(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2011년08월24일
(11) 등록번호 10-1058481
(24) 등록일자 2011년08월16일

(51) Int. Cl.
G06F 3/048 (2006.01) *G06F 9/00* (2006.01)
G06F 15/02 (2006.01) *H04W 88/02* (2009.01)
(21) 출원번호 10-2008-7006330
(22) 출원일자(국제출원일자) 2007년05월15일
심사청구일자 2008년03월14일
(85) 번역문제출일자 2008년03월14일
(65) 공개번호 10-2008-0037089
(43) 공개일자 2008년04월29일
(86) 국제출원번호 PCT/CA2007/000864
(87) 국제공개번호 WO 2007/131359
국제공개일자 2007년11월22일
(30) 우선권주장
60/800,416 2006년05월16일 미국(US)
(56) 선행기술조사문헌
US20010030667 A1
US20040111673 A1
WO2003017077 A2
전체 청구항 수 : 총 21 항

(73) 특허권자
리서치 인 모션 리미티드
캐나다 온타리오 워털루 필립 스트리트 295 (우편번호 엔2엘 3더블유8)
(72) 발명자
레이시 존 데이빗 케네쓰
캐나다 온타리오주 엘4에스 1엑스4 리치몬드 힐 러싱브룩드라이브 84
포밋체프 미하일
캐나다 온타리오주 엘4제이 8제트3 쏜힐 어텀 힐 블러바드 152
(뒷면에 계속)
(74) 대리인
신정건, 김태훈

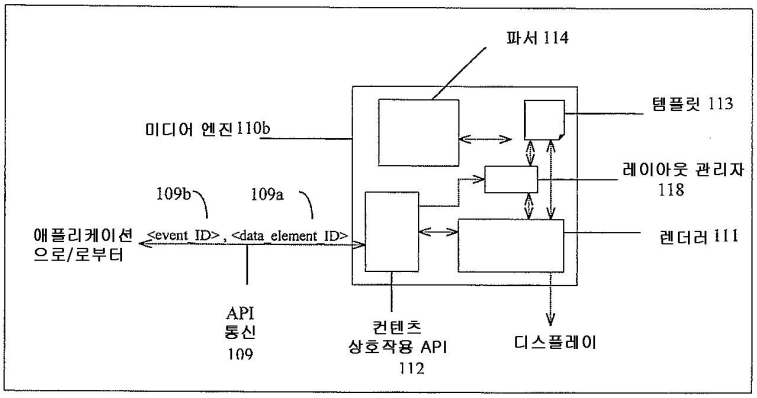
심사관 : 정재우

(54) 애플리케이션의 사용자 인터페이스를 스킨화하는 시스템 및방법

(57) 요 약

장치 상에 애플리케이션에 대한 그래픽 인터페이스를 생성하는 미디어 엔진이 개시되어 있다. 이 미디어 엔진은 애플리케이션에 대한 변경을 미디어 엔진에 통지하는 상호작용 인터페이스, 그래픽 인터페이스를 렌더링하는 렌더러, 렌더러가 그래픽 인터페이스를 렌더링하는 방법을 제어하기 위한 템플릿 정보를 템플릿 파일로부터 파싱하는 파서, 및 템플릿 정보의 디스플레이를 선택적으로 제어하는 레이아웃 관리자를 포함한다.

대 표 도 - 도3b



(72) 발명자

로하스 호세 호세

캐나다 온타리오주 엠6케이 3엔1 토론토 킹 스트리트 웨스트 1000

가루드 스튜어트

캐나다 온타리오주 엘6엑스 1엠1 브램톤 아치볼드 스트리트 17

특허청구의 범위

청구항 1

프로세서에 의해 실행되며 장치상에서의 디스플레이를 위한 애플리케이션에 대한 그래픽 인터페이스를 생성하는 미디어 엔진(media engine)을 포함하는 컴퓨터 판독가능한 기록 매체로서, 상기 미디어 엔진은,

상기 애플리케이션에 대한 변경들을 상기 미디어 엔진에 통지하는 상호작용 인터페이스(interaction interface),

상기 디스플레이 상에 상기 그래픽 인터페이스를 렌더링하는 렌더러(renderer),

상기 렌더러가 상기 그래픽 인터페이스를 렌더링하는 방법을 제어하기 위한 템플릿 정보를 템플릿 파일로부터 파싱하는 파서(parser), 및

템플릿 정보의 디스플레이를 제어하는 레이아웃 관리자(layout manager)를 포함하는 것인, 컴퓨터 판독가능한 기록 매체.

청구항 2

제1항에 있어서, 상기 템플릿 정보는,

데이터 요소 객체(data element objects)의 세트, 및

커스텀 이벤트 객체(custom event objects)의 세트

를 포함하는 것인, 컴퓨터 판독가능한 기록 매체.

청구항 3

제2항에 있어서, 상기 템플릿 정보는,

상기 레이아웃 관리자가 상기 템플릿 정보의 디스플레이를 제어하는 방법을 제어하기 위한 레이아웃 관리자 제어 정보와,

렌더러 제어 정보 중 적어도 하나를 더 포함하고,

상기 렌더러 제어 정보는,

상기 렌더러가 상기 데이터 요소 객체의 세트를 렌더링하는 방법, 및

상기 커스텀 이벤트 객체의 세트 중 적어도 하나의 커스텀 이벤트 객체에 기초하여 상기 렌더러가 상기 그래픽 인터페이스를 렌더링하는 방법

을 제어하기 위한 것인, 컴퓨터 판독가능한 기록 매체.

