(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2010년06월09일
(11) 등록번호 10-0962893
(24) 등록일자 2010년05월31일

(51) Int. Cl.
H04L 29/08 (2006.01) *H04L 29/06* (2006.01)
(21) 출원번호 10-2008-7002545
(22) 출원일자(국제출원일자) 2006년07월26일
심사청구일자 2008년09월29일
(85) 번역문제출일자 2008년01월30일
(65) 공개번호 10-2008-0041634
(43) 공개일자 2008년05월13일
(86) 국제출원번호 PCT/EP2006/064658
(87) 국제공개번호 WO 2007/028670
국제공개일자 2007년03월15일
(30) 우선권주장
05108239.4 2005년09월08일
유럽특허청(EPO)(EP)
(56) 선행기술조사문헌
US20030145048 A1
WO0200127791 A1
US6012098 A
전체 청구항 수 : 총 10 항

(73) 특허권자
인터내셔널 비지네스 머신즈 코포레이션
미국 10504 뉴욕주 아몬크 뉴오차드 로드
(72) 발명자
라스플 스테판
독일 71732 탐 에케너스트라쎄 3
세일레트 야닉
독일 70563 슈투트가르트 베르그넬켄스트라쎄 4
(74) 대리인
김원준, 김창세, 장성구

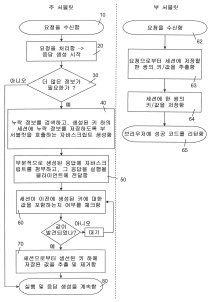
심사관 : 오제욱

(54) 클라이언트-서버 통신 개선 방법, 클라이언트-서버아키텍처 내의 서버, 클라이언트-서버 아키텍처 시스템 및컴퓨터 판독가능한 저장 매체

(57) 요 약

본 발명은 기존의 통신 프로토콜 및 클라이언트를 변경하지 않으면서 클라이언트-서블릿 통신을 개선하는 방법 및 시스템을 제공한다. 기존의 종래 기술인 클라이언트와 서블릿 사이의 단방향 통신 경로는 최초 요청이 원하는 정보를 검색하기 위한 모든 정보를 포함하면 변경되지 않는다. 그러나, 서블릿이 원하는 정보를 검색하라는 초기 클라이언트의 웹브라우저 요청에 포함되지 않은 누락 정보를 식별하는 경우에, 서블릿은 상기 초기 HTTP-요청의 HTTP-응답 기능을 사용함으로써 서블릿에 누락 정보를 제공하는 다른 통신 경로를 자동으로 개방하는데, 다른 통신 경로는 서블릿에 의해 스크립트를 생성하는 단계 -클라이언트의 웹브라우저에서 실행되는 경우에 누락 정보를 검색하고 또 다른 서블릿 기능 부품을 호출함- 와, 부분적인 응답으로서 나타내는 HTTP-응답에 스크립트를 첨부하는 단계와, 클라이언트의 웹브라우저에 스크립트를 포함하는 HTTP-응답을 전달하는 단계와, 누락 정보가 이용가능해질 때까지 서블릿에 의해 초기 HTTP-응답의 실행을 중지하는 단계와, 또 다른 서블릿 기능 부품에 의해 누락 정보를 수신하는 단계 -누락 정보는 클라이언트의 웹브라우저에서 실행하는 동안 스크립트에 의해 생성된 새로운 HTTP-요청에 포함됨- 와, 서블릿에 누락 정보를 제공하는 단계와, HTTP-응답의 나머지를 검색하기 위해 누락 정보를 사용하여 서블릿에 의해 초기 HTTP-응답의 실행을 계속하고, 디스플레이하기 위해 HTTP-응답의 나머지를 클라이언트의 웹브라우저로 제공하는 단계를 특징으로 하는 또 다른 서블릿 기능 부품의 지원을 받는다.

대 표 도 - 도1



특허청구의 범위

청구항 1

기존의 통신 프로토콜 및 클라이언트를 변경하지 않으면서 클라이언트(1)-서버(2) 통신을 개선하는 방법 -상기 통신 프로토콜은 상기 클라이언트(1)에 의한 서블릿(8)으로부터 정보를 검색하라는 요청을 개시하는 단방향 통신 경로와, 적어도 상기 요청의 성공 또는 실패를 명시하는 리턴 코드를 포함하고 이용가능하면 상기 요청의 결과를 포함하는 상기 서블릿(8)에 의한 응답을 전달하는 것을 특징으로 함- 에 있어서,

상기 서버 측에서 상기 방법은,

상기 서블릿(8)이 완전한 응답을 제공하기 위해 최초 요청에 포함되지 않은 누락 정보를 식별하면,

상기 최초 요청의 응답 기능을 사용함으로써 상기 누락 정보를 상기 서블릿(8)에 제공하기 위해 상기 서블릿(8)으로부터 상기 클라이언트의 웹브라우저를 통해 상기 서블릿(8)으로의 다른 통신 경로를 자동으로 개방하는 단계 -상기 다른 통신 경로는 또 다른 서블릿 기능 부품(10)의 지원을 받음- 와,

상기 서블릿(8)에 의해, 상기 클라이언트의 웹브라우저에서 실행되는 경우에 상기 누락 정보를 검색하고 상기 또 다른 서블릿 기능 부품(10)을 호출하는 스크립트(12)를 생성하는 단계와,

상기 응답에 상기 스크립트(12)를 부분적인 응답으로서 첨부하는(appending) 단계와,

상기 클라이언트의 웹브라우저에 상기 스크립트(12)를 포함하는 상기 응답을 전달하는 단계와,

상기 누락 정보가 이용가능해질 때까지 상기 서블릿(8)에 의한 최초 응답의 실행을 중지하는 단계와,

상기 또 다른 서블릿 기능 부품(10)에 의해 상기 누락 정보를 수신하는 단계 -상기 누락 정보는 상기 클라이언트의 웹브라우저에서 실행하는 동안 상기 스크립트(12)에 의해 생성된 새로운 요청에 포함됨- 와,

상기 서블릿(8)에 상기 누락 정보를 제공하는 단계와,

상기 응답의 나머지를 검색하기 위해 상기 누락 정보를 사용하여 상기 서블릿(8)에 의한 상기 최초 응답의 실행을 계속하고, 디스플레이하기 위해 상기 응답의 나머지를 상기 클라이언트의 웹브라우저로 제공하는 단계를 포함하는

클라이언트-서버 통신 개선 방법.

청구항 2

제 1 항에 있어서,

상기 통신 프로토콜은 HTTP-프로토콜인

클라이언트-서버 통신 개선 방법.

청구항 3

제 1 항에 있어서,

상기 또 다른 서블릿 기능 부품(10)은 개별적인 부 서블릿(secondary servlet)으로 구현되는

클라이언트-서버 통신 개선 방법.

청구항 4

제 3 항에 있어서,

상기 부 서블릿(10)은 상기 스크립트(12)에 의해 전달된 새로운 HTTP-요청을 수신하고, 2개의 서블릿(8,10)에 의해 사용된 공유 메모리에 상기 누락 정보를 저장하는

클라이언트-서버 통신 개선 방법.

청구항 5

제 3 항에 있어서,

상기 누락 정보는 상기 부 서블릿(10)에 의해 상기 클라이언트(1)와 상기 서버(2) 사이의 현재의 HTTP-세션 객체에 저장되고 상기 서블릿에 전달되는

클라이언트-서버 통신 개선 방법.

청구항 6

제 1 항에 있어서,

상기 또 다른 서블릿 기능 부품(10)은 상기 서블릿(8)의 일부이고, 상기 서블릿에 의한 새로운 HTTP-요청의 수신은 상기 서블릿의 제 2 인스턴스(instance)를 자동으로 시작하며, 상기 누락 정보로 현재의 HTTP-세션 객체를 갱신하는

클라이언트-서버 통신 개선 방법.

청구항 7

제 1 항에 있어서,

상기 웹브라우저에 의한 상기 스크립트(12)의 실행은 상기 누락 정보를 자동으로 검색하거나 다른 사용자 입력에 의해 상기 누락 정보를 검색하는

클라이언트-서버 통신 개선 방법.

청구항 8

제 1 항 내지 제 7 항 중 어느 한 항에 따른 방법을 이용하는 클라이언트-서버 아키텍처 내의 서버.

