로부터 개인 식별 번호(PIN)를 수신할 수 있다. 고객으로부터 PIN 정보를 수신함에 있어서의 다양한 처리 동작은, 예컨대, ATM 디스플레이 상에 프롬프트를 디스플레이하는 것과, PIN 정보의 데이터 입력을 확인하기 위해 루틴을 실행하는 것을 포함할 수 있다. 다음 실행 단계에서, 계좌 검증 프로세스(208)는 데이터베이스(210)에 대한 접속을 구축하고, PIN 정보를 키(key)로 이용하여 고객의 레코드를 식별한다. 계좌 검증 프로세스(208)가 고객의 레코드와의 자신의 트랜잭션을 완료한 후, 데이터베이스(210)에 대한 접속이 종료될 수 있다. 최종 실행 단계에서, 계좌 검증 프로세스(208)는 앞에서 이루어진 트랜잭션에 기초하여 결과를 생성할 수 있다. 이와 같이, 각각의 이들 실행 단계는, 계좌 검증 프로세스(208)가 일시적으로 유예(suspend)되거나 및/또는 재개될 수 있는 별개의 논리적 종점(또는 처리 중단점)을 포함한다.

[0022] 몇몇 상황에서, 정상적인 시스템 작동의 과정에 영향을 주기 쉬운 하나 이상의 이벤트가 발생할 수도 있다. 이러한 이벤트는 처리 시스템에 포함된 하드웨어 또는 소프트웨어 모듈 중의 하나에 의해 발생되는 예외 또는 에러일 것이다. 예컨대, 하드웨어 예외 또는 에러는 하나 이상의 하드웨어 유닛으로부터의 리셋, 인터럽트 또는 기타 신호를 포함할 수 있다. 예외는 또한 영(0)으로의 나눔과 같은 수치 에러, 오버플로우, 명령어 디코딩 에러, 정의되지 않은 명령어 등에 대하여 산술 논리 유닛에 의해 발생될 수도 있다. 하나 이상의 이러한 이벤트의 발생에 따라, 정정 동작(예컨대, 유지보수, 2차 시스템으로의 스위칭 오버 등)을 취하기 위해 하나 이상의 데이터베이스의 작동을 일시적으로 정지(halt)시키는 것이 필요할 수도 있다.

[0023] 작동의 정지 및 정정 동작을 요청할 수도 있는 기타 이벤트는 데이터베이스(210~224) 중의 하나 이상의 데이터베이스의 장애의 검출을 포함할 수 있다. 이러한 장애는 다양한 이유로 발생할 수 있다. 예컨대, 메모리 할당에서의 에러 또는 메모리 공간에의 기입 시의 충돌이 있을 수 있다. 또한, 프로세스가 잔고가 없는 계좌로부터 자금을 인출하려고 시도할 때와 같은 기초 데이터 조작(underlying data operation)에서의 에러가 있을 수도 있다. 일시적인 장애가 있는 곳에서의 이벤트 외에, 조작자 개입에 의해 트리거되는 이벤트가 있을 수도 있다. 일부 구현예에서는, 조작자가 장애를 유발한 상황을 바로잡을 수도 있고, 또는 시스템이 그 상황을 제때에 바로잡을 수도 있다. 이벤트의 예는 네트워크에 접속된 하나 이상의 장치의 장애, 유지보수를 위한 하나 이상의 장치 또는 소프트웨어 서비스의 셧다운, 장치 또는 소프트웨어 서비스의 장애 및 스위치 오버, 저장 공간과 같은 자원의 부족, 하나 이상의 소프트웨어 서비스의 타임아웃을 포함할 수 있으며, 이벤트의 예는 위에 언급한 것으로 한정되지는 않는다.

[0024] 이러한 이벤트를 검지하고 해소하기 위해, 데이터 처리 시스템은 장애의 이벤트 시에 또는 시스템이 유지보수 또는 스위치-오버를 위해 오프라인으로 되는 때에 시스템 비작동 시간을 최소로 하기 위해 체크포인팅 기술로도 지칭되는 하나 이상의 기술을 이용할 수 있다. 체크포인팅 기술은 일반적으로 프로세스의 현재 상태의 세부내용을 체크포인트 레코드로서 저장하여 프로세스가 저장된 정보를 이용하여 그 상태로부터 추후에 재기동될 수 있도록 하는 것을 포함한다. 예컨대, 계좌 검증 프로세스(204)는 각각의 실행 단계의 완료 시에 자신의 현재 상태를 체크포인트 레코드에 저장할 수 있다(다음 실행 단계의 실행 또는 다른 처리를 개시하기 전에).

[0025] 체크포인트 레코드는 프로세스 값, 성공적으로 처리된 레코드에 대한 정보, 및 프로세스의 현재 실행 단계에 관

련된 기타 세부내용과 같은 다양한 타입의 정보를 포함할 수 있다. 예컨대, 체크포인트 레코드는 처리되고 있는 데이터를 보낸 데이터 큐(예컨대, 도 1의 데이터 큐 112)에서의 현재 위치에 대한 정보를 포함할 수 있다. 이로써, 정지 조작 후에, 이 큐 위치에서부터 처리가 재개될 수 있다. 이들 라인을 따라, 시스템 장애로부터의 복구 후에, 프로세스는 초기 상태에서부터 재기동하지 않고 저장된 중간 체크포인트 상태에서부터 재기동할 수 있다.

[0026] 일례로서, PIN 데이터베이스(210)가 장애가 발생하면, 계좌 검증 프로세스(208)는 전체 처리 시스템을 종료시키기 위해 예외를 발생할 수 있다. 재기동 시에, 처리 시스템의 프로세스(또는 프로세스의 일부분)는 자신의 마지막 체크포인트 상태에서부터 처리를 지속할 수 있다. 이 예에서, 장애 및 재기동이 고객이 자신의 PIN 정보를 제공한 후의 시점에서 발생하기 때문에, PIN 정보가 프로세스에 복원되고, 고객으로부터 재수집될 필요가 없다. 이로써, 고객에게 자신의 PIN 정보를 제공하도록 다시 촉구할 필요가 없을 것이다.