청구항 4

제3항에 있어서, 상기 레이아웃 관리자 제어 정보는,

렌더링되기 이전에 상기 데이터 요소 객체의 세트 중 적어도 하나의 데이터 요소 객체의 길이를 동적으로 수정하기 위한 정보, 및

상기 렌더러가 렌더링할 수 없는 상기 그래픽 인터페이스에 대한 템플릿 정보의 디스플레이를 제어하기 위한 정보

중 적어도 하나를 포함하는 것인, 컴퓨터 판독가능한 기록 매체.

청구항 5

제1항에 있어서, 상기 미디어 엔진은 상기 애플리케이션에 이벤트 통지를 제공하는 이벤트 엔진을 더 포함하는, 컴퓨터 판독가능한 기록 매체.

청구항 6

제1항에 있어서, 상기 상호작용 인터페이스는 컨텐츠 상호작용 애플리케이션 프로그래밍 인터페이스(API)를 포함하며,

상기 API는,

데이터 요소를 갱신하는 것,

커스텀 이벤트를 통지하는 것,

새로운 이벤트를 문의하는 것,

새로운 이벤트를 가져오는 것, 및

이벤트 리스너(event listener)를 등록하는 것

을 위한 호출들(calls)을 포함하는 것인, 컴퓨터 판독가능한 기록 매체.

청구항 7

제3항에 있어서, 상기 템플릿 파일은,

상기 데이터 요소 객체의 세트,

상기 커스텀 이벤트 객체의 세트,

상기 레이아웃 관리자 제어 정보, 및

상기 렌더러 제어 정보의

마크업 언어 기술(markup language description)을 포함하는 것인, 컴퓨터 판독가능한 기록 매체.

청구항 8

모바일 통신 장치로서,

디스플레이,

프로세서상에서 실행되며 상기 디스플레이 상에 그래픽 인터페이스를 생성하는 오퍼레이팅 시스템,

상기 그래픽 인터페이스로 사용자 상호작용을 가능하도록 하는 입력 장치,

상기 오퍼레이팅 시스템에 의해 상기 프로세서상에서 실행되는 적어도 하나의 애플리케이션,

상기 오퍼레이팅 시스템에 의해 액세스할 수 있는 메모리에 저장된 적어도 하나의 템플릿 파일, 및

상기 프로세서에 의해 실행되는 미디어 엔진을 포함하고,

상기 미디어 엔진은,

상기 적어도 하나의 애플리케이션에 대한 변경들을 상기 미디어 엔진에 통지하는 상호작용 인터페이스;

상기 디스플레이 상에서 상기 그래픽 인터페이스를 렌더링하는 렌더러;

상기 렌더러가 상기 디스플레이 상에서 상기 그래픽 인터페이스를 렌더링하는 방법을 제어하기 위해 적어도 하나의 템플릿 파일로부터 템플릿 정보를 파싱하는 파서; 및

상기 템플릿 정보의 디스플레이를 제어하는 레이아웃 관리자를 포함하는 것인, 모바일 통신 장치.

청구항 9

제8항에 있어서, 상기 템플릿 정보는,

데이터 요소 객체의 세트, 및

커스텀 이벤트 객체의 세트를 포함하는 것인, 모바일 통신 장치.

청구항 10

제9항에 있어서, 상기 템플릿 정보는,

상기 레이아웃 관리자가 상기 템플릿 정보의 디스플레이를 제어하는 방법을 제어하기 위한 레이아웃 관리자 제어 정보와,

렌더러 제어 정보 중 적어도 하나를 더 포함하고,

상기 렌더러 제어 정보는,

상기 렌더러가 상기 데이터 요소 객체의 세트를 렌더링하는 방법, 및

상기 커스텀 이벤트 객체의 세트 중 적어도 하나의 커스텀 이벤트 객체에 기초하여 상기 렌더러가 상기 그래픽 인터페이스를 렌더링하는 방법

을 제어하기 위한 것인, 모바일 통신 장치.

청구항 11

제10항에 있어서, 상기 레이아웃 관리자 제어 정보는,

렌더링되기 이전에 상기 데이터 요소 객체의 세트 중 적어도 하나의 데이터 요소 객체의 길이를 동적으로 수정하기 위한 정보, 및

상기 렌더러가 렌더링할 수 없는 그래픽 인터페이스에 대한 상기 템플릿 정보의 디스플레이를 제어하기 위한 정보

중 적어도 하나를 포함하는 것인, 모바일 통신 장치.

청구항 12

제9항에 있어서, 상기 적어도 하나의 애플리케이션의 각각의 애플리케이션은,

적어도 하나의 애플리케이션 데이터 요소, 및

애플리케이션 코드를 포함하는 것인, 모바일 통신 장치.

청구항 13

제12항에 있어서, 상기 애플리케이션 코드는,

상기 적어도 하나의 애플리케이션 데이터 요소를 수정하는 코드,

적어도 하나의 애플리케이션 커스텀 이벤트를 생성하는 코드, 및

상기 상호작용 인터페이스를 사용하여 상기 애플리케이션 데이터 요소 중 적어도 하나 및 상기 애플리케이션 커스텀 이벤트 중 적어도 하나를 상기 미디어 엔진으로 전달하는 코드를 포함하는 것인, 모바일 통신 장치.

청구항 14

제13항에 있어서, 상기 적어도 하나의 애플리케이션 데이터 요소 및 상기 적어도 하나의 애플리케이션 커스텀 이벤트를 상기 미디어 엔진으로 전달하기 위한 코드는 상기 적어도 하나의 애플리케이션 데이터 요소를 상기 템플릿 정보의 상기 데이터 요소 객체의 세트 중 적어도 하나의 데이터 요소 객체에 바인딩시키고, 상기 적어도 하나의 애플리케이션 커스텀 이벤트를 상기 템플릿 정보의 상기 커스텀 이벤트 객체의 세트 중 적어도 하나의 커스텀 이벤트 객체에 바인딩시키는 것인, 모바일 통신 장치.