청구항 9

정보를 검색하는 적어도 하나의 서블릿을 구비한 클라이언트-서버 아키텍처 시스템 -상기 클라이언트-서버 아키텍처는 통신 프로토콜을 사용하고, 상기 통신 프로토콜은 상기 클라이언트의 웹브라우저에 의한 상기 서블릿으로부터 정보를 검색하라는 요청을 개시하는 단방향 통신 경로와, 적어도 상기 요청의 성공 또는 실패를 명시하는 리턴 코드를 포함하고 이용가능하면 상기 요청의 결과를 포함하는 상기 서블릿에 의한 응답을 전달하는 것을 특징으로 함- 에 있어서,

서버 측은,

a) 완전한 응답을 제공하기 위해 최초 요청에 포함되지 않은 누락 정보를 식별하는 수단과,

b) 상기 수단이 누락 정보를 식별하면 상기 최초 요청의 응답 기능을 사용함으로써 상기 누락 정보를 상기 서블릿에 제공하기 위해 상기 서블릿으로부터 상기 클라이언트의 웹브라우저를 통해 상기 서블릿으로의 다른 통신 경로를 자동으로 개방하는 수단을 포함하며,

상기 다른 통신 경로는

c) 상기 서블릿에 의해, 상기 클라이언트의 웹브라우저에서 실행되는 경우에 상기 누락 정보를 검색하고 상기 또 다른 서블릿 기능 부품을 호출하는 스크립트를 생성하는 수단 - 과,

d) 상기 응답에 상기 스크립트를 부분적인 응답으로서 첨부하는 수단과,

e) 상기 클라이언트의 웹브라우저에 상기 스크립트를 포함하는 상기 응답을 전달하는 수단과,

f) 상기 누락 정보가 이용가능해질 때까지 상기 서블릿에 의한 상기 최초 응답의 실행을 중지하는 수단과,

g) 상기 누락 정보를 수신하는 수단 -상기 누락 정보는 상기 클라이언트의 웹브라우저에서 실행하는 동안 상기 스크립트에 의해 생성된 새로운 요청에 포함됨- 과,

h) 상기 서블릿에 상기 누락 정보를 제공하는 수단과,

i) 상기 응답의 나머지를 검색하기 위해 상기 누락 정보를 사용하여 상기 서블릿에 의한 상기 최초 응답의 실행을 계속하고, 디스플레이하기 위해 상기 응답의 나머지를 상기 클라이언트의 웹브라우저로 제공하는 수단을 포

함하는

클라이언트-서버 아키텍처 시스템.

청구항 10

제 1 항 내지 제 7 항 중 어느 한 항에 따른 방법을 수행하는 컴퓨터 실행가능한 인스트럭션을 포함하는

컴퓨터 판독가능한 저장 매체.

명 세 서

기 술 분 야

[0001] 본 발명은 일반적으로 클라이언트-서버 통신에 관한 것으로, 구체적으로 기존의 통신 프로토콜 및 클라이언트를 변경하지 않으면서 월드 와이드 웹(web) 내의 클라이언트-서블릿(servlet) 통신을 개선하는 것에 관한 것이며, 보다 구체적으로는 서블릿이 원하는 정보를 검색하라는 초기 클라이언트의 웹브라우저 요청에 포함되지 않은 누락 정보를 식별하는 경우에 이러한 서블릿의 유연성 및 상호작용 강화에 관한 것이다.

배 경 기 술

[0002] 용어 "클라이언트-서버"는 통신 프로토콜(예컨대, HTTP)을 통해 서로 통신하는 클라이언트와 서버들로 구성된 네트워크 애플리케이션 아키텍처를 지칭한다.

[0003] 웹은 리소스로 지칭되는 관심 항목이 인터넷 식별자(URI:Uniform Resource Identifier)로 불리는 포괄적 식별자에 의해 식별되는 정보 공간을 지칭한다. 웹은 정보의 각 페이지가 그 페이지가 발견될 수 있는 고유한 "주소"로 주어지는 방법을 명시하는 URL(Uniform Resource Locator), 클라이언트와 서버가 서로에게 정보를 전달하는 방법을 명시하는 HTTP(Hyper Text Transfer Protocol) 및 정보를 인코딩하는 방법으로 다양한 장치 상에서 디스플레이될 수 있는 HTML(Hyper Text Markup Language)과 같은 3 가지 표준으로 구성된다.

[0004] 웹은 다음과 같이 동작한다. 클라이언트의 웹브라우저(예컨대, 마이크로소프트사의 인터넷 익스플로러, 애플사의 사파리)가 개방될 때, HTTP를 사용하여 클라이언트(또는 사용중인 컴퓨터)를 대신하여 통신하고 웹페이지 요청을 한다. 일단 요청이 전달되면, 클라이언트의 웹브라우저 컴퓨터는 서버로부터 하이퍼텍스트 데이터 스트림을 대기할 것이다. 서버가 요청을 받으면, 그 서버는 요청된 파일을 찾고, 만일 존재한다면, 요청한 대로 그 파일을 클라이언트의 웹브라우저에 전달한다.

[0005] 서블릿은 서버에서 실행하는 프로그램으로서 클라이언트의 웹브라우저로부터 발생하는 요청을 대기하고 이러한 요청에 대하여 (예컨대, 동적으로 생성된 웹페이지로서) 응답을 생성한다.

[0006] 종래 기술 방식의 서블릿 작업은 다음과 같다. 클라이언트의 웹브라우저가 URL을 호출함으로써 서블릿에 HTTP 요청을 전달한다. 클라이언트의 웹브라우저는 반드시 자신이 서블릿을 호출하는 중임을 알고 있는 것은 아니다. 서블릿을 호출하는 것은 다른 웹페이지를 로딩하는 것과 유사하되, 차이점은 요청시에 파라미터가 서블릿에 전달될 수 있고 서버에 의해 반환된 응답이 정적 문서가 아니라 동적으로 생성된다는 것이다.

[0007] 요청은 서블릿을 호스팅하는 애플리케이션 서버에 의해 분석되고 요청된 URL에 대응하는 서블릿에 전달된다.

[0008] 요청은 서블릿에 의해 분석되고, 최종 파라미터는 파싱 및 해석되며, 클라이언트의 웹브라우저에 리턴하는 문서는 동적으로 생성된다(문서의 콘텐츠는 요청시에 클라이언트의 웹브라우저에 의해 전달된 최종 파라미터에 의존할 수 있다).

[0009] 생성된 문서는 클라이언트의 웹브라우저에 의해 수신되고 디스플레이되며 또는 (생성된 문서가 자바스크립트와 같은 몇몇 실행가능한 스크립트를 포함하는 경우) 해석된다.

[0010] 서블릿은 동적 콘텐츠를 나타내는 강력하고 스케일링 가능한 프레임워크를 제공함으로써 인터넷을 변환하는데, 이는 콘텐츠가 정기적으로 갱신되어야 하는 정적 문서를 발행하는 대신에, 서블릿이 클라이언트의 웹브라우저가 HTTP에 파라미터를 추가하게 한다 -요청하고 그 요청에 기반하여 현재의 정보를 획득함. 서블릿이 없으면, 사

용자가 그 사람의 클라이언트의 웹브라우저로부터 서버와 상호작용하는 경우에 대화식 웹 애플리케이션을 생성하는 것은 불가능할 것이다.

[0011] 그러나, 서블릿은 하나의 한계가 있는데, 이는 요청이 수신된 후에, 서블릿이 클라이언트의 웹브라우저에 누락 정보를 요청할 방법이 없다는 것이다.

[0012] 서블릿은 착신 요청을 사용하여 서버에서 실행되는 작용에 영향을 주는 파라미터 리스트 또는 서버에 의해 리턴되는 정보를 수신할 수 있다. 예컨대, 인터넷에서 무언가를 주문하면, "보내기" 버튼을 누르고 주문된 품목의 리스트를 포함하는 요청을 서버에 전달할 것이다. 서버는 이 리스트를 이용하여 구동하고(데이터 갱신 등), 보통 선택된 모든 품목을 열거하는 확인 페이지를 생성한다.

[0013] 그러나 종래의 서블릿을 사용하면, 서버에서 요청을 처리하는 데 필요한 모든 파라미터가 클라이언트의 웹브라우저에 의해 전달된 초기 요청에 포함되어야 하는데, 만일 요청을 처리하는 동안 서블릿이 초기 요청시에 클라이언트의 웹브라우저에 의해 몇몇 다른 정보가 제공되지 않음을 알리면, 그 처리를 중지시키고, 클라이언트의 웹브라우저에 누락 정보를 요청하며 처리가 인터럽트되었던 지점으로부터 그 처리를 다시 계속하게 할 방법은 존재하지 않는다. 이는 서블릿의 애플리케이션 영역을 제한한다.

[0014] 클라이언트의 웹브라우저는 서블릿이 필요로 하는 정보가 무엇인지를 정확히 알아야 한다.

[0015] 요청시에, 클라이언트의 웹브라우저는 서블릿이 정말로 필요로 하는 것보다 많은 정보를 제공할 필요가 있다.

[0016] 종래의 서블릿은 클라이언트의 웹브라우저 요청에 의해 제공된 것보다 많은 정보를 요구하는 예측되지 않은 경우를 처리할 수 없다. 만일 요청이 필요한 모든 정보를 포함하지 않으면, 그 문제점을 해결하기 위해 종래의 방법은 처리를 중단하고 웹페이지를 클라이언트의 웹브라우저에 리턴하여, 원래의 페이지로 되돌리고 에러 메시지를 디스플레이하거나 누락 정보를 요청하고 새로운 정보를 가진 동일하거나 다른 서블릿을 호출하는 새로운 페이지를 디스플레이한다.