[0027] 시스템 장애의 이벤트에서, 처리 시스템의 종료 및 가장 최근에 완료된 체크포인트에서부터의 재기동은 과도한 양의 처리 시간 및 자원을 소모할 수 있다. 처리 시스템이 예외 조건에 응답하여 자신의 활동(activity)을 종료한 후, 처리 시스템은 이러한 시스템에 경험이 있는 정보 기술 전문가에 의해 수동으로 다시 초기화될 필요가 있을 수도 있다. 이것은 커다란 시스템 비작동 시간을 초래할 수 있다. 일부 예에서, 별도의 프로세스가 시스템 장애를 검지하고 전문가에게 알려주도록 설계될 필요가 있을 수도 있다. 체크포인팅 시스템의 예는 "Continuous Flow Checkpointing Data Processing"을 발명의 명칭으로 하는 미국 특허 제6,584,581호, "Overpartitioning system and method for increasing checkpoints in component-based parallel applications"을 발명의 명칭으로 하는 미국 특허 제5,819,021호, 및 "Methods and Systems for Reconstructing the State of a Computation"을 발명의 명칭으로 하는 미국 특허 제5,712,971호에 개시되어 있으며, 이들 특허의 각각의 내용이 참조에 의해 본 명세에 전체적으로 원용된다.

[0028] 효율을 향상시키고 처리 자원 소모를 감소시키기 위해, 처리 시스템 내의 프로세스는, 프로세스에 대한 장애 발생 접속(failed connection to a process)이 복구될 시에, 전체 시스템을 재기동하는 대신 자신의 마지막 기록된 체크포인트에서부터 실행된다. 일구현예에서는, 전체 시스템을 종료하고 재기동하지 않고, 시스템의 개별 프로세스가 장애 발생 접속이 복구될 때까지 처리를 유예하도록 통지될 수 있다.

[0029] 도 3은 데이터 소스 프로세스(302), 프로세스(304a~304n), 데이터 싱크 프로세스(data sink process)(306), 및 각각의 다른 프로세스(프로세스 302, 304a~304n)와 통신하는 내고장성 관리부(fault-tolerance manager)(308)를 포함하는 멀티-프로세스 시스템(300)의 블록도를 도시하고 있다. 일부 구현예에서, 내고장성 관리부(308)는 멀티-프로세스 시스템(300) 내의 또 다른 프로세스로서 실행될 수도 있다. 일부 상황에서, 내고장성 관리부(308)는 별도의 컴퓨터 시스템(도시하지 않음)에서 작동하는 애플리케이션이거나, 또는 예컨대 "Continuous Flow Checkpointing Data Processing"을 발명의 명칭으로 하는 미국 특허 제6,584,581호에 개시된 체크포인트 프로세서와 같은 전용 프로세서로 구현될 수도 있으며, 이 특허의 내용이 참조에 의해 본 명세서에 전체적으로 원용된다.

[0030] 프로세스(302~306)와 내고장성 관리부(308) 간의 통신을 구축하기 위해 하나 이상의 기술이 실시될 수 있다. 예컨대, 프로세스(302~306)에서 발생할 수 있는 예외 조건에 대한 정보를 통신하기 위해 개별적인 예외 채널(310a~310n)이 이용될 수 있다. 예외 채널(310a~310n)은 유선, 무선, 또는 유선과 무선이 조합된 네트워크 시스템의 일부분이어도 된다. 예외 채널(310a~310n)은 프로세스(302~306)에 대한 에러 정보를 내고장성 관리부(308)에 통신하기 위해 프로세스(302~306)에 의해 이용될 수 있다. 예컨대, 프로세스 304a와 통신하는 외부 장치가 장애가 발생하면, 프로세스 304a는 즉각적으로 에러 플래그를 발생하고, 이 에러를 예외 채널 310b를 통해 내고장성 관리부(308)에 통신할 수 있다.

[0031] 예외 채널(310a~310n)에 추가하여, 내고장성 관리부(308)는 명령 메시지(예컨대, 체크포인트 명령 메시지)를 대응하는 통신 채널(312a~312e)을 통해 프로세스(302~306)에 전송할 수 있다. 통신 채널(312a~312e)은 명령 메시지를 내고장성 관리부(308)로부터 순차적으로 각각의 프로세스(302~306)에 전송하도록 배치된다. 예컨대, 내고장성 관리부(308)로부터의 메시지가 먼저 데이터 소스 프로세스(302)에 전송되고, 그리고나서 통신 채널(312b~312d)을 통해 각각의 프로세스(304a~304n) 및 데이터 싱크 프로세스(306)에 직렬로 통과된다. 데이터 싱크 프로세스(306)는 명령 메시지를 내고장성 관리부(308)에 통신하기 위해 채널 312e를 이용할 수 있다.

[0032] 각각의 프로세스(302, 304a~304n, 306)에 연관된 각각의 체크포인트 데이터를 저장하기 위해, 저장 영역(예컨대, 메모리)이 각각의 프로세스에 할당될 수 있다. 각각의 프로세스는 별도의 실행 단계의 끝에서 자신의 현재 작동을 주기적으로 유예하고, 자신의 체크포인트 데이터를 연관된 저장 영역에 저장한다. 예컨대, 데이터 소스

프로세스(302)는 처리 중에 있는 별개의 실행 단계의 끝(인입 데이터의 스트림에서의 논리적 중단점과 같은)에서 자신의 현재 작동을 주기적으로 유예하고, 체크포인트 정보를 저장 영역(312)에 저장한다. 이러한 방식으로, 프로세스(302, 304a~304n, 306)의 각각이 실행될 때, 대응하는 저장 영역(312, 314a~314n, 316)이 체크포인트 데이터를 주기적으로 저장한다. 저장 영역(312~316)은 예컨대 하드 디스크 드라이브와 같은 자기 매체 등의 비휘발성 저장 장치와 같은 다양한 타입의 저장 기술로 구현될 수 있다. 일부 예에서, 2개 이상의 프로세스가 단일 저장 영역을 공유할 수도 있다. 체크포인트 데이터는 프로세스(302~306)에 연관된 데이터 및/또는 현재 상태에 대한 정보를 포함하여 추후에 이들 상태의 재구축을 가능하게 할 수 있다.

[0033] 일부 예에서, 데이터 저장 영역(312~316)은 또한 프로세스에 연관된 데이터 큐(예컨대, 도 1의 큐(110, 112, 114))를 저장할 수 있다. 예컨대, 저장 영역(316)은 프로세스 304n로부터 처리된 데이터를 수신하는 큐를 포함할 수 있으며, 이 큐로부터의 데이터가 출력되거나 공개(예컨대, 인쇄 또는 디스플레이)될 수 있다.