청구항 15

제8항에 있어서, 상기 미디어 엔진은 상기 애플리케이션에 이벤트 통지를 제공하는 이벤트 엔진을 더 포함하는, 모바일 통신 장치.

청구항 16

제8항에 있어서, 상기 상호작용 인터페이스는 컨텐츠 상호작용 애플리케이션 프로그래밍 인터페이스(API)를 포함하며,

상기 API는,

데이터 요소를 갱신하는 것,

커스텀 이벤트를 통지하는 것,

새로운 이벤트를 문의하는 것,

새로운 이벤트를 가져오는 것, 및

이벤트 리스너(event listener)를 등록하는 것

을 위한 호출들(calls)을 포함하는 것인, 모바일 통신 장치.

청구항 17

제9항에 있어서, 상기 템플릿 파일은,

상기 데이터 요소 객체의 세트,

상기 커스텀 이벤트 객체의 세트,

상기 레이아웃 관리자 제어 정보, 및

상기 렌더러 제어 정보의

마크업 언어 기술(markup language description)을 포함하는 것인, 모바일 통신 장치.

청구항 18

장치의 프로세서상에서 실행되며, 상기 장치의 디스플레이 상에 표시될 애플리케이션을 스킨화(skinning)하는 방법으로서,

메모리에 저장된 템플릿 파일을 파싱하는 단계;

템플릿 내의 상기 파싱된 정보를 메모리에 저장하는 단계;

상호작용 인터페이스를 통해 상기 프로세서에 의해 실행되는 애플리케이션으로부터 갱신된 데이터 요소 정보를 수신하는 단계;

수신된 갱신 데이터 요소 정보로 상기 템플릿을 갱신하는 단계;

레이아웃 관리자를 사용하여 상기 데이터 요소 정보의 디스플레이를 제어하는 단계;

상기 템플릿을 렌더러로 전달하는 단계; 및

상기 디스플레이 상에 상기 템플릿을 그래픽 인터페이스로서 렌더링하는 단계

를 포함하는, 애플리케이션 스킨화(skinning an application) 방법

청구항 19

제18항에 있어서, 상기 템플릿은,

상기 애플리케이션으로부터 수신된 데이터 요소를 나타내는 적어도 하나의 템플릿 데이터 요소, 및

커스텀 이벤트를 나타내는 적어도 하나의 템플릿 이벤트를 포함하는 것인, 애플리케이션 스킨화 방법.

청구항 20

제19항에 있어서, 상기 애플리케이션에 이벤트 통지를 제공하는 단계를 더 포함하는, 애플리케이션 스킨화 방법.

청구항 21

장치의 디스플레이 상에 표시될 애플리케이션을 스킨화하는 방법을 컴퓨터에서 실행하는 데 사용하기 위한 명령어(instructions) 또는 명령문(statements)을 저장하는 컴퓨터 판독가능한 기록 매체로서, 프로세서에 의해 실

행되는 명령어는,

메모리 내에 저장된 템플릿 파일을 파싱하는 것,

템플릿 내의 상기 파싱된 정보를 메모리에 저장하는 것,

상호작용 인터페이스를 통해 애플리케이션으로부터 갱신된 데이터 요소 정보를 수신하는 것,

수신된 갱신 데이터 요소 정보로 상기 템플릿을 갱신하는 것,

레이아웃 관리자를 사용하여 상기 데이터 요소 정보의 길이를 제어하는 것,

상기 템플릿을 렌더러로 전달하는 것, 및

상기 템플릿을 그래픽 인터페이스로서 렌더링하는 것을 포함하는 것인, 컴퓨터 판독가능한 기록 매체.

명 세 서

기 술 분 야

[0001] 본 특허 개시 내용은 일반적으로 복수의 장치에 통신을 제공하는 통신 시스템에 관한 것이며, 상세하게는 테마(theme)를 스킨화(skin)하는 시스템 및 방법에 관한 것이다.

[0002] 본 특허 문서의 개시 내용의 일부는 저작권의 보호를 받는 자료를 포함하고 있다. 저작권 소유자는, 특허 상표청 특허 파일 또는 기록에 있는 그대로, 임의의 자에 의한 본 특허 문서 또는 특허 개시 내용의 팩시밀리 재현에 대해 이의를 하지 않지만, 그렇지 않은 경우에는 무엇이든지 모든 저작권을 보유한다.

배 경 기 술

[0003] SVG(Scalable Vector Graphics)는 2차원 벡터 그래픽을 기술하는 XML(extensible markup language) 마크업이다. 애플리케이션은 그의 사용자 인터페이스(UI)를 표현하기 위해 SVG 파일을 사용할 수 있다. 이 SVG 파일은 애플리케이션의 스킨(skin)이라고도 할 수 있다. 기본적인 그래픽 UI 요건은 애플리케이션이 그의 상태를 그래픽으로 표현하고 그래픽 사용자 인터페이스로부터의 입력에 반응할 수 있어야만 한다는 것이다. 애플리케이션은 이하의 것들을 할 수 있어야만 한다.

[0004] a) GUI로부터의 사용자 입력에 반응한다. 이것이 없는 경우, 애플리케이션은 상태를 수동적으로 표현하는 것으로 제한될 것이다. 대부분의 애플리케이션은 어떤 형태의 사용자 상호작용을 필요로 한다.

[0005] b) 사용자에 대한 상태를 그래픽으로 반영한다. 이것이 없는 경우, 애플리케이션은 비지니스 로직에 관련된 상태를 표현할 수 없게 될 것이다. UI는 사용자 입력을 받는 것으로 제한된다. 대부분의 애플리케이션은 그의 상태에 관한 정보를 사용자에게 디스플레이해야만 한다. 일반적으로, a) 및 b) 둘다가 만족되어야만 한다.