[0017] 이것은 하나의 서블릿 호출시에 요청 처리를 중단하는 것 및 다른 서블릿 호출시에 트랜잭션을 계속하는 것이 간단하지 않을 수 있으므로 누락 정보를 처리하기에 상당히 좋은 방법은 아니다. 또한, 상당히 자주, 동일한 페이지가 사용자에게 어떤 정보가 누락되고 있는지에 대하여 알리는 작은 주석을 사용하여 다시 디스플레이되므로 사용자를 혼란스럽게 할 수 있으며, 이는 유용성 관점에서 문제가 있다.

발명의 상세한 설명

[0018] 본 발명의 목적은 기존의 통신 프로토콜 및 클라이언트를 변경하지 않으면서 월드 와이드 웹(web) 내의 클라이언트-서블릿 통신을 개선하며, 보다 구체적으로는 서블릿이 원하는 정보를 검색하라는 초기 클라이언트의 웹브라우저 요청에 포함되지 않은 누락 정보를 식별하는 경우에 이러한 서블릿의 유연성 및 상호작용을 향상시키는 방법 및 시스템을 제공하는 것이다.

[0019] 본 발명은 기존의 통신 프로토콜 및 클라이언트를 변경하지 않으면서 클라이언트-서블릿 통신을 개선하는 방법 및 시스템을 제공한다.

[0020] 기존의 종래 기술인 클라이언트와 서블릿 사이의 단방향 통신 경로는 초기 요청이 원하는 정보를 검색하기 위한 모든 정보를 포함하면 변경되지 않는다. 그러나, 서블릿이 원하는 정보를 검색하라는 초기 클라이언트의 웹브라우저 요청에 포함되지 않은 누락 정보를 식별하는 경우에, 서블릿은 초기 HTTP-요청의 HTTP-응답 기능을 사용함으로써 서블릿에 누락 정보를 제공하는 다른 통신 경로를 자동으로 개방하는데, 다른 통신 경로는 서블릿에 의해 스크립트를 생성하는 단계 -클라이언트의 웹브라우저에서 실행되는 경우에 누락 정보를 검색하고 또 다른 서블릿 기능 부품을 호출함- 와, 부분적인 응답으로서 나타내는 HTTP-응답에 스크립트를 첨부하는 단계와, 클라이언트의 웹브라우저에 스크립트를 포함하는 HTTP-응답을 전달하는 단계와, 누락 정보가 이용가능해질 때까지 서블릿에 의해 초기 HTTP-응답의 실행을 중지하는 단계와, 또 다른 서블릿 기능 부품에 의해 누락 정보를 수신하는 단계 -누락 정보는 클라이언트의 웹브라우저에서 실행하는 동안 스크립트에 의해 생성된 새로운 HTTP-요청에 포함됨- 와, 서블릿에 누락 정보를 제공하는 단계와, HTTP-응답의 나머지를 검색하기 위해 누락 정보를 사용하여 서블릿에 의해 초기 HTTP-응답의 실행을 계속하고, 디스플레이하기 위해 HTTP-응답의 나머지를 클라이언트의 웹브라우저로 제공하는 단계를 특징으로 하는 또 다른 서블릿 기능 부품의 지원을 받는다.

[0021] 서블릿이 종래 기술에 설명된 한계를 가지는 이유는 HTTP 프로토콜 및 웹브라우저/서버가 설계되는 방법에서 유

래한다.

[0022] HTTP는 클라이언트의 웹브라우저와 웹서버 사이의 통신에 사용된 프로토콜이다.

[0023] 보통 HTTP 프로토콜은 서버(POST 요청)에 몇몇 정보를 전달하거나 서버(GET 요청)로부터 문서를 획득하도록 클라이언트의 웹브라우저가 웹서버(또는 애플리케이션 서버/서블릿)에 요청을 전달하게 한다.

[0024] 클라이언트의 웹브라우저에 의해 전달된 요청의 유형(GET 또는 POST)에 상관없이, 서버는 항상 그 요청을 처리하고 리턴 코드(예컨대, 요청이 성공적으로 처리될 수 있는지 없는지 여부를 나타냄)를 구비하는 헤더를 포함하는 응답 및 최종 결과 문서를 리턴해야 한다. 요청이 성공적이었으면, 클라이언트의 웹브라우저는 보통 요청을 처리하였던 요소의 링크에 포함된 원래의 페이지를 서버로부터의 응답으로 대체하는, 서버에 의해 리턴되는 문서를 디스플레이한다.

[0025] 만일 요청이 장시간 실행되면, 서버는 전체 문서 생성이 완료되기 전에 응답 문서의 일부를 정기적으로 전달할 수 있다. 이것은 사용자가 전체 문서가 생성 및 전달되기를 기다려야할 필요 없이 문서의 시작을 판독할 수 있는 방법이다.

[0026] 이것은 프로토콜만이 클라이언트의 웹브라우저에 의해 개시된 통신을 예측하는 단방향 통신이다. 서버는 보통 클라이언트의 웹브라우저에 요청을 전달할 방법이 없다.

[0027] 이러한 한계를 극복하기 위해, 본 발명은 전체 응답 문서가 생성되기 전에 서버가 클라이언트의 웹브라우저에 "부분적인 문서"를 전달할 수 있는 사실 및 전체 문서가 완전히 전달되기 전에 이러한 부분적인 문서가 클라이언트의 웹브라우저에 의해 실행될 수 있는 자바스크립트를 포함할 수 있다는 사실을 이용한다. 자바스크립트는 HTML 페이지에 내장될 수 있고 클라이언트의 웹브라우저에 의해 해석되는 언어이다.

[0028] 본 발명은 다음과 같이 동작하는 웹클라이언트와 웹서버 사이의 양방향 통신을 제공한다(도 1 참조).

[0029] 클라이언트의 웹브라우저는 애플리케이션 서버 상의 주 서블릿에 HTTP-요청을 전달한다. 그 요청에 있어서 서블릿이 필요로 하는 몇몇 초기 파라미터를 제공하여 그 요청을 처리할 수 있다.

[0030] 주 서블릿은 요청을 수신하고(10) 최종 파라미터를 추출 및 해석하며 서버 측에서 응답 문서를 생성하는 몇몇 동작을 수행함으로써 그 요청을 처리하기 시작하는데(20), (지금까지는, 종래 기술의 서블릿이 동작하는 방법이다). 이러한 요청을 처리하는 동안, 주 서블릿은 몇몇 정보가 누락되고 있으며 이러한 정보가 요청의 발신자에 의해 제공되어야 함을 알린다(30). 이러한 정보는 사용자가 입력해야만 하는 몇몇 누락 데이터, 요청이 처리되는 동안 클라이언트의 웹브라우저 자체에 대한 또는 클라이언트의 웹브라우저에 디스플레이된 페이지 내의 기타 요소의 상태에 대한 몇몇 시스템 정보일 수 있다. 이것은 클라이언트의 웹브라우저에서 자바스크립트를 실행함으로써 사용자 상호작용으로 또는 사용자 상호작용 없이 검색될 수 있는 정보이어야 한다.

[0031] 주 서블릿은 요청의 처리 및 응답의 생성을 중지하고, 클라이언트의 웹브라우저에서 실행되는 경우에 후속하는 동작을 수행하는 자바스크립트를 생성한다(40).

[0032] 주 서블릿은 누락 정보를 검색 (또는 사용자에게 누락 정보를 요청)한다.

[0033] 주 서블릿은 추가로 검색된 정보를, 정보를 요청하는 서블릿에 의해 생성된 파라미터 및 키로서 전달하는 부 서블릿을 호출한다.

[0034] 주 서블릿은 부분적으로 생성된 응답 문서에 이전 단계에서 생성된 자바스크립트를 첨부하고, 그 부분적인 응답을 클라이언트의 웹브라우저에 전달한다(50).

[0035] 주 서블릿은 이전에 생성되었던 키 하에서, 웹 세션 또는 공유 메모리에서 찾고 있는 정보를 발견할 수 있을 때까지 대기함으로써 실행을 중지한다(60).

[0036] 한편 클라이언트의 웹브라우저는 주 서블릿으로부터 수신된 부분적인 문서를 디스플레이하기 시작한다. 만일 몇몇 콘텐츠가 이미 생성되었다면, 그 콘텐츠가 디스플레이될 것이다. 응답에 내장된 자바스크립트는 실행된다.

[0037] 부분적인 응답 문서에 내장된 자바스크립트를 실행하는 동안, 클라이언트의 웹브라우저는 주 서버가 요청한 누락 정보를 검색하고, 이러한 정보 및 그 하에서 정보가 저장되어야 하는 키를 부 서블릿의 URL로 인코딩하며, 이러한 부 URL에 요청을 전달한다.