[0034] 내고장성 관리부(308)는, 체크포인트 명령 메시지를 생성하고, 이 메시지를 통신 채널(312a~312e)을 통해 각각의 프로세스(302~306)에 순차적으로 통과시킴으로써, 체크포인팅 작동을 관리한다. 체크포인트 명령 메시지는 각각의 프로세스(302~306)를 통과하게 되어, 프로세스가 메시지의 수신 시에 자신의 현재 상태를 체크포인트할 수 있게 된다. 이와 같이, 체크포인트 명령 메시지는 데이터 소스 프로세스(302)로 이동하게 되고, 그리고나서 내고장성 관리부(308)에 리턴되기 전에 각각의 프로세스(304a~304n) 및 데이터 싱크 프로세스(306)를 순차적으로 통과하게 된다. 체크포인팅 작동은 규칙적인 간격으로 자동으로 개시될 수 있다. 예컨대, 내고장성 관리부(308)는 예컨대 매 5분 마다와 같은 사전에 정해진 주기적인 비율로 체크포인트 메시지를 개시할 수 있다. 주기적 비율은 디폴트 값으로 설정될 수도 있고, 또는 사용자에 의해 조정될 수도 있다.

[0035] 시스템에 포함된 프로세스에 대한 체크포인트를 저장하기 위한 작동을 개시하기 위해 하나 이상의 기술이 실시될 수 있다. 예컨대, 하나 이상의 외부 트리거가 체크포인트 정보를 저장하기 위한 작동을 개시할 수 있다. 일례에서, 네트워크 메시지는 내고장성 관리부(308)에게 임박한 네트워크 셧다운을 알려주고, 그러므로 체크포인팅 작동을 트리거할 수 있다. 일부 구현예에서, 체크포인팅 작동은 처리되고 있는 데이터 레코드 내의 값 또는 처리되고 있는 데이터 레코드로부터 구해지는 값에 응답하여 트리거될 수 있다. 예컨대, 처리된 데이터 레코드는 체크포인팅이 발생할 수 있는 논리적인 포인트로서 간주될 수 있는 타임스탬프(timestamp) 또는 중단점 값(breakpoint value)을 포함할 수 있다.

[0036] 데이터가 시스템에 의해 처리되고 있는 기간 동안 체크포인트 정보를 저장하는 것과 함께, 데이터를 처리하기 전에 정보가 저장될 수 있다. 일구현예에서, 초기 체크포인팅 작동은 예컨대 기동 동안과 같은 멀티-프로세스 시스템(300)이 최초로 초기화될 시에 트리거될 수 있다. 내고장성 관리부(308)는 초기 체크포인트 명령 메시지를 각각의 프로세스(302~306)에 통과시킬 수 있다. 도 3에 도시된 예에서, 초기 체크포인트 메시지는 먼저 데이터 소스 프로세스(302)에 통신된다. 데이터 소스 프로세스(302)는 즉각적으로 체크포인트한다. 즉, 데이터 소스 프로세스(302)는 예컨대 자신의 초기 상태를 나타내는 데이터를 연관된 데이터 저장 공간(312)에 저장하고, 초기 체크포인트 메시지를 하류측으로 다음 프로세스(304a)에 통과시킨다. 이 초기 체크포인트 상태는 "체크포인트 상태 0"로 지칭된다.

[0037] 유사하게, 직렬 방식으로, 각각의 프로세스(304~306)는 이에 대응하여 자신의 초기 상태 및 연관된 데이터 값을 적합한 저장 영역에 체크포인트 상태 0로서 저장할 수 있다. 몇몇 예에서, 초기 상태 및 연관된 데이터 값은 전역 변수(global variable), 참조 데이터 정보, 및 카운터의 초기값을 포함한 감사 변수(auditing variable)의 초기값을 포함할 수 있다.

[0038] 각각의 프로세스(302~306)가 자신의 초기 상태를 저장한 후, 초기 체크포인트 명령 메시지가 채널 312e를 통해 내고장성 관리부(308)에게 리턴된다. 프로세스(302, 304a~304n, 306)를 통한 왕복 주행(round trip) 후에 내고장성 관리부(308)에게 리턴되는 메시지에 기초하여, 내고장성 관리부는 프로세스(302~306)가 체크포인트 상태 0를 완료한 것을 알게 된다. 일부 구현예에서는, 하류측 프로세스가 자신의 현재 상태를 저장하는 동안, 소스 및 기타 상류측 프로세스는, 모든 프로세스가 자신의 상태를 저장하는 것을 대기하지 않고, 데이터를 수신하고 기타 기능을 수행하는 것을 지속할 수 있다.

[0039] 유사하게, 프로세스(302~306)의 각각의 별개의 실행 단계에 대해 추가의 체크포인팅이 수행될 수 있다. 이로써, 초기 체크포인트 정보를 나타내는 데이터를 저장하는 것과 함께, 내고장성 관리부(308)는 예컨대 후속의 체크포인트 사이클(예컨대, 체크포인트 상태 1, 2, 3,... 등)에 연관된 정보를 나타내는 추가의 정보의 저장을 개시할 수 있다. 후속 체크포인트 정보의 저장을 개시하기 위해, 프로세스(302, 304a~304n, 306)를 통해 추가의 체크포인트 명령 메시지를 전파하는 것과 같은 기술이 이용될 수 있다. 체크포인트 명령 메시지의 수신 시에,

프로세스 304a는 임의의 진행중인 태스크를 완료하거나 또는 임의의 미해결된 태스크(outstanding task)를 유예할 수 있다. 일부 예에서, 프로세스 304a는 데이터 저장 장치(314)에 저장된 이전에 생성된 체크포인트 레코드를 삭제하고, 저장 공간을 회복시킬 수 있다. 프로세스 304는 그 후 자신의 현재 상태 및 연관된 데이터에 대해 새로운 체크포인트 레코드를 생성할 수 있다. 일부 시나리오에서는, 더 이전의 체크포인트 레코드가 메모리에 지속적으로 저장되고, 새로운 체크포인트 레코드에 의해 오버라이트되지 않는다. 체크포인팅 레코드에 저장된 정보의 추가의 예가 미국 특허 제6,584,581호에 제공되어 있으며, 이 특허의 내용이 본 명세서에 전체적으로 원용된다.