[0006] 전통적인 애플리케이션 그래픽 사용자 인터페이스는 애플리케이션 로직(application logic)과 함께 하드-코딩된다(hard-coded). 즉, 사용자 인터페이스의 모습(look) 및 거동(behavior)이 Java 또는 C++과 같은 프로그램에서 애플리케이션 로직으로 프로그램되고 애플리케이션과 함께 임베딩(embedding)된다.

발명의 상세한 설명

[0007] 본 발명의 실시예는 장치상의 애플리케이션에 대한 그래픽 인터페이스를 생성하는 미디어 엔진에 관한 것이다. 이 미디어 엔진은 애플리케이션에 대한 변경을 미디어 엔진에 통지하는 상호작용 인터페이스, 그래픽 인터페이스를 렌더링하는 렌더러, 및 렌더러가 그래픽 인터페이스를 렌더링하는 방법을 제어하기 위한 템플릿 정보를 템플릿 파일로부터 파싱하는 파서를 포함한다.

[0008] 본 발명의 다른 실시예는 장치 상의 애플리케이션에 대한 그래픽 인터페이스를 생성하는 미디어 엔진에 관한 것이다. 이 미디어 엔진은 애플리케이션에 대한 변경을 미디어 엔진에 통지하는 상호작용 인터페이스, 그래픽 인터페이스를 렌더링하는 렌더러, 템플릿 파일로부터 템플릿 정보를 파싱하는 파서, 및 일련의 데이터 요소 객체 중 적어도 하나의 데이터 요소 객체의 길이를 동적으로 제어하는 레이아웃 관리자를 포함한다. 이 템플릿 정보는 일련의 데이터 요소 객체 및 일련의 커스텀 이벤트 객체를 포함한다. 이 템플릿 정보는 렌더러가 그래픽 인터페이스를 렌더링하는 방법을 제어한다.

[0009] 본 발명의 다른 실시예는 디스플레이, 오퍼레이팅 시스템, 입력 장치, 적어도 하나의 애플리케이션, 적어도 하

나의 템플릿 장치, 및 미디어 엔진을 포함하는 모바일 통신 장치에 관한 것이다. 미디어 엔진은 적어도 하나의 애플리케이션에 대한 변경을 미디어 엔진에 통지하는 상호작용 인터페이스, 디스플레이 상에 그래픽 인터페이스를 렌더링하는 렌더러, 및 렌더러가 디스플레이 상에서 그래픽 인터페이스를 렌더링하는 방법을 제어하기 위한 템플릿 정보를 적어도 하나의 템플릿 파일로부터 파싱하는 파서, 및 템플릿 정보의 디스플레이를 선택적으로 제어하는 레이아웃 관리자를 포함한다.

[0010] 본 발명의 다른 실시예는 장치 상의 애플리케이션을 스킨화하는 방법에 관한 것이며, 이 방법은 템플릿 파일을 파싱하는 단계, 파싱된 정보를 템플릿에 저장하는 단계, 상호작용 인터페이스를 통해 애플리케이션으로부터 갱신된 데이터 요소 정보를 수신하는 단계, 수신된 갱신 데이터 요소 정보로 템플릿을 갱신하는 단계, 레이아웃 관리자를 사용하여 데이터 요소 정보의 길이를 제어하는 단계, 이 템플릿을 렌더러로 전달하는 단계, 및 이 템플릿을 그래픽 인터페이스로서 렌더링하는 단계를 포함한다.

[0011] 본 발명의 다른 실시예는 컴퓨터에 의해 판독 및 실행될 수 있는 컴퓨터 실행가능 명령어를 포함하는 신호를 전달하는 전송 신호 반송파에 관한 것이며, 이 컴퓨터 실행가능 명령어는 장치 상의 애플리케이션을 스킨화하는 방법을 실행하는 데 사용되고, 이 방법은 템플릿 파일을 파싱하는 단계, 파싱된 정보를 템플릿에 저장하는 단계, 상호작용 인터페이스를 통해 애플리케이션으로부터 갱신된 데이터 요소 정보를 수신하는 단계, 수신된 갱신 데이터 요소 정보로 템플릿을 갱신하는 단계, 레이아웃 관리자를 사용하여 데이터 요소 정보의 길이를 제어하는 단계, 이 템플릿을 렌더러로 전달하는 단계, 및 이 템플릿을 그래픽 인터페이스로서 렌더링하는 단계를 포함한다.

[0012] 본 발명의 다른 실시예는 장치 상의 애플리케이션을 스킨화하는 방법을 컴퓨터에서 실행하는 데 사용하기 위한 명령어(instruction) 또는 명령문(statement)을 저장하는 컴퓨터 판독가능 매체에 관한 것이며, 이 방법은 템플릿 파일을 파싱하는 단계, 파싱된 정보를 템플릿에 저장하는 단계, 상호작용 인터페이스를 통해 애플리케이션으로부터 갱신된 데이터 요소 정보를 수신하는 단계, 수신된 갱신된 데이터 요소 정보로 템플릿을 갱신하는 단계, 레이아웃 관리자를 사용하여 데이터 요소 정보의 길이를 제어하는 단계, 이 템플릿을 렌더러로 전달하는 단계, 및 이 템플릿을 그래픽 인터페이스로서 렌더링하는 단계를 포함한다.

실 시 예

[0028] 이제부터, 이하의 도면을 참조하여 단지 예로서 본 발명의 실시예에 대해 기술할 것이다.