[0038] 주 서블릿에 대한 요청이 여전히 구동하는 동안 이러한 2차 요청의 결과는 클라이언트의 웹브라우저에 의해 디

스플레이된 현재의 문서를 대체해서는 안 되며 1차 요청을 인터럽트해서도 안 된다는 것이 중요하다. 이는 2차 요청의 결과를 개별 프레임 또는 페이지의 특정 요소(영상, 내장된 프레임)로 지시방향을 변경하거나, 자바스크립트 API를 사용하여 HTTP 요청을 프로그램적으로 개방함으로써 달성될 수 있다.

[0039] 부 서블릿은 세션 (또는 주 서블릿과 공유된 메모리)에 저장하기 위해 파라미터에 정보를 가진 한 쌍의 키/값을 포함하라는 요청을 수신한다(62). 이어서 부 서블릿은 클라이언트의 웹브라우저에 성공 코드를 리턴한다(65). 클라이언트의 웹브라우저는 부 서블릿의 응답을 무시할 수 있다.

[0040] 주 서블릿은 세션(공유 메모리)에서 생성되었던 키에 값이 저장되었음을 검출한다. 주 서블릿은 이러한 값(요청을 처리하도록 누락되고 있었던 정보)을 추출하고, 세션/공유 메모리로부터 그 값을 제거하며(70), 요청의 처리를 다시 시작한다.

[0041] 처리는 응답 문서의 나머지가 생성되는 것으로 계속된다. 요청을 처리하는 동안 더 많은 정보가 나중에 누락되고 있음을 나타내면, 이어서 사용자에게 많은 정보를 요청하도록 마지막 단계가 반복될 수 있다(80).

[0042] 마지막으로 요청이 완료되고 클라이언트의 웹브라우저가 완전한 응답 문서를 수신 및 디스플레이한다(80).

[0043] 본 발명의 바람직한 실시예에서, 서블릿은 자바 서블릿으로서 구현된다. 그러나, 동일한 방법이 CGI 스크립트 또는 서버에서 호스팅되고 HTTP 요청에 응답하여 동적 콘텐츠를 제공하는 임의의 서블릿-유사 프로그램에 사용될 수 있다.

[0044] 본 발명의 다른 바람직한 실시예에서, 스크립트는 자바 스크립트로서 구현된다. 그러나, 누락 정보를 검색하도록 클라이언트의 웹브라우저에 리턴되는 스크립트(자바스크립트)는 웹페이지에 내장되고 클라이언트의 웹브라우저에 의해 실시간으로 실행될 수 있는 어떠한 언어(자바, 비주얼 베이직 등)로도 존재할 수 있다.

[0045] 본 발명의 다른 실시예에서, 사용자로부터 (마우스 또는 키 이벤트와 같은) 상당히 낮은 레벨의 정보를 수신하는 일반적인 서블릿을 생성하고, 이러한 정보의 복잡한 처리를 수행하는 것이 가능하다. 클라이언트의 웹브라우저는 사용자에 의해 산출된 마우스/키 이벤트에 대한 정보를 간단히 전달하고, 요청을 전달하는 경우에 서블릿이 무엇을 해야 하는지 또는 이러한 정보를 사용하여 어떤 일을 해야 할 것인지를 알지 못한다. 서블릿은 이러한 정보를 해석하고, 특정 이벤트가 발생할 때 동작이 취해져야 한다고 판단하면, 클라이언트의 웹브라우저에 이벤트 또는 클라이언트의 웹브라우저 측 상의 기타 요소에 대한 더 많은 정보를 제공하고 클라이언트의 웹브라우저에 의해 리턴되는 정보에 따라 이벤트의 동적 처리를 실행하라고 요청할 수 있다.

[0046] 본 발명의 또 다른 실시예에서, 서블릿은 복잡한 처리 경로와 함께 사용된다. 초기 요청은 처리를 시작하기에 충분한 정보를 포함한다. 처리하는 동안, 서블릿은 파라미터에 따라 상이한 실행 경로 또는 백-엔드 시스템(데이터베이스 등)으로부터 서버에 의해 검색된 데이터를 이용할 수 있다. 만일 어떤 시점에서 사용자가 처리를 계속하는 방법에 대하여 결정을 내려야 한다면, 서블릿은 사용자에게 결정을 촉구하도록 클라이언트의 웹브라우저에 요청할 수 있다. 이러한 결정은 요청을 인터럽트하지 않으면서 요청의 다른 처리에 영향을 줄 수 있다. 처리는 단일 트랜잭션으로 실행될 수 있다.

[0047] 본 발명의 다른 바람직한 실시예에서, 부 서블릿이 추가로 제공될 수 있다. 부 서블릿은 자바스크립트에 의해 전달된 새로운 HTTP-요청을 수신하고, 현재의 HTTP-세션 객체에 한 쌍의 키 값을 저장한다.

[0048] 본 발명의 다른 실시예에서, 부 서블릿을 사용하는 대신에, 자바스크립트는 주 서블릿에 HTTP-세션을 갱신하라는 요청을 전달할 수 있다. 그러나 이러한 경우에 (여전히 1차 요청에 대한 응답을 생성하고 있으며 누락 값을 대기하는 제 1 인스턴스와 병렬로) 주 서블릿의 제 2 인스턴스가 시작할 것이다.

[0049] 본 발명의 다른 실시예에서, HTTP-프로토콜은 클라이언트의 웹브라우저(3)에 의해 서블릿(8)으로부터 정보를 검색하라는 요청을 개시하고, 서블릿에 의해 적어도 요청의 성공 또는 실패를 명시하는 리턴 코드를 포함하고 (이용가능하면) 요청의 결과를 포함하는 응답을 전달하는 단방향 통신 경로를 특징으로 하는 다른 통신 프로토콜로 대체될 수 있다.

[0050] 도면을 예로써만 참조함으로써 본 발명의 후속하는 바람직한 실시예를 보다 상세히 설명할 것이다.

실 시 예

[0055] 본 발명은 클라이언트(1) 및 서버(2) 측으로 구성된다. 클라이언트 측(1)은 웹 서버에 HTTP 요청을 전달하고

서버에 의해 리턴되는 HTML 문서를 디스플레이할 수 있는 표준 클라이언트의 웹브라우저(3)인 것이 바람직하다. 클라이언트의 웹브라우저(3)는 문서(13)에 내장된 자바스크립트(12)를 실행하여 디스플레이할 수 있어야 한다. 또한, 클라이언트의 웹브라우저(3)는 전체 문서(13)가 전달되기 전에 서버에 의해 리턴된 문서(13)의 일부분에 내장된 자바스크립트(12)를 디스플레이 및 실행할 수 있어야 한다. 이는 시장에서 이용가능한 모든 실제 클라이언트의 웹브라우저(3)의 표준 성향이다.

[0056] 이와 달리, 문서(13)의 포맷은 서버에 의해 리턴되고 HTML 문서(13)가 필요할 필요가 없는 클라이언트의 웹브라우저(3)에 디스플레이된다. 클라이언트의 웹브라우저(3)에 의해 디스플레이될 수 있고 렌더링 동안 웹브라우저(3)에 의해 해석되는 스크립트 요소를 포함할 수 있는 모든 문서 유형이 사용될 수 있다. 사용될 수 있는 문서의 예는 XML, XHTML, SVG(스케일링 가능한 벡터 그래픽), XUL(XML 사용자 인터페이스 언어) 등 (특히 모든 종류의 XML 문서)이다. 문서에 내장된 스크립트(12)는 자바스크립트가 아닌 다른 언어로 기록될 수 있다. 클라이언트의 웹브라우저(3) 내의 자바스크립트(12) (또는 표준화 버전 ECMA 스크립트)의 지원은 표준이다. 그러나, 웹브라우저(3)에 의해 렌더링되도록 문서(13)에 내장될 수 있고, 문서(13) 내의 다른 요소 또는 클라이언트의 웹브라우저(3)나 사용자 환경에 대한 정보를 검색하도록 클라이언트의 웹브라우저(3) 내에서 실행될 수 있는 임의의 종류의 스크립트 또는 플러그인이 사용될 수 있다. 예는 비주얼 베이직 스크립트, 액티브엑스 구성요소, 자바 애플릿 또는 고유 클라이언트의 웹브라우저(3) 플러그인일 수 있다. HTTP 프로토콜은 클라이언트의 웹브라우저(3)에 의해 서블릿(8)으로부터 정보를 검색하라는 요청을 개시하고, 서블릿(8)에 의해 적어도 그 요청의 성공 또는 실패를 명시하는 리턴 코드 및 (이용가능하면) 그 요청의 결과를 포함하는 응답을 전달하는 단방향 통신 경로를 특징으로 하는 다른 통신 프로토콜로 대체될 수 있다.

[0057] 서버 측(2)은 바람직하게 서블릿 컨테이너(6)를 가지는 애플리케이션 서버(4)(예컨대, 웹스피어(Websphere)와 같은 J2EE 서버 또는 톰캣(Tomcat))인 것이 바람직하다. 본 발명의 바람직한 실시예에서 2 개의 서블릿 -주 서블릿(8) 및 부 서블릿(10)- 은 이러한 애플리케이션 서버(4)에 설치된다.