[0040] 일부 구현예에서, 프로세스 304a는 외부 데이터베이스(318)(컴퓨터(320) 상에서 실행되는)와 추가로 통신하게 될 수도 있다. 간혹, 데이터베이스(318)에 대한 접속에 장애가 발생할 수도 있거나, 또는 데이터베이스(318)가 유지보수를 위해 오프라인으로 될 수도 있다. 장애는 데이터베이스(318)를 실행하는 컴퓨터 시스템(320)의 하드웨어 장애일 수 있다. 이러한 상황에서, 프로세스 304a는 내고장성 관리부(308)에게 접속의 상실을 통지하기 위해 예외 채널(310a)을 통해 에러 플래그를 발생할 수 있다.

[0041] 에러의 통보를 수신할 시에, 내고장성 관리부(308)는 중단 명령 메시지(abort command message)를 생성하여 이 메시지를 프로세스(302~306)를 통해 전파할 수 있다. 중단 명령 메시지는 통상적으로 채널 312a에 의해 데이터 소스 프로세스(302)에 먼저 통신되고, 그리고나서 채널 312b~312d에 의해 각각의 프로세스(302~306)를 통해 통신되고, 최종적으로 채널 312e에 의해 다시 내고장성 관리부(308)에게 통신된다. 중단 명령 메시지를 수신할 시에, 각각의 프로세스(302~306)는 자신의 현재 활동을 비교적 작은 지연(있는 경우에)으로 중단하고, 마지막 체크포인트 상태 이후에 처리되었을 수도 있는 임의의 미해결된 태스크 또는 레코드를 버리거나(flush) 폐기한다. 프로세스가 활동을 중단한 후, 그 프로세스는 중단 명령 메시지를 다음의 하류측 프로세스에 통과시킬 수 있다. 이러한 방식으로, 중단 명령 메시지는 내고장성 관리부(308)에게 리턴되기 전에 싱크 프로세스(306)에 전파된다. 내고장성 관리부(308)는 싱크 프로세스(306)로부터 중단 명령 메시지를 수신할 때까지 대기하여, 프로세스(302~306)의 전부가 현재 처리하고 있는 태스크를 중단하도록 한다(예컨대, 진행중단 상태(quiescent state)로 한다).

[0042] 데이터베이스(318)가 컴퓨터 시스템(320)에서의 하드웨어 장애로 인해 장애가 발생한 시나리오에서, 프로세스(302~306)는 자신의 처리를 중단하도록 지시를 받게 된다. 일부 구현예에서, 시스템이 자신의 처리를 완전히 중단한 후, 프로세스 302는 장애를 바로잡는데 소요되는 평균 시간량을 반영하여야 하는 특정한 양의 시간을 대기하고, 마지막 저장된 체크포인트 상태에서부터 다시 처리를 개시할 수 있다. 일부 구현예에서, 프로세스 304a는 자신의 상황을 체크(즉, 데이터베이스(312)가 작동하는지를 체크)하기 위해 데이터베이스(318)를 주기적으로 폴링(polling)할 수 있다. 일부 예에서, 컴퓨터 시스템(320)은 데이터베이스(318)가 작동 상태로 복구되는 때에 프로세스 304a에 자동으로 통지하도록 구성될 수 있다. 데이터베이스(318)와의 접속이 복구될 때, 처리 시스템(300)은 마지막 저장된 체크포인트 상태에서부터 다시 처리를 개시할 수 있다.

[0043] 이에 관하여, 프로세스 304a는 접속이 복구되었다는 것을 내고장성 관리부(308)에 통보한다. 내고장성 관리부(308)는 각각의 프로세스(302~306)에 대해 마지막의 성공적으로 완료된 체크포인트 상태를 결정하고, 각각의 프로세스(302~306)에 처리 재개 메시지(resume processing message)를 전송한다. 다른 명령 메시지와 마찬가지로, 처리 재개 메시지는 통신 채널(312a~312e)을 통해 각각의 프로세스(302~306)에 순차적으로 전파된다. 처리 재개 메시지는 체크포인트 상태를 특정하고, 프로세스(302~306)가 이 체크포인트 상태에서부터 처리를 재개할 것이다.

[0044] 일부 경우에, 내고장성 관리부(308)는 이전의 체크포인팅 작동이 현재 실행되고 있어도 추가의 체크포인팅 작동을 개시할 수 있다. 예컨대, 프로세스 304n이 임의의 체크포인트 상태(예컨대, 체크포인트 명령 메시지 N에 대응하는 체크포인트 상태 N)를 처리하고 있어도, 내고장성 관리부(308)는 후속 체크포인트 명령 메시지 N+1을 생성하여 소스 프로세스 302에 전송함으로써 후속 체크포인트 상태(예컨대, 체크포인트 상태 N+1)를 시작할 수 있다. 이들 라인을 따라, 체크포인트 명령 메시지 N이 여전히 프로세스(302~306)를 통해 이동하고 있을 때에, 새로운 후속 체크포인트 명령 메시지 N+1이 생성되어 프로세스(302~306)에 통과되는 것이 가능하다. 이러한 방식으로, 내고장성 매니저(308)는 이전의 체크포인팅 상태가 완료될 때까지 대기할 필요 없이 프로세스 상태의 더욱 빈번한 체크포인팅을 야기할 수 있다.

[0045] 일부 상황에서, 하나 이상의 체크포인트 명령 메시지가 프로세스(302, 304a~304n, 306)에 전달 중인 동안 시스템 장애가 발생할 수도 있다. 예컨대, 내고장성 관리부(308)가 체크포인트 명령 메시지 N을 생성함으로써 체크포인트 상태 N을 개시한 시나리오를 고려하면, 체크포인트 명령 메시지 N이 프로세스(302~306)에 의해 처리되