[0029] 스킨화의 목적은 애플리케이션의 프리젠테이션(GUI)을 비지니스 로직으로부터 분리시키고, 애플리케이션 코드에 "지연 바인딩(late binding)"될 수 있도록 생성된 어떤 외부 파일을 통해 GUI가 정의될 수 있게 해주는 데 있다. 이 지연 바인딩을 가능하게 해주기 위해, 애플리케이션 로직과 스킨 간에 공통의 합의(understanding) 또는 약정(contract)이 있어야만 한다. 애플리케이션이 UI를 제어하기 위해 인터페이스할 수 있는 인터페이스가 정의되어야만 하고, 이 인터페이스를 만족시키는 스킨이 생성되어야만 한다.

[0030] 본 발명은 애플리케이션에 대한 테마를 스킨화하는 테마 스킨화 문서(skinning themes document)를 제공한다. 테마 스킨화 문서는 애플리케이션에 의해 발생된 데이터를 표현하는 적어도 하나의 데이터 요소 및 애플리케이션에 의해 발생된 이벤트를 표현하는 적어도 하나의 커스텀 이벤트를 포함한다. 테마 스킨화 문서에 의해 정의된 GUI를 애플리케이션 정보, 즉, 데이터 요소 및 커스텀 이벤트에 지연 바인딩하는 것을 고려하고 있는 인터페이스가 미디어 엔진에 의해 제공된다.

[0031] 이제부터,본 발명의 실시예들이 가장 잘 제조 및 사용될 수 있는 방법의 다양한 예를 참조하여 이들 실시예에 대해 기술한다. 편의상, 유사 또는 대응하는 부분을 나타내기 위해 유사 참조 번호들이 상세한 설명과 일부 도면들에 걸쳐 사용되고 있으며, 여기서 여러 구성요소들은 반드시 실제적인 비례를 갖고 도시되어 있는 것은 아니다.

[0032] 도 1은 본 발명의 실시예에 따른, 모바일 장치(100)의 일례의 기능을 개략적으로 나타낸 것이다. 모바일 장치(100)는 디스플레이(101), 입력 장치(102), 오퍼레이팅 시스템(operating system)(103), 다양한 애플리케이션(105), 미디어 엔진(110a) 및 템플릿 라이브러리(115)를 포함한다. 모바일 장치 상에 다수의 애플리케이션이 존재할 수 있지만, 이하의 설명은 단지 하나의 애플리케이션(105)을 갖는 모바일 장치에 대해 기술한다. 템플릿 라이브러리(115)는 다수의 템플릿 파일을 포함할 수 있다. 다른 대안으로서, 템플릿 라이브러리(115) 대신에 하나의 템플릿 파일이 사용될 수 있다. 애플리케이션(105)의 예들은 홈 스크린(home screen) 애플리케이션, 일정표(calendar) 애플리케이션, 이메일 애플리케이션, 시간 애플리케이션, 기타 등등을 포함한다.

[0033] 도 2는 본 발명에 따른, 애플리케이션(105)의 일례의 기능을 개략적으로 나타낸 것이다. 애플리케이션(105)은 어떤 비지니스 로직을 수행하는 적어도 애플리케이션 코드(106) 및 애플리케이션 코드(106)가 조작 또는 수정할 수 있는 애플리케이션 데이터 요소(107)의 집합체(collection)이다. 애플리케이션(105)은 장치의 디스플레이 상에 애플리케이션 정보를 디스플레이하기 위해 미디어 엔진(110a)과 관련하여 동작한다. 비지니스 로직에 부가하여, 애플리케이션 코드(106)는 또한 컨텐츠 상호작용 로직 또는 코드(108)도 포함한다. 이 컨텐츠 상호작용 코드는 미디어 엔진(110a)에 데이터 요소(107)에 대한 갱신 또는 커스텀 이벤트의 발생을 통지하기 위해 미디어 엔진(110a)과 통신을 한다. 이 통신은 109로 개략적으로 나타내어져 있다. 데이터 요소(107)에 대한 갱신을 미디어 엔진에 알려주는 통신은 109a로 나타내어져 있으며, 커스텀 이벤트의 발생을 미디어 엔진에 알려주는 통신은 109b로 나타내어져 있다.

[0034] 도 3a는 본 발명에 따른 미디어 엔진(110a)의 일례의 기능을 개략적으로 나타낸 것이다. 미디어 엔진(110a)은 그래픽 인터페이스를 장치 디스플레이에 렌더링하는 렌더러(111), 애플리케이션(105)과 인터페이스하는 컨텐츠 상호작용 API(112), 그래픽 인터페이스 정보를 보유하는 템플릿(113), 및 템플릿 파일로부터 그래픽 인터페이스 정보를 파싱하는 파서(114)를 포함한다. 미디어 엔진(110a)이 템플릿 파일로부터 파싱되는 템플릿 정보를 보유하는 템플릿(113)을 생성하지만, 미디어 엔진(110a)은 다른 대안으로서 중간 템플릿(113) 없이 파서(114)로부터 직접 그 정보를 사용할 수 있다.