[0058] 애플리케이션 서버(4)는 J2EE 서버일 필요가 없다. 클라이언트의 웹브라우저(3)로부터 발생하는 HTTP 요청에 동적으로 응답할 수 있는 임의의 종류의 서버가 사용될 수 있다. J2EE 애플리케이션 서버를 대신하여 사용될 수 있는 기술의 예는 마이크로소프트 .NET 또는 동적 콘텐츠를 디스플레이할 수 있는 CGI 스크립트를 실행할 수 있는 간단한 웹서버이다. 서블릿(8,10) 및 서블릿 컨테이너(6)는 ASP, PHP 또는 CGI 스크립트와 같은 유사한 기술로 대체될 수 있다.

[0059] 본 발명이 본 발명의 바람직한 실시예로 구현되는 경우에 다음과 같이 동작한다(도 2a의 화살표 참조).

[0060] 클라이언트 측(1)에서 클라이언트의 웹브라우저(3)는 주 서블릿에 요청을 전달한다 ①. 주 서블릿(8)은 초기 요청에 응답하여 클라이언트의 웹브라우저(3)에 전달하도록 문서를 생성하는 데 필요한 로직을 구현하는 서블릿이다. 주 서블릿(80)은 요청을 수신하고, 그 요청과 함께 전달된 최종 파라미터를 판독하며, 클라이언트의 웹브라우저(3)에 리턴하는 문서(13)의 생성을 시작한다. 요청과 함께 전달된 파라미터는 문서(13)의 생성에 영향을 줄 수 있다. 본 발명은 응답 문서(13)를 생성하기 위해 주 서블릿(8)에 의해 사용된 로직에 의존하지 않는다.

[0061] 응답의 생성 동안 어떤 시점에서, 주 서블릿(8)은 (파라미터와 같이) 요청시에 클라이언트의 웹브라우저(3)에 의해 제공되지 않았지만 클라이언트의 웹브라우저(3) 내의 클라이언트 측(1) 상의 자바스크립트(12)를 실행함으로써 쉽게 검색될 수 있었던 다른 정보를 필요로 한다.

[0062] 주 서블릿(8)은 생성하기 시작된 문서(13)에 자바스크립트(12)를 삽입한다. 이러한 스크립트(12)는 클라이언트의 웹브라우저(3)에 의해 실행되는 경우에, 누락 정보를 검색하고, HTTP 요청을 부 서블릿(10)에 전달하는데, 그 요청은 주 서블릿(8)에 의해 생성된 키를 파라미터로서 포함한다(키는 임의로 생성된 임의의 고유한 스트링 및 이전 단계에서 검색된 값일 수 있다).

[0063] 일단 스크립트(12)가 부분적인 응답에 생성되고 삽입되었으면, 주 서블릿(8)은 버퍼를 플러시(flush)하여, 이 시점까지 생성되어온 응답 문서(13)의 일부가 클라이언트의 웹 브라우저에 즉시 전송되게 한다 ②. 웹브라우저(3)로의 통신 채널은 폐쇄되지 않는다. 클라이언트의 웹브라우저(3)에 전달된 HTTP 응답의 헤더는 HTTP 1.1에서 정의된 바와 같이 그 응답이 "청크 전송 코딩(chunked Transfer Coding)"을 사용함을 나타낸다. 이러한 인코딩은 응답이 하나의 단일 조각이 아니라 일련의 청크로 발생할 것이며, 서버가 접속할 때까지 클라이언트의 웹브라우저(3)가 새로운 콘텐츠를 대기해야 함을 의미한다.

[0064] 일단 부분적으로 생성된 문서(13)(문서의 개시를 포함하는 제 1 청크 및 누락 정보를 검색하는 자바스크립트

(12))가 클라이언트의 웹브라우저(3)에 전달되면, 클라이언트의 웹브라우저(3)는 지금까지 수신되었던 문서(13)의 일부를 렌더링하기 시작하고, 문서의 나머지를 대기하는 동안 내장된 스크립트(12)를 실행한다.

[0065] 이러한 스크립트(12)의 실행은 누락 값을 검색하고 부 서블릿(10)에 다른 HTTP 요청을 전달한다. 이러한 요청은 키 및 이전 단계 ③에서 검색된 값을 파라미터로서 포함할 수 있다.

[0066] 1차 HTTP 요청의 응답이 로딩되는 동안 상이한 URL에 다른 HTTP 요청을 전달하는 능력은 웹브라우저(3)의 전형적인 성향이다. 이러한 특성이 없으면, 클라이언트의 웹브라우저(3)는 예컨대, 전체 문서가 완전히 로딩되기 전에 긴 문서가 로딩되기 시작할 때 포함된 영상을 디스플레이할 수 없을 것이다.

[0067] 부 서블릿(10)은 자바스크립트(12)에 의해 전달된 HTTP 요청을 수신하고, 요청의 파라미터에 포함된 키 및 값을 디코딩하며, 이러한 한 쌍의 키 값을 현재의 HTTP 세션 객체에 저장한다 ④. HTTP 세션은 클라이언트(1)와 서버(2) 사이의 현재의 세션에 대한 정보를 저장하는 객체이다. 세션 객체는 요청을 수신할 때마다 서블릿에 전달된다. 만일 동일한 클라이언트의 웹브라우저(3)가 동일한 서블릿 컨테이너(6) 및 동일한 웹 애플리케이션에 포함된 몇몇의 서블릿(8)에 HTTP 요청을 전달하면, 다른 서블릿(8)에 동일한 세션 객체가 전달될 것이다. 서블릿(8)은 세션 객체 내의 기록 값을 판독할 수 있다(값은 나중에 그 값을 검색하기 위해 사용될 수 있는 키 하에 저장됨). HTTP 세션을 사용하는 것은 서로 다른 서블릿(8) 사이에서 값을 전달하는 J2EE의 방법이다.

[0068] 부 서블릿(10)을 사용하는 대신에, 자바스크립트(12)는 주 서블릿(8)에 HTTP 세션을 갱신하라는 요청을 전달할 수 있다. 그러나 이러한 경우에 (여전히 1차 요청에 대한 응답을 생성하고 있으며 누락 값을 대기하는 제 1 인스턴스와 병렬로) 주 서블릿(8)의 제 2 인스턴스가 시작할 것이다. 이어서 요청을 수신하는 경우 주 서블릿(8)은 그 요청이 문서를 요청하는지 여부 또는 그 요청이 문서 요청(1차 요청)을 처리하는 서블릿의 다른 인스턴스에 의해 요구되었던 다른 정보를 제공하는지 여부를 테스트할 필요가 있을 것이다. 기술적으로 이는 요청과 함께 전달된 파라미터를 테스트함으로써 처리될 것이다. 만일 특정 파라미터가 그 요청에 포함되면, 주 서블릿(8)은 요청의 다른 파라미터에 포함된 한 쌍의 키/값으로 HTTP 세션을 갱신해야 함을 알 것이다. 기타 경우에, 주 서블릿(8)은 응답 문서(13)를 생성해야 한다. 이를 수행함으로써, 주 서블릿(8)은 요청의 파라미터에 의존하는 2 개의 상이한 로직을 가질 것이다. 이는 각 서블릿이 단일 로직을 구현하는 2 개의 상이한 서블릿을 가지는 것과 정확히 같다.

[0069] 부 서블릿(10)은 HTTP 세션에서 2차 요청(자바스크립트에 의해 전달된 요청)에 의해 전달된 다른 값을 저장할 필요가 없는데, 이는 그 값이 주 서블릿(8)에 의해 알려지게 하기 위해서이다. 다른 방법은 서블릿 컨테이너(6)의 자바 가상 머신의 시스템 특성에 값을 저장하는 것 또는 (애플리케이션 서버에 의해 사용된 기술에 따른) 다른 공유 메모리 메커니즘을 사용하는 것일 수 있다. 다른 가능성은 부 서블릿(10)이 서버의 로컬 디스크 상의 파일에 이러한 값을 저장하게 하고 주 서블릿(8)이 동일한 파일을 판독하게 하는 것이다.

[0070] 일단 (HTTP 세션에서) 요청된 값이 공유 메모리 내의 주 서블릿(8)에 의해 발견되면, 주 서블릿(8)은 구동되어 문서(13)의 생성을 계속한다. 누락 정보를 사용하여, 이제 문서는 완전히 생성될 수 있고 문서의 나머지는 브라우저에 전달된다 ⑤ -마지막 단계 동안 주 서블릿(8)은 HTTP 세션의 콘텐츠를 정기적으로 체크하여 그 값이 이용가능했었는지 여부를 확인한다. 주 서블릿(8)이 세션을 체크하여 그 값이 이용가능하지 않았던 시간마다, 주 서블릿(8)은 짧은 시간 동안 대기 상태였고 다시 체크하였다.