는 동안, 프로세스들 중의 하나(예컨대, 프로세스 304a)와 외부 시스템(예컨대, 데이터베이스 312) 간의 접속에 장애가 발생할 수도 있다. 이 상황에 대해 경고가 발생될 때에, 내고장성 관리부(308)는 중단 명령 메시지를 프로세스(302~306)에 통과시킴으로써 응답할 수 있다. 중단 명령 메시지는 여전히 체크포인트 상태 N을 처리하는(예컨대, 체크포인트 N에 연관된 체크포인트 정보를 저장하는) 프로세스(예컨대, 프로세스 304n)에 도달할 수 있다. 중단 명령의 수신에 따라, 프로세스 304n은 하나 이상의 동작을 취할 수 있다. 예컨대, 프로세스 304n는 체크포인트 상태 N을 완료하고, 더 이상의 처리 전부를 중단할 수 있다. 또 다른 시나리오에서, 프로세스 304n은 이전의 체크포인트 상태 N-1 이후의 현재 및 후속 상태에 연관된 결과를 즉각적으로 폐기하고, 더 이상의 처리를 중단할 수 있다. 그 결과, 시스템(300)이 진행중단 상태를 달성한 때에, 각각의 프로세스(320~306)는 상이한 체크포인트 상태에 있을 수도 있다. 예컨대, 프로세스 304n의 상류측의 모든 프로세스(예컨대, 데이터 싱크 프로세스 306)는 체크포인트 상태 N을 완료하게 될 수도 있는 한편, 프로세스 304n의 하류측의 모든 프로세스(예컨대, 프로세스 304a 및 데이터 소스 프로세스 302)는 체크포인트 상태 N-1만을 완료하게 될 수 있다.

[0046] 시스템(300)이 처리를 재개할 준비가 된 때에, 내고장성 관리부(308)는 통신 채널(312a~312e)을 통해 각각의 프로세스에 하나 이상의 처리 재개 메시지를 전송한다. 처리 재개 메시지는 프로세스들에게 이들 프로세스가 실행할 가장 직전의 완전히 행해진(또는 완료된) 체크포인트 상태(예컨대, 체크포인트 상태 N-1)를 지시한다. 몇몇 예에서, 체크포인트 상태 N을 이미 완료하였을 수도 있는 프로세스는 단순히 체크포인트 상태 N-1에서부터 체크포인트 상태 N까지의 결과들을 재생할 수 있다. 이러한 방식으로, 프로세스(320~306)는 자신의 직전의 노력을 중복해서 행하는 것을 방지할 수 있다. 일례에서, 체크포인트 상태 N-1에서부터 체크포인트 상태 N까지의 결과들을 재생하는 것은 2개의 체크포인트 상태들 사이에서 발생하였을 수도 있는 보다 이전의 처리 동작의 결과를 재생하는 것을 포함한다.

[0047] 몇몇 예에서, 시스템 장애는 실질적으로 기동 직후에 발생할 수 있다. 이러한 상황에서, 프로세스(320~306)의 다수는 완료된 체크포인트 상태 0만을 가질 수 있다. 이들 프로세스(320~306)는 대응하는 체크포인트 레코드에 저장된 기동 값(start-up value) 및 초기화 데이터에 기초하여 체크포인트 상태 0에서부터 처리를 재개할 수 있다.

[0048] 도 4는 멀티-프로세스 시스템 내에서의 프로세스(예컨대, 도 3의 프로세스 302)의 일례의 실행을 보여주는 흐름도이다. 기동 시에, 프로세스는 자신의 초기 상태를 데이터 저장장치에 체크포인트 상태 0으로서 즉각적으로 저장한다(단계 402), 그리고나서, 프로세스가 별개의 실행 단계(예컨대, 실행 단계 1, 2,...N-1)로 실행될 수 있다. 각각의 실행 단계의 끝에서, 프로세스는 자신의 최종 상태(end state)를 데이터 저장장치에 저장할 수 있고, 예컨대 최종 상태가 체크포인트 상태로서 저장될 수 있다. 예컨대, 제1 실행 단계 후, 프로세스는 제1 실행 단계의 최종 상태를 체크포인트 상태 1로서 저장할 수 있다(단계 404). 유사하게, 후속 실행 단계 후에, 프로세스는 실행 단계의 최종 상태를 체크포인트 상태 2,...,N-1, N으로서 저장할 수 있다(단계 406-410).

[0049] 도 5는 외부 시스템이 장애가 발생하거나 또는 유지보수를 위해 오프라인으로 되는 이벤트에서 실행되는 일례의 단계를 보여주는 흐름도이다. 예컨대, 외부 시스템은 처리 시스템의 프로세스와 통신하는 데이터베이스(예컨대, 도 3의 데이터베이스 312)이어도 된다. 외부 시스템이 유지보수를 위해 오프라인으로 되거나, 또는 장애가 발생할 때에, 예컨대 장애가 발생한 외부 시스템과 통신하는 프로세스에 의해 에러 플래그가 발생될 수 있다(단계 502). 에러 플래그에 응답하여 중단 명령 메시지가 생성되고 프로세스를 통해 전파될 수 있다(단계 504). 프로세스가 중단 명령 메시지를 수신할 때에 각각의 프로세스의 현재 활동이 중단된다(단계 506). 또한, 마지막 체크포인트 상태 이후에 수행된 임의의 트랜잭션이 프로세스에 의해 폐기될 수도 있다. 외부 시스템에 대한 장애가 발생한 접속이 복구될 때까지 더 이상의 동작이 중단된다(단계 508). 접속이 복구될 때, 각각의 프로세스에 처리 재개 메시지가 전송된다(단계 510). 처리 재개 메시지는 프로세스가 이 체크포인트 상태에서부터 처리를 재개할 체크포인트 상태를 지시한다. 이와 같이, 각각의 프로세스는 자신의 연관 저장 영역으로부터 체크포인트 상태에 관한 관련 정보를 검색한다(단계 512).

[0050] 도 6은 프로세스를 실행하는 동안 체크포인트를 저장하고 이 체크포인트에서부터 재개할 시에 실행되는 일례의 단계를 도시하는 흐름도이다. 예컨대, 프로세스를 초기화할 시에, 프로세스의 초기 상태에 관한 정보가 연관된 저장 영역에 저장된다(단계 602). 그리고나서, 프로세스는 별개의 실행 단계로 실행된다. 이로써, 각각의 실행 단계의 종료 시에, 프로세스는 실행 단계의 최종 상태를 나타내는 정보를 저장한다(단계 604). 사전에 정해진 이벤트가 발생할 때, 예컨대 외부 장치에 대한 접속의 상실이 발생할 때(단계 608), 프로세스의 실행이 중단된다(단계 606). 한편, 프로세스는 중단을 유발한 이벤트가 해소되었는지(예컨대, 외부 장치에 대한 접속의 복구)를 확인한다. 이 시간 동안, 프로세스는 셧다운되지 않고, 이벤트가 해소된 것으로 간주될 때까지 처리가

중단된다. 프로세스의 실행은 마지막 저장된 초기 상태 또는 최종 상태에서부터 재개된다(단계 610).