[0035] 스킨화하는 한가지 방법은 애플리케이션의 사용자 인터페이스에 대한 템플릿 파일로서 역할하는 SVG(scalable vector graphics) 문서를 정의하는 것이다. 보다 구체적으로는, 사용자 입력에 반응하는 것 및 애플리케이션 상태를 그래픽으로 반영하는 것의 두가지 요건을 모두 만족시키기 위해, SVG 템플릿 문서와 애플리케이션 코드(106) 사이의 약정을 구현하는 메카니즘이 개발되었다. 일 실시예에서, 애플리케이션(105)은 사용자 입력을 처리하는 일을 맡고 있고, 미디어 엔진(110a)은 그래픽 인터페이스(120)를 정의 및 디스플레이하는 일을 맡고 있다. 애플리케이션 상태를 그래픽으로 반영하기 위해, 애플리케이션 코드(106)는 애플리케이션(105)에 의해 발생되는 특정의 데이터 및 이벤트를 나타내는 일련의 "데이터 요소" 및 "커스텀 이벤트"를 정의한다. 데이터 요소(107)는 애플리케이션(105)이 조작 또는 수정하는 어떤 정보이며 그래픽 인터페이스(120) 상에 제공될 수 있다. 데이터 요소(107)는 문자열인 이름을 할당받는다. 커스텀 이벤트는 그래픽 사용자 인터페이스(120)에 반영될 필요가 있을 수 있는 애플리케이션의 비지니스 로직에서 일어나는 어떤 중요한 이벤트이다. 데이터 요소(107)와 유사하게, 커스텀 이벤트도 문자열인 이름을 할당받는다. 이들 이벤트의 결과로서 어떤 사용자 인터페이스 변경이 일어나는지 또는 데이터 요소(107)의 정보를 어떻게 디스플레이하는지를 정의하는 것은 애플리케이션(105)의 책임이 아니다. 애플리케이션의 책임은 데이터 요소(107)와 연관된 정보를 갱신할 필요가 있을 때마다 또 커스텀 이벤트가 일어날 때 미디어 엔진(110a)에 알려주는 것이다.

[0036] 일 실시예에서, 템플릿 파일은 그래픽 인터페이스에 디스플레이되는 커스텀 이벤트 및 데이터 요소를 정의한다. 애플리케이션 코드(106)는 어떤 애플리케이션 데이터 요소(107) 또는 커스텀 이벤트(사용자 입력 이벤트를 포함할 수 있음)를 그래픽 인터페이스에 정의된 데이터 요소 및 이벤트에 바인딩할지를 결정한다.

[0037] 데이터 요소(107)의 이름 및 커스텀 이벤트의 이름을 사용하여 애플리케이션(105)에 대해 테마 스킨화 문서가 작성된다. 테마 스킨화 문서 또는 템플릿 파일은 데이터 요소(107) 및 커스텀 이벤트와 연관된 정보를 어떻게 디스플레이할지를 기술한다.

[0038] 미디어 엔진(110a)은 애플리케이션(105)과 그래픽 인터페이스 템플릿 파일 간의 인터페이스로서 역할한다. 애플리케이션(105)은 데이터 요소(107)에 대한 갱신 및 커스텀 이벤트의 발생을 미디어 엔진(110a)에 통지한다. 미디어 엔진(110a)은 스킨 템플릿 파일을 사용하여 이 정보를 디스플레이(101) 상에 어떻게 디스플레이할지를 정의한다. 스킨 문서를 템플릿으로 사용하여, 미디어 엔진(110a) 또는 보다 상세하게는 미디어 엔진(110a)의 렌더러(111)가 그래픽 인터페이스(120)를 디스플레이(101)에 렌더링한다.

[0039] 미디어 엔진(110a)은 그래픽 인터페이스(120)를 렌더링하는 방법을 기술하는 템플릿 파일을 사용한다. 미디어 엔진은 파서(114)를 사용하여 템플릿 파일로부터 템플릿 정보를 파싱하고 이를 미디어 엔진(110a)이 액세스할 수 있는 템플릿(113)에 저장할 수 있다. 템플릿 파일이 마크업 언어를 사용하여 그래픽 인터페이스(120)를 기술하는 경우, 파서(114)는 마크업 언어를 판독하고 템플릿 정보를 템플릿(113)에 저장할 수 있을 것이다. 예를 들어, 템플릿 파일이 SVG로 그래픽 인터페이스(120)를 정의하는 경우, SVG 파서는 SVG 파일 내의 템플릿 정보를 판독하고 파싱된 정보를 템플릿(113)에 저장한다. 파서는 트랜스코더(transcoder)라고도 할 수 있는데, 그 이유는 파서가 템플릿 파일 코드를 템플릿(113)을 표현하는 코드로 변환하기 때문이다. 템플릿(113)은 미디어 엔진(110a) 및 렌더러(111)에 의해 예측되는 형태로 템플릿 정보를 보유한다. 앞서 기술한 바와 같이, 템플릿

(113)은 반드시 필요하지는 않는다. 파서(114)가 템플릿 정보를 직접 미디어 엔진(110a) 또는 렌더러(111)에 제공할 수 있다.

[0040] 상기한 미디어 엔진(110a)은 사용자 입력의 처리에 관여하지 않는다. 미디어 엔진(110a)은 그래픽 인터페이스(120)를 정의하는 템플릿 파일, 및 데이터 요소(107) 및 커스텀 이벤트에 관한 애플리케이션(105)으로부터의 통지를 수신한다. 상기 예에서, 애플리케이션(105)은 사용자 입력을 처리하는 일을 맡고 있다. 이것은 입력 장치 드라이버에 의해 제공되는 기능과, 오퍼레이팅 시스템 기능 또는 공지되어 있을지도 모르는 다른 방법들과 상호작용함으로써 달성될 수 있다.

[0041] 도 3b는 본 발명의 일 실시예에 따른, 미디어 엔진(110b)의 다른 예의 기능을 개략적으로 나타낸 것이다. 미디어 엔진(110b)은 템플릿 정보의 디스플레이를 선택적으로 제어하는 레이아웃 관리자(118)를 더 포함한다.

[0042] 도 3c는 본 발명의 다른 실시예에 따른, 미디어 엔진(110c)의 다른 예의 기능을 개략적으로 나타낸 것이다. 미디어 엔진(110c)은 앞서 기술한 미디어 엔진(110a)과 유사하지만, 미디어 엔진(110c)은 이벤트 엔진(117)도 포함한다. 이벤트 엔진(117)은 미디어 엔진(110c)에 기능을 추가한다. 이는 사용자 입력 제어를 제공할 수 있다. 이벤트 엔진(117)의 사용자 입력 기능을 사용하는 애플리케이션(105)은 애플리케이션(105)에 임의의 사용자 입력을 통지하는 코드를 포함할 수 있다. 당업자라면 많은 다른 형태의 사용자 입력이 가능하다는 것을 잘 알 것이다. 애플리케이션(105)은 다른 방식으로 통지받을 수 있다.