[0071] 일단 서블릿이 웹브라우저(3)와 접속하면(이는 서블릿의 doGet(...) 또는 doPost(...) 방법이 리턴하는 경우에 자동으로 처리된다), 클라이언트의 웹브라우저는 전체 문서가 다운로드되었음을 인식한다.

[0072] 서블릿(8)은 요청을 처리하기 위해 클라이언트의 웹브라우저로부터 하나 이상의 다른 정보를 필요로 할 수 있다. 다수의 누락 정보의 검색은 요청을 처리하는 동안 본 명세서에 설명된 단계를 몇 번 반복함으로써 또는 하나의 HTTP 요청시에 부 서블릿(10)에 대해 모든 누락 정보를 검색하고 전송하는 하나의 자바스크립트(12)를 생성함으로써 처리될 수 있다. 이어서 부 서블릿(10)은 자바스크립트에 의해 전달된 2차 요청으로부터 몇몇 한 쌍의 키 값을 디코딩하고 이에 따라 HTTP 세션을 갱신해야 한다.

[0073] 이러한 방법을 사용함으로써, 서블릿은 클라이언트의 웹브라우저(3)가 초기 요청을 보낸 후에 및 요청을 처리하는 동안 클라이언트의 웹브라우저(3)로부터 다른 정보를 요청할 수 있었다. 그 요청의 처리는 본 발명을 이용하지 않는 것처럼 애초부터 다시 중지 및 시작될 필요가 없었다.

[0074] 웹브라우저(3) 및 애플리케이션 서버(4)에 어떠한 변경도 필요하지 않다. 주 서블릿(8)과 부 서블릿(10) 양자 모두 HTTP 프로토콜에 규정된 바와 같이, HTTP 요청을 수신하고 그 요청의 발신자에 하나의 단일 응답을 리턴하는 별개의 전형적인 서블릿이다. 프로토콜은 변경되지 않는다. 클라이언트의 웹브라우저(3)와 서버(2) 사이의

모든 통신은 (HTTP 프로토콜에 의해 예상되는 바와 같이) 클라이언트의 웹브라우저에 의해 구동된다.

[0075] 2 개의 서블릿(8,10) 간의 협력 및 클라이언트의 웹브라우저(3)의 응답 문서와 이에 내장된 자바스크립트 명령을 청킹함으로써 서블릿이 클라이언트의 웹브라우저(3)를 원격 제어할 수 있다는 사실만이 서버와 클라이언트의 웹브라우저 사이의 양방향 통신을 가능하게 한다.

[0076] 후속하는 간단한 예는 본 발명이 실제로 작용하는 방법을 도시한다(도 2b 참조).

[0077] 클라이언트의 웹브라우저(3)로부터의 요청을 수신하고, 요청을 전달하였던 클라이언트의 웹브라우저(3)의 몇몇 특성에 의존하는 방식으로 몇몇 콘텐츠를 생성하는 주 서블릿(8)이 존재한다고 가정한다. 주 서블릿(8)이 요청을 전달하였던 클라이언트의 웹브라우저 윈도우의 폭을 알 필요가 있고, 이러한 정보가 주 서블릿(8)에 전달된 초기 요청에 포함되지 않았다고 더 가정한다.

[0078] 클라이언트의 웹브라우저 윈도우의 폭은 클라이언트의 웹브라우저(3) 내의 자바스크립트 표현 "window.outerWidth"를 실행함으로써 쉽게 요청될 수 있는 특성이다.

[0079] 이러한 예에 대한 주 서블릿(8) 및 부 서블릿(10)의 코드 소스는 본 문서의 마지막에 있는 부록에 존재한다. 부 서블릿이 1차 서블릿의 로직과 무관하며 상이한 로직을 구현하는 몇몇 "주" 서블릿에 의해 재사용 또는 공유될 수 있음을 알아야 한다.

[0080] 주 서블릿에 대한 요청이 여전히 구동하는 동안 이러한 2차 요청의 결과는 클라이언트의 웹브라우저에 의해 디스플레이된 현재의 문서를 대체해서는 안 되며 1차 요청을 인터럽트해서도 안 된다는 것이 중요하다. 이는 2차 요청의 결과를 개별 프레임 또는 페이지의 특정 요소(영상, 내장된 프레임)로 지시방향을 변경하거나, 자바스크립트 API를 사용하여 HTTP 요청을 프로그램적으로 개방함으로써 달성될 수 있다.

[0081] 부 서블릿(10)은 세션 (또는 주 서블릿과 공유된 메모리)에 저장하기 위해 파라미터에 정보를 가진 한 쌍의 키/값을 포함하라는 요청을 수신한다. 이어서 부 서블릿(10)은 웹브라우저(3)에 성공 코드를 리턴한다. 웹브라우저(3)는 부 서블릿(10)의 응답을 무시할 수 있다.

[0082] 주 서블릿(8)은 생성되었던 키에 대해 저장된 값을 세션(공유 메모리)에서 검출한다. 주 서블릿(8)은 (요청을 처리하는 데 있어서 누락되었던 정보인) 이 값을 추출하고, 세션/공유 메모리로부터 그 값을 제거하며 요청의 처리를 다시 시작한다.

[0083] 처리는 응답 문서의 나머지가 생성되는 것을 계속한다. 요청을 처리하는 동안 더 많은 정보가 나중에 누락되고 있음을 나타내면, 이어서 사용자에게 많은 정보를 요청하도록 마지막 단계가 반복될 수 있다.

[0084] 마지막으로 요청이 완료되고 클라이언트의 웹브라우저(3)가 완전한 응답 문서를 수신 및 디스플레이한다.

[0085] 도 2b는 본 발명에 따른 이러한 예에서 클라이언트의 웹브라우저와 2 개의 서블릿 사이에서 발생하는 동작 및 통신을 설명한다.

[0086] (1) 클라이언트의 웹브라우저는 서버(2)에 HTTP GET 명령(이와 달리, POST 명령도 사용될 수 있음)을 전달함으로써 주 서블릿(8)의 URL을 요청하는 초기 요청을 주 서블릿(8)에 전달한다. 이러한 요청은 응답 문서를 생성하는 동안 주 서블릿(8)에 의해 사용될 수 있는 다른 파라미터도 포함할 수 있다. 이 예에서, 파라미터는 존재하지 않는다.

[0087] (2) 주 서블릿(8)은 누락 정보(클라이언트의 웹브라우저의 폭)를 필요로 하는 시점까지 문서의 개시를 기록함으로써 응답 문서의 생성을 시작한다.

[0088] <html>

[0089] <head>

[0090] </head>

[0091] <body>

[0092] 몇몇 콘텐츠는 누락 값이 요청되기 전에 서블릿에 의해 산출되었다 <p>.

[0093] 이 시점에서, 문서의 나머지를 기록하기 위해, 주 서블릿(8)은 누락 정보를 필요로 한다. 주 서블릿(8)은 고유한 키 -클라이언트의 웹브라우저가 부 서블릿(10)에 그 고유한 키 전달한 후에 HTTP 세션에서 누락 값을 저장하고 발견하는 데 사용될 것임- 를 생성하고, 지금까지 생성된 콘텐츠에 클라이언트의 웹브라우저에 의해 실행되

는 경우에 값을 검색하고 키와 함께 그 값을 부 서블릿(10)에 전달할 자바스크립트를 첨부한다.

[0094] 이 예에서 스크립트를 어떻게 나타내는지를 도시하고 있다. 라인 2는 웹브라우저(3) 윈도우의 폭을 검색하고, 다음 라인들은 웹브라우저(3)가 부 서블릿(10)에 키(여기서 임의의 키 값은 key_112108임) 및 검색된 값(여기서 가변 "값"에 포함됨)을 파라미터로서 전달하는 HTTP 요청을 개방하게 한다.

[0095] <script language="JavaScript">

[0096] var value = window.outerWidth;

[0097] var httpReq = new XMLHttpRequest();

[0098] httpReq.open("GET","SecondaryServlet?key=key_112108&value="+value,true);

[0099] httpReq.send(null);

[0100] </script>

[0101] (3) 부분적인 문서(지금까지 생성된 콘텐츠 및 누락 값을 검색하는 스크립트)는 주 서블릿(8)의 버퍼를 플러시함으로써 클라이언트의 웹브라우저(3)에 전달된다. 이를 실행하는 경우에, 애플리케이션 서버는 HTTP 응답의 인코딩을 콘텐츠가 몇몇 청크 조각으로 전달될 클라이언트의 웹브라우저(3)를 나타내는 "청크"로 변경한다. 도 2는 이 시점에서 클라이언트의 웹브라우저에 전달된 문서의 콘텐츠를 도시한다.

[0102] (4) 부 서블릿(8)은 HTTP 세션에서 단계 (2)에서 생성하였던 키 하에 저장된 값을 발견할 때까지 실행을 중지한다. 기술적으로 주 서블릿(8)은 실행을 중지하고 값이 이용가능한지 여부를 정기적으로 체크한다. 일단 값이 발견되면, 부 서블릿(8)은 실행을 다시 시작한다.