[0051] 전술한 기술은 컴퓨터 상에서 실행하기 위한 소프트웨어를 이용하여 구현될 수 있다. 예컨대, 소프트웨어는 하나 이상의 프로세서, 하나 이상의 데이터 저장 시스템(휘발성 및 비휘발성 메모리 및/또는 저장 요소를 포함), 하나 이상의 입력 장치 또는 입력 포트, 및 하나 이상의 출력 장치 또는 출력 포트를 각각 포함하는, 하나 이상의 프로그래밍 되거나 또는 프로그래밍 가능한 컴퓨터 시스템(분산형, 클라이언트/서버형, 또는 그리드형 등의 다양한 아키텍처로 될 수 있음) 상에서 실행하는 하나 이상의 컴퓨터 프로그램에서의 프로시저(procedure)를 형성한다. 소프트웨어는 예컨대 데이터 흐름 그래프의 설계 및 구성에 관련된 다른 서비스를 제공하는 대형 프로그램의 하나 이상의 모듈을 형성할 수 있다. 그래프의 노드 및 요소는 컴퓨터 판독가능 매체에 저장된 데이터 구조 또는 데이터 레포지토리 내에 저장된 데이터 모델에 부합하는 다른 조직화된 데이터로서 구현될 수 있다.

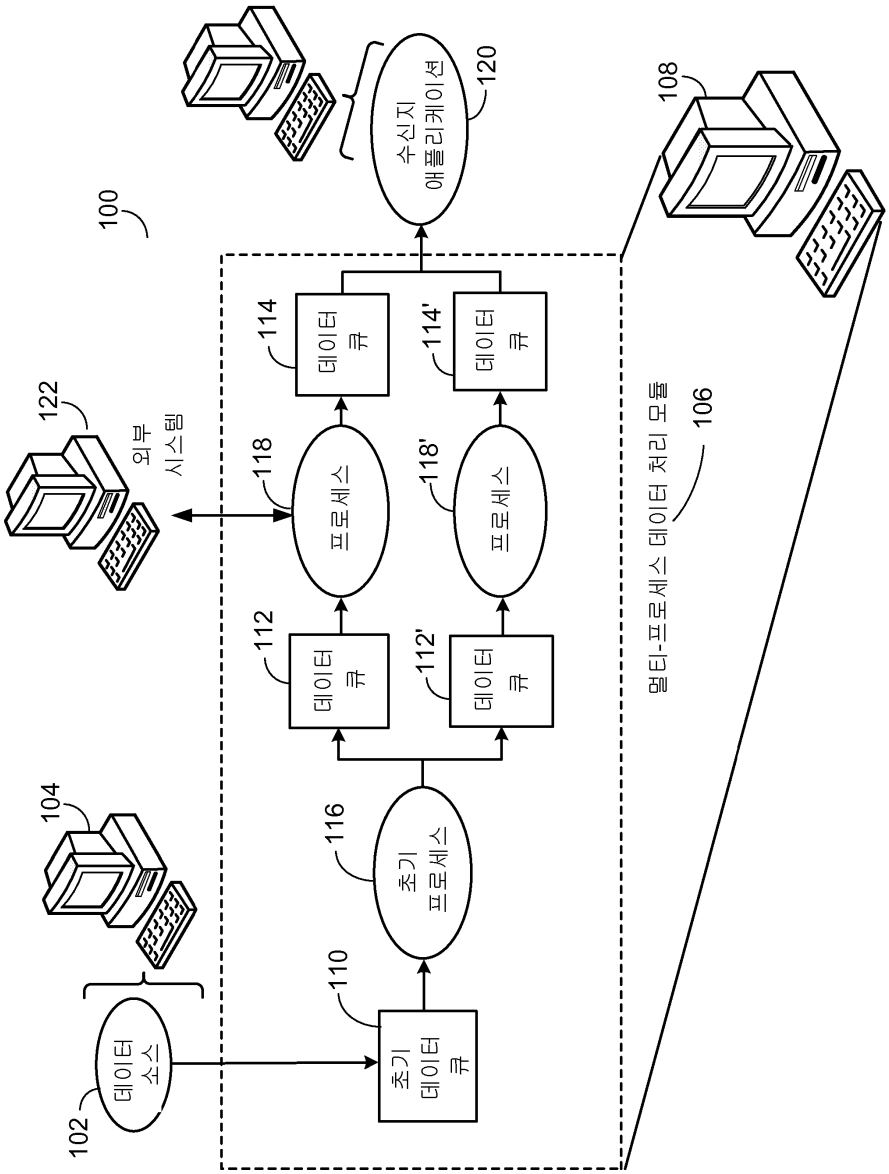
[0052] 소프트웨어는 범용 또는 전용의 프로그래밍 가능한 컴퓨터로 판독가능한 CD-ROM과 같은 저장 매체에 제공되거나, 네트워크의 통신 매체를 통하여 이러한 소프트웨어가 실행되는 컴퓨터로 전달(전파 신호로 부호화된 상태로)될 수 있다. 모든 기능은 전용의 컴퓨터상에서 수행될 수도 있고, 또는 코프로세서와 같은 전용의 하드웨어를 사용해서 수행될 수도 있다. 소프트웨어는 해당 소프트웨어에 의해 특정된 연산의 상이한 부분이 상이한 컴퓨터에 의해 수행되는 분산 방식으로 구현되어도 된다. 이러한 각각의 컴퓨터 프로그램은, 본 명세서에서 설명된 프로시저를 수행하기 위해 저장 매체 또는 저장 장치가 컴퓨터 시스템에 의해 판독될 때에 컴퓨터를 구성 및 운영하기 위한, 범용 또는 전용의 프로그래밍 가능한 컴퓨터에 의해 판독가능한 저장 매체 또는 저장 장치(예컨대, 솔리드 스테이트 메모리 또는 매체, 자기 또는 광학 매체)에 저장되거나 다운로드되는 것이 바람직하다. 본 발명의 시스템은 컴퓨터 프로그램에 맞게 구성된 컴퓨터 판독가능 저장 매체로서 구현될 수도 있으며, 이와 같이 구성된 매체는 컴퓨터 시스템을 본 명세서에 설명된 기능의 수행을 위해 특정되고 미리 정해진 방식으로 동작하게 한다.