[0043] 애플리케이션(105)은 사용자 입력이 있을 때마다 통보받고자 함을 이벤트 엔진(117)에 통지를 할 수 있다. 이 통지는 미디어 엔진(110c)의 컨텐츠 상호작용 API(112b)에 대한 API 호출(call)에 의해 달성될 수 있다. 컨텐츠 상호작용 API(112b)는 애플리케이션(105)에 사용자 입력 제어를 제공하기 위한 부가적인 호출들을 포함한다. 이 호출은 애플리케이션(105)에 통지하기 위하여 사용될 수 있는 정보를 이벤트 엔진(117)이 포함하고 있어야만 한다. 이 정보는 애플리케이션 식별자, 및 이벤트가 발생할 때 이벤트 엔진(117)이 호출해야만 하는 함수를 포함할 수 있다. 이 호출은 또한 애플리케이션(105)이 관심을 가지고 있는 사용자 입력 이벤트의 유형을 나타낼 수 있다. 예를 들어, 애플리케이션(105)은 단지 스크롤 휠로부터 또는 키보드나 포인팅 장치 등으로부터의 입력만을 통지받고자 할 수 있다. 사용자 입력 이벤트가 있을 때마다, 이벤트 엔진(117)은 애플리케이션(105)에게 통지해야 할 것이 있는지를 검사하고, 만약 통지해야할 어떤 것이 있는 경우, 이벤트의 통지를 전송한다.

[0044] 애플리케이션은 입력 이벤트의 통지를 수신하고자 함을 이벤트 엔진(117)에게 반드시 통지할 필요는 없다. 다른 대안으로서, 애플리케이션은 이벤트가 일어났는지 및 이벤드를 수신해야만 하는지를 판정하기 위해 이벤트 엔진(117)에 폴링할 수 있다. 애플리케이션은 사용자 입력 이벤트를 문의 또는 요청하는 컨텐츠 상호작용 API(112b)에 대한 호출을 한다.

[0045] 상기한 바와 같이, 본 발명에 따른 애플리케이션(105)은 다양한 방법으로 사용자 입력을 획득할 수 있다. 이상의 설명은 사용자 입력 이벤트에 뿐만 아니라 시스템 이벤트 또는 기타 발생된 이벤트에도 적용될 수 있다. 예를 들어, 애플리케이션(105)은 디스플레이(101)가 새로 고침될 때마다 통지를 수신하기 위해 컨텐츠 상호작용 API(112b)를 통해 이벤트 엔진(117)에 등록을 할 수 있다.

[0046] 상기한 이벤트 핸들링에 부가하여, 이벤트 엔진(117)은 또한 미디어 엔진(110c)의 컨트롤러로서도 역할할 수 있다. 이벤트 엔진(117)은 그래픽 인터페이스(120)가 새로고침되는 때를 제어할 수 있다. 이것은 어떤 주파수로 이벤트를 발생하는 이벤트 엔진(117) 타미어를 사용하여 달성될 수 있다. 이벤트가 발생할 때, 이벤트 엔진(117)은 템플릿(113)에 액세스하여 데이터 요소(107)의 값 및 이벤트의 발생에 관한 정보와 함께 렌더러(111)에 템플릿 정보를 제공할 수 있다. 렌더러(111)는 이어서 이 정보를 사용하여 그래픽 인터페이스(120)를 디스플레이(101)에 렌더링한다.

[0047] 이상의 단락에 기술된 바와 같이, 미디어 엔진(110c) 또는 보다 상세하게는 이벤트 엔진(117)은 템플릿 정보를 데이터 요소 및 이벤트 정보와 따로 저장한다. 모든 정보를 템플릿(113)에 저장하는 것이 바람직할 수 있다. 데이터 요소(107)에 대한 변경의 통지가 컨텐츠 상호작용 API(112b)를 통해 도달할 때, 이벤트 엔진(117)은 이 정보를 템플릿(113)에 저장할 수 있다. 이와 같이, 렌더러(111)는 그래픽 인터페이스(120)를 렌더링하기 위해 템플릿(113) 객체를 참조하기만 하면 된다. 이것은 다양한 방법으로 달성될 수 있다. 예를 들어, 이벤트 엔진(117)은 템플릿(113) 정보의 복사본을 렌더러(111)에 전달할 수 있거나 템플릿(113)에 대한 참조를 렌더러(111)에 전달할 수 있다. 다른 대안으로서, 렌더러(111)는 특정의 위치에 있는 템플릿(113)에 액세스하도록 설계될 수 있다. 이벤트 엔진(117)은 적절한 템플릿이 그 위치에 있도록 보장해주는 일을 맡고 있다. 이 방법이 유익하게도 사용될 수 있는데, 그 이유는 이벤트 엔진(117)이 애플리케이션 데이터 요소 정보 및 이벤트 정보로

갱신된 채로 유지하는 수많은 템플릿을 미디어 엔진(110c)이 가지고 있을 수 있기 때문이다. 이벤트 엔진(117)은 이어서 원하는 템플릿으로 특정의 위치를 갱신함으로써 그래픽 인터페이스(120)를 전환하는 데 사용될 수 있다.

[0048] 이상의 설명은 컨텐츠 상호작용 API(112, 112b)에 대해 행해질 수 있는 다양한 호출에 대해 기술하였다. 이하에는 미디어 엔진(110a, 110b, 110c)이 지원해야만 하는 가능한 API 호출의 리스트가 있다. 이는 최소한 데이터 갱신 호출(update data call)을 지원해야만 한다. 이것을 지원하지 않는 경우, 애플리케이션은 디스플레이하기 위해 미디어 엔진(110a, 110b, 110c)으로 정보를 전달할 수 없다.