[0103] (5) 한편 클라이언트의 웹브라우저(3)는 주 서블릿(8)에 의해 전달된 문서의 제 1 청크를 수신한다. 클라이언트의 웹브라우저(3)는 지금까지 문서에 포함된 디스플레이 가능한 콘텐츠를 디스플레이하고 문서에 내장된 자바스크립트를 실행한다. 자바스크립트가 문서의 콘텐츠에 전혀 영향을 주지 않음을 알아야 한다.

[0104] (6) 자바스크립트를 실행하는 경우에, 클라이언트의 웹브라우저(3)는 윈도우의 폭(여기서 454 픽셀)을 검색하고 HTTP GET 요청을 부 서블릿(10)에 전달한다. (이와 달리, POST 요청도 사용될 수 있다). 이러한 요청은 이전에 생성된 키(key_112108) 및 검색된 값(454)을 파라미터로서 포함한다.

[0105] GET SecondaryServlet?key=key_112108&value=454,

[0106] (7) 부 서블릿(10)은 이러한 요청을 수신하고, 파라미터의 범위를 벗어난 키 및 값을 디코딩하며 HTTP 세션 내에 키(key_112108) 하의 값(454)을 배치한다. 이어서 부 서블릿(10)은 성공 코드(HTTP 코드 200)(8)를 즉시 리턴한다.

[0107] (8) 키 "key_112108" 하에 저장된 값에 대한 HTTP 세션을 정기적으로 체크하였던 주 서블릿(8)은 이러한 값을 찾는다. 주 서블릿(8)은 세션으로부터 그 값을 제거하고 검색되었던 값을 사용하여 문서의 생성을 다시 시작한다. (이러한 예에서, 값은 텍스트로 간단히 기록된다. 보다 복잡한 예에서, 이는 보다 복잡한 문서의 레이아웃을 처리하는 데 사용될 수 있다.)

[0108] 클라이언트의 웹브라우저(3)에 전달된 문서의 제 2 청크 부분은 다음과 같다.

[0109] Finish the generation of the page: Width=454

[0110] </body>

[0111] </html>

[0112] (9) 문서의 제 2 청크 부분은 클라이언트의 웹브라우저에 전달되고, 주 서블릿(8)은 클라이언트의 웹브라우저(3)와의 통신 채널을 폐쇄하는데, 이는 더 이상 후속하는 콘텐츠가 없음을 나타낸다.

[0113] (10) 클라이언트의 웹브라우저(3)는 문서의 나머지를 디스플레이한다.

[0114] 보다 상세한 실행도는 도 1을 참조한다.

[0115] 이러한 예의 완전한 소스 코드는 본 문서의 마지막에 있는 부록을 참조한다.

[0116] <부록>

```
  Appendix:

  Source 1: Main Servlet

  import java.io.*;
  import javax.Servlet.*;
  import javax.Servlet.http.*;

  public class MainServlet extends HttpServlet implements Servlet {

/**
 * URL of the secondary Servlet storing the requested key/value pairs
 * in the HTTP session
 */
private final static String SECONDARY_SERVLET_URL = "SecondaryServlet";

/**
* @see javax.Servlet.http.HttpServlet#void
  (javax.Servlet.http.HttpServletRequest,
   javax.Servlet.http.HttpServletResponse)
*/
public void doGet(HttpServletRequest req, HttpServletResponse resp)
  throws ServletException, IOException {
  resp.setContentType("text/html; charset=ISO-8859-4");
  HttpSession session = req.getSession();
  PrintWriter out = resp.getWriter();
  out.println("<html>");
  out.println("  <head>");
  out.println("  </head>");
  out.println("  <body>");

  // Starts the generation of the response
  out.println(
    "  Some content produced by the Servlet before the missing value is asked<p>");

  // (...)
  // at some point during the processing of the answer,
  // the Servlet notices that some information are missing
```

[0117]

```
    // Suspends the generation of the response.

    // Asks the client's web-browser's to retrieve the missing value and send it to
    // the server through the secondary Servlet
    // The 3rd parameter is the JavaScript expression retrieving
    // the value when executed in the browser.
    // In this example we ask the client's web-browser for the width of the
    browser window
    Object value = requestValue(req, resp, "window.outerWidth;");

    // When this method returns, the local variable "value" contains the
    requested value

    // Continue the generation of the response
    // (...)
    out.println("Finish the generation of the page: Width=" + value);

    out.println("  </body>");
    out.println("</html>");
  }

  /**
   * Generates a unique key that can be used to store the missing value in
    the Web session
   */
  private String generateKey() {
    return "key_" + System.currentTimeMillis();
  }

  /**
   * Requests the client's web-browser to provide the Servlet a missing value.
   * This methods generates and transmit a JavaScript to be interpreted
   * by the client's web-browser. This JavaScript retrieves the missing value
   * by executing a JavaScript expression given as parameter,
   * and sends a request to the secondary Servlet that will store
   * the retrieved value in the HTTP session, to that it is accessible
   * for the main Servlet.
   * Note that this method will block and suspend the main Servlet, until
[0118]  * the missing value has been returned by the client's web-browser.
```

```
 *
 * @param req The main HTTP request sent by the client's web-browser to this main
   Servlet
 * @param resp The HTTP response used by this Servlet to respond to the request
 * @param JavaScriptExpression The JavaScript expression, which when executed
 *  in the client's web-browser, returns the missing information needed by this Servlet
 * @return The missing value returned by the client's web-browser.
 */
private Object requestValue(
  HttpServletRequest req,
  HttpServletResponse resp,
  String JavaScriptExpression)
  throws IOException {
  PrintWriter out = resp.getWriter();
  HttpSession session = req.getSession();

  // Generates a unique key, that will be used to store/retrieve the
  // missing value in the current HTTP session
  String key = generateKey();

  // Starts the generation of a JavaScript that will retrieve
  // the information from the client's web-browser
  out.println("    <script language=\"JavaScript\">");
  // In this example the client's web-browser will prompt the user to enter
  the missing value
  out.println("var value = " + JavaScriptExpression + ";");
  // once the value has been retrieved on stored on the client side
  // in a JavaScript variable, transmit this value to the server through
  // the secondary Servlet

  // Makes the client's web-browser open an HTTP request to the secondary Servlet,
  // passing the key and value of the missing information
  out.println("var httpReq = new XMLHttpRequest();");
  out.println(
    "httpReq.open(\"GET\",\""
      + SECONDARY_SERVLET_URL
      + "?key="
      + key
      + "&value=\"+value,true);");
[0119]  out.println("httpReq.send(null);");
```

```
out.println("    </script>");
// Flushes the buffer of the Servlet, so that the JavaScript retrieving the
// missing value is transmitted and executed immediately by the client's
web-browser
resp.flushBuffer();

// Suspends the execution of the main Servlet until a value can be found
// in the HTTP session under the key defined previously
Object value = session.getAttribute(key);
while (value == null) {
  try {
    Thread.sleep(100);
  } catch (InterruptedException e) {
    e.printStackTrace();
  }
  value = session.getAttribute(key);
}

// Now that the value has been extracted, remove its key/value pair from
// the session, to avoid memory leaks and allow the same key to be reused
// several times
session.removeAttribute(key);

return value;
  }
```

[0120] }

Source 2: Secondary servlet

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

/**
 * Secondary Servlet updating the HTTP session with key/value pairs.
 * The key/value pairs to store are contained in the incoming requests
 * under the keys "key" and "value".
 * For example a request containing the parameters
 * "mySecondaryServletURL?key=myKey&value=myValue"
 * will store the value "myValue" under the key "myKey" in the HTTP session.
 */
public class SecondaryServlet extends HttpServlet implements Servlet {