[0053] 본 발명에 대하여 많은 실시예를 설명하였다. 그렇지만, 본 발명의 사상 및 범위를 벗어남이 없이 다양한 변형이 가능하다는 것을 알 수 있을 것이다. 예컨대, 전술한 단계들 중 몇몇은 반드시 그 순서대로 수행되지 않아도 되며, 설명된 것과 다른 순서대로 수행되어도 된다.

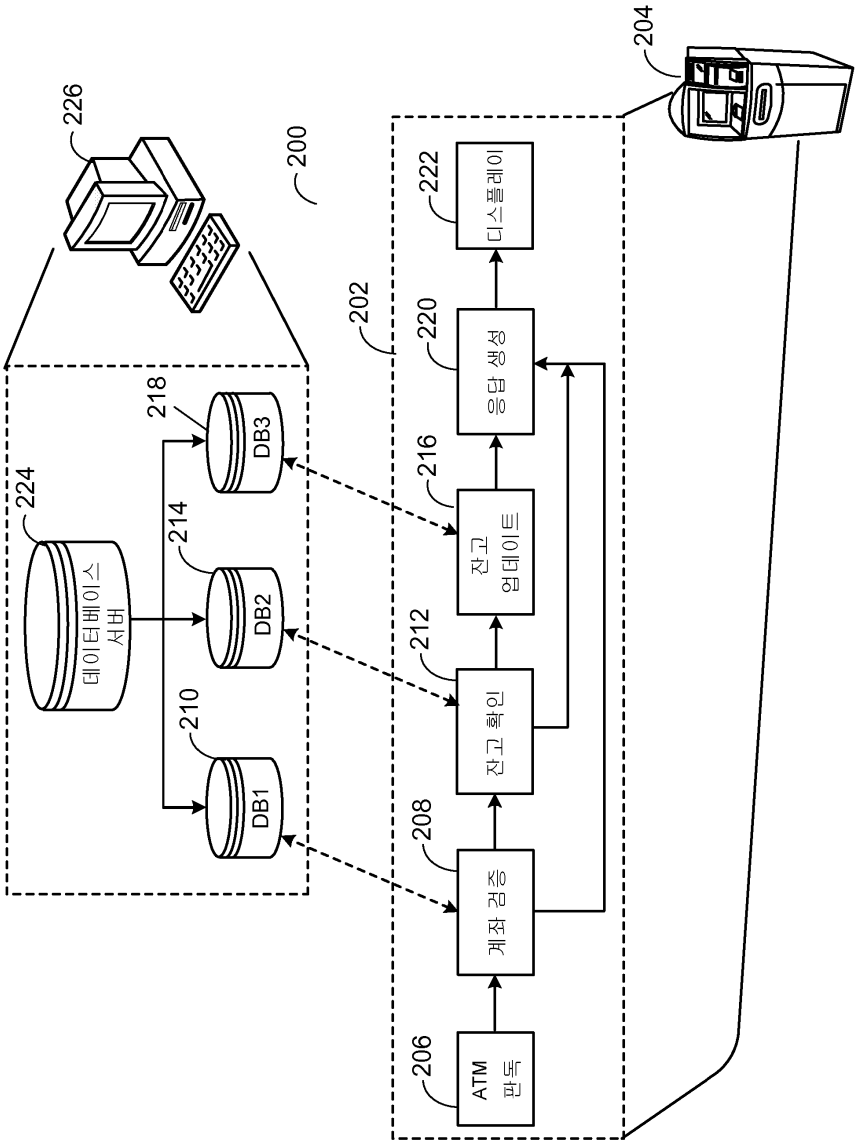
[0054] 이상의 설명은 청구범위에 의해 정해지는 본 발명의 범위를 제한하기 위한 것이 아니라 예시를 위한 것임을 이해할 것이다. 예컨대, 전술한 많은 기능적 단계들은 전체적인 처리에 실질적인 영향을 미치지 않으면서 다른 순서로 수행되어도 된다. 다른 실시예가 이하의 청구항들의 범위에 포함된다.

도면

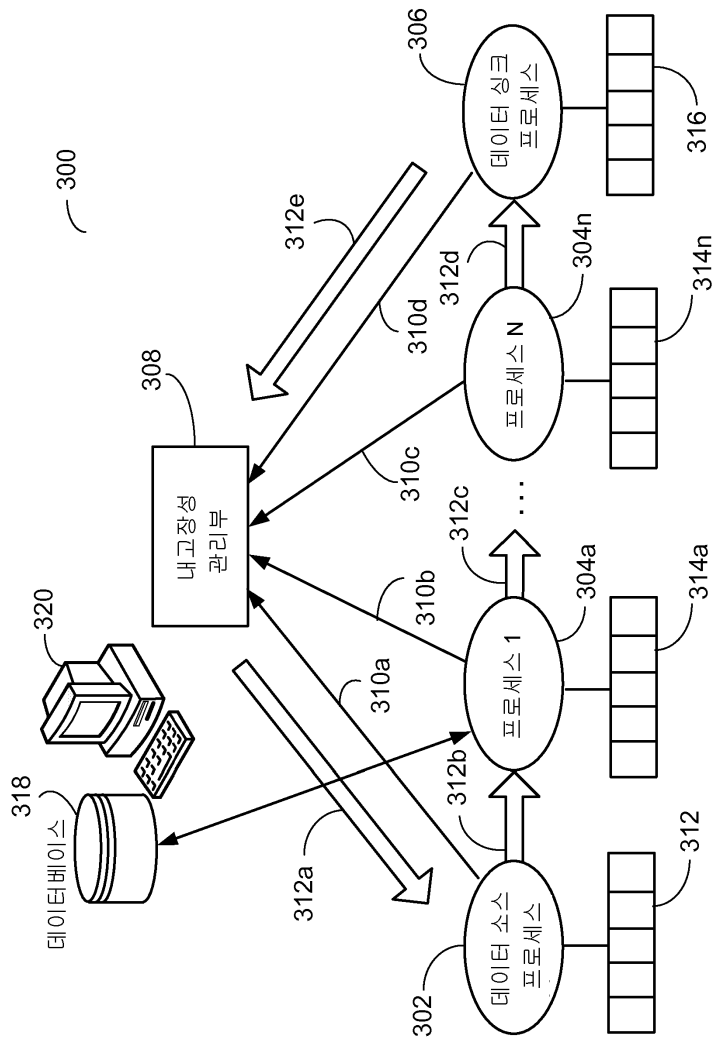
도면1


100
102
데이터 소스
104
108
110
초기 데이터 큐
116
초기 프로세스
112
데이터 큐
112'
데이터 큐
118
프로세스
118'
프로세스
114
데이터 큐
114'
데이터 큐
120
수신지 애플리케이션
122
외부 시스템
멀티-프로세스 데이터 처리 모듈
106