 /**
 * @see javax.servlet.http.HttpServlet#void
(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse)
 */
 public void doGet(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {
      // Extracts from the request coming from the browser the key/value pair
      // to store in the HTTP session
      String key = req.getParameter("key");
      String value = req.getParameter("value");
      if (key != null) {
            // Store this key/value pair in the HTTP session
            req.getSession().setAttribute(key, value);
            }
  }
```

[0121] }

도면의 간단한 설명

[0051] 도 1은 본 발명의 양방향 통신 경로를 도시한다.

[0052] 도 2a는 본 발명의 양방향 통신 경로의 바람직한 실시예의 구현을 가진 클라이언트-서버 아키텍처를 도시한다.

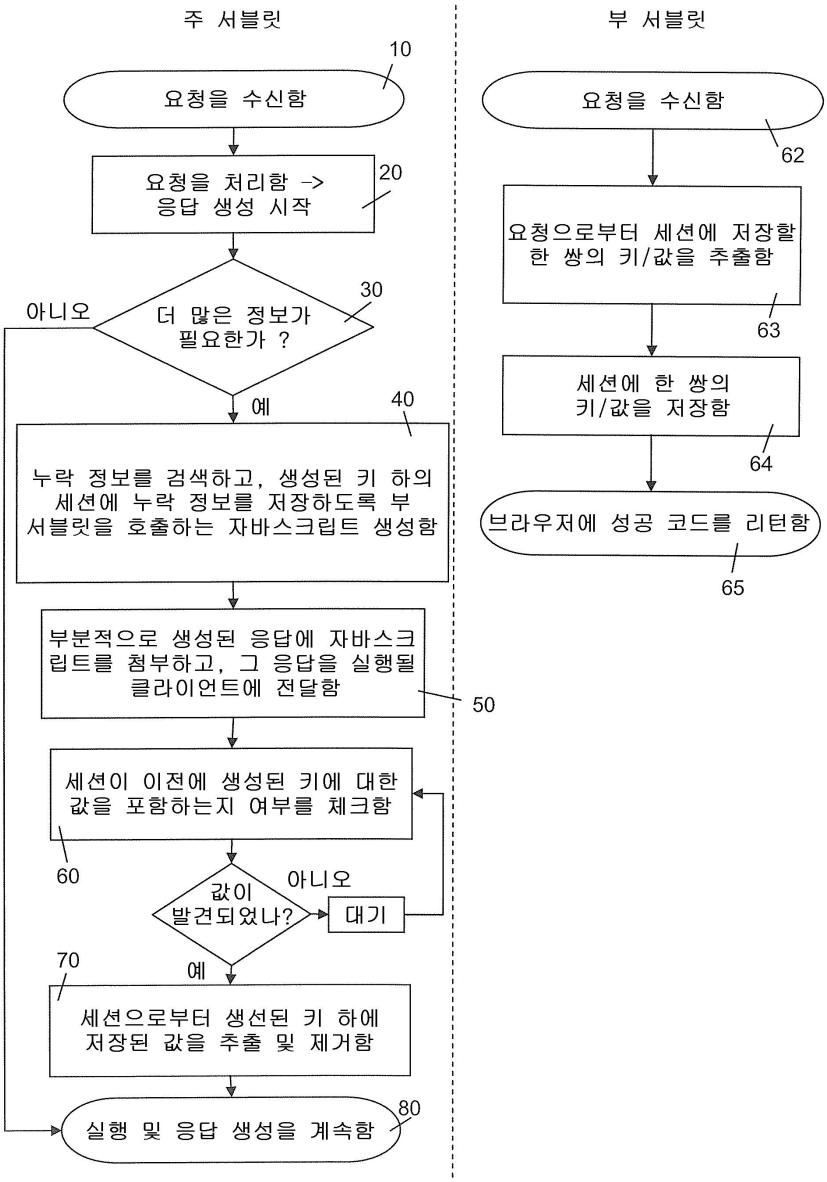
[0053] 도 2b는 주어진 예에서 클라이언트의 웹브라우저, 주 서블릿 및 부 서블릿 사이의 통신 단계를 설명하는 순서도

를 도시한다.

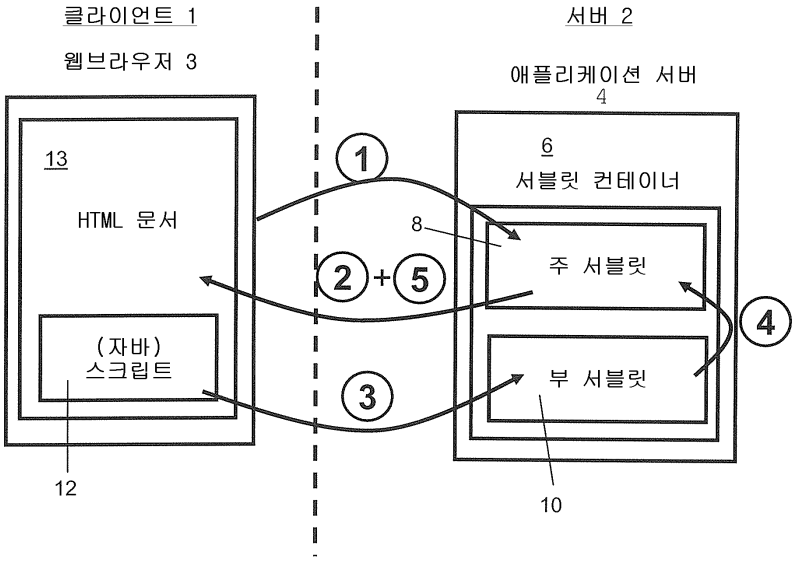
[0054] 도 2a는 클라이언트-서버 아키텍처에서 본 발명의 바람직한 실시예를 도시한다.

도면

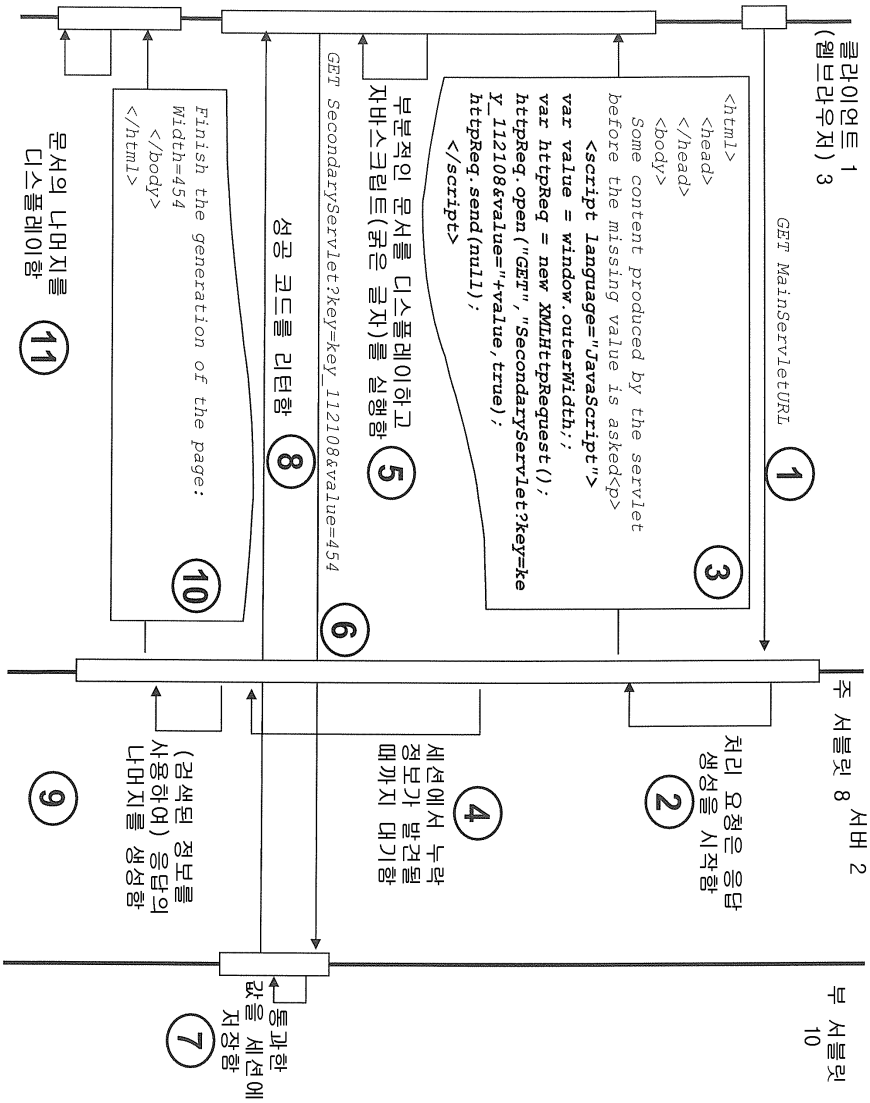
도면1



도면2a


클라이언트 1
웹브라우저 3
서버 2
애플리케이션 서버
4
13
HTML 문서
6
서블릿 컨테이너
8
주 서블릿
부 서블릿
(자바)
스크립트
12
10
1
2+5
3
4

도면2b


클라이언트 1
(웹브라우저) 3
GET MainServletURL
1
<html>
<head>
</head>
<body>
Some content produced by the servlet
before the missing value is asked<p>
<script language="JavaScript">
var value = window.outerWidth;
var httpReq = new XMLHttpRequest();
httpReq.open("GET","SecondaryServlet?key=ke
y_112108&value="+value,true);
httpReq.send(null);
</script>
3
부분적인 문서를 디스플레이하고
자바스크립트(굵은 글자)를 실행함
5
GET SecondaryServlet?key=key_112108&value=454
6
성공 코드를 리턴함
8
Finish the generation of the page:
Width=454
</body>
</html>
10
문서의 나머지를
디스플레이함
11
주 서블릿 8
서버 2
처리 요청은 응답
생성을 시작함
2
세션에서 누락
정보가 발견될
때까지 대기함
4
(검색된 정보를
사용하여) 응답의
나머지를 생성함
9
값을 통과함
세션에
저장함
7
부 서블릿
10