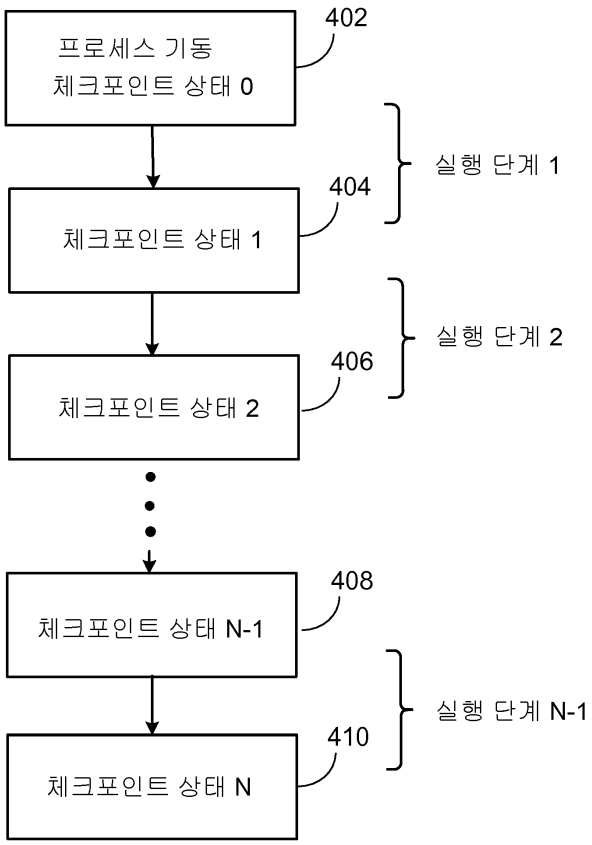
도면2


204
226
200
202
222
디스플레이
220
응답 생성
216
잔고 업데이트
212
잔고 확인
208
계좌 검증
206
ATM 판독
218
DB3
214
DB2
210
DB1
224
데이터베이스 서버

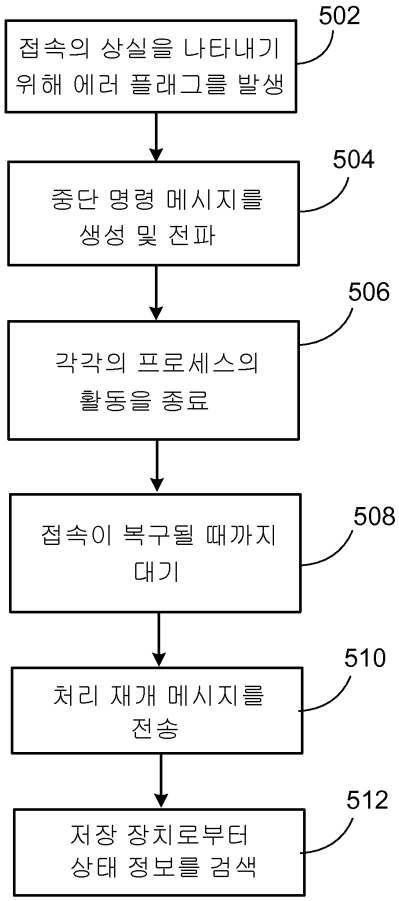
도면3


300
306
데이터 싱크 프로세스
316
312e
310d
312d
304n
프로세스 N
314n
308
내고장성 관리부
310c
312c
310b
304a
프로세스 1
314a
320
310a
318
데이터베이스
312b
312a
312
302
데이터 소스 프로세스

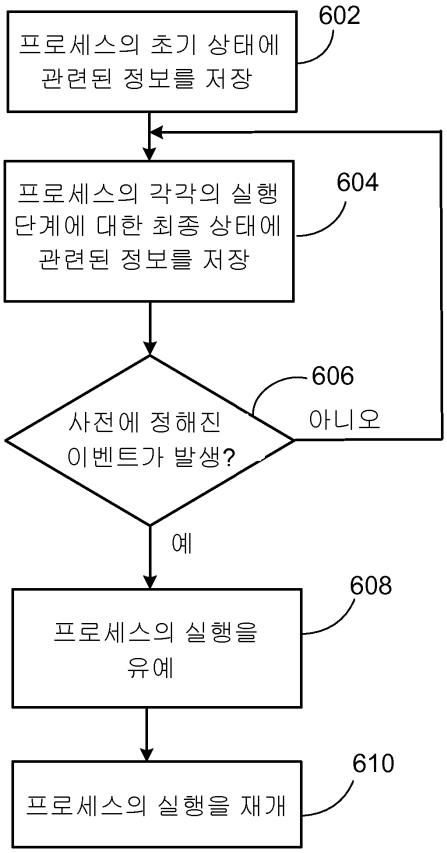
도면4


프로세스 기동
체크포인트 상태 0
402
체크포인트 상태 1
404
실행 단계 1
체크포인트 상태 2
406
실행 단계 2
체크포인트 상태 N-1
408
체크포인트 상태 N
410
실행 단계 N-1

도면5


502
접속의 상실을 나타내기 위해 에러 플래그를 발생
504
중단 명령 메시지를 생성 및 전파
506
각각의 프로세스의 활동을 종료
508
접속이 복구될 때까지 대기
510
처리 재개 메시지를 전송
512
저장 장치로부터 상태 정보를 검색

도면6


602
프로세스의 초기 상태에 관련된 정보를 저장
604
프로세스의 각각의 실행 단계에 대한 최종 상태에 관련된 정보를 저장
606
사전에 정해진 이벤트가 발생?
아니오
예
608
프로세스의 실행을 유예
610
프로세스의 실행을 재개