[0049] int update_data_element (string element_ID, string element_value)

[0050] - 새로운 element_value으로 element_ID를 갱신하도록 미디어 엔진에 통지하는 데 사용됨

[0051] - 에러를 위해 int를 반환하며, 다른 유형일 수 있거나 반환이 없을 수 있음

[0052] int notify_custom_event (string custom_event_ID)

[0053] - custom_event_ID의 발생을 미디어 엔진에 통지하는 데 사용됨

[0054] - 에러를 위해 int를 반환하며, 다른 유형일 수 있거나 반환이 없을 수 있음

[0055] bool query_new_event (string event_filter)

[0056] - 새로운 이벤트가 발생했는지를 판정하는 데 사용됨. 이 event_filter는 관심을 끄는 이벤트의 유형을 지정하는 데 사용될 수 있다. 이 호출은 통상적으로 이벤트 엔진을 사용하는 미디어 엔진에서 구현한다.

[0057] - event_filter과 일치하는 새로운 이벤트가 발생한 경우에 true를 반환하고, 그렇지 않은 경우 false를 반환한다.

[0058] string get_new_event (string event_filter)

[0059] - event_filter와 일치하는 새로운 이벤트를 가져오는 데 사용됨. 이 호출은 통상 이벤트 엔진을 사용하는 미디어 엔진에서 구현된다.

[0060] - 이벤트를 나타내는 문자열을 반환함

[0061] bool register_event_listner(string application_ID, string listener_ID, string event_filter)

[0062] - 이벤트 리스너(event listener)를 미디어 엔진에 등록하는 데 사용됨. application_ID는 애플리케이션을 식별하는 데 사용되는 식별자이다. listener_ID는 등록할 리스너를 결정하는 데 사용된다. 이것은 애플리케이션의 함수 ID일 수 있다. event_filter는 관심을 끄는 이벤트의 유형을 지정하는 데 사용될 수 있다. 이 호출은 통상 이벤트 엔진을 사용하는 미디어 엔진에서 구현된다.

[0063] - application_ID와 연관된 listener_ID가 event_filter와 일치하는 이벤트를 통지받기 위해 등록될 수 있는 경우 true를 반환한다.

[0064] 상기한 호출 및 설명은 단지 예에 불과하며, 당업자라면 미디어 엔진(110a, 110b)의 기능을 유지하면서 서로 다른 파라미터 및 반환값을 갖는 더 적은 또는 더 많은 호출이 포함될 수 있다는 것을 잘 알 것이다.

[0065] 도 3d는 본 발명의 일 실시예에 따른, 미디어 엔진(110d)의 다른 예의 기능을 개략적으로 나타낸 것이다. 미디어 엔진(110d)은 앞서 기술한 미디어 엔진(110a, 110c)과 유사하지만, 미디어 엔진(110d)은 미디어 엔진(110b)과 유사하게 레이아웃 관리자(118)도 갖는다. 레이아웃 관리자(118)는 템플릿(113)에서 발견되는 데이터 요소의 동적 레이아웃을 처리하는 데 사용될 수 있다.

[0066] 데이터 요소들은 길이가 변할 수 있는 문자열로서 미디어 엔진에 전달될 수 있다. 문자열의 길이가 변하는 것은 그래픽 인터페이스(120)를 레이아웃할 때 문제를 제기한다. 문자열이 너무 긴 경우, 이는 그래픽 인터페이스(120)의 다른 부분과 중첩할 수 있다. 텍스트가 너무 짧은 경우, 다른 요소들이 적절히 배열되지 않을 수 있다. 미디어 엔진(110a, 110c)의 상기 예들에서, 비록 그래픽 인터페이스(120)의 규격의 유연성에 대비를 하고 있지만, 데이터 요소들의 크기를 동적으로 처리하는 쉬운 방법이 기술되어 있지 않았다. 미디어 엔진이 너무 긴 모든 문자열을 절단(truncate)하는 등의 문자열 길이에 대한 처리방법에 대하여 어떤 미리 정해진 결정을 해야만 한다. 이것이 잘 동작하더라도, 레이아웃 관리자(118)는 데이터 요소의 동적 디스플레이를 어떻게 처리해야 하는지를 지정하는 방법을 제공함으로써 미디어 엔진(110b, 110d)에 훨씬 더 많은 유연성을 제공한다. 이것

은 그래픽 인터페이스 설계자가 레이아웃 관리자(118) 제어 정보를 템플릿 파일에 포함시킴으로써 데이터 요소들이 어떻게 처리되어야 하는지를 기술할 수 있게 해준다. 일례로서, 데이터 요소가 10 문자 길이가 되어야 하고 또 미디어 엔진에 제공된 데이터 요소가 10 문자보다 더 긴 경우, 5 문자 길이로 절단되어야만 하며 문자열 '(...)'이 문자열의 끝에 첨부되어야 한다는 것을 지정하기 위해 템플릿(113) 및 레이아웃 관리자(118)가 사용될 수 있다. 다른 예로서, 그래픽 인터페이스 설계자는 특정의 데이터 요소의 문자들이 모두 대문자로 나타내도록 하고자 할 수 있다. 이것은 템플릿 파일에 지정될 수 있다. 레이아웃 관리자는 데이터 요소 문자열을 수신하고 이를 템플릿(113)에 의해 결정된 대로 수정하며, 이 경우에, 모든 문자를 대문자로 변경한다. 이러한 데이터 요소 조작은 레이아웃 관리자(118)에 의해 수행된다. 레이아웃 관리자는 데이터 요소의 길이가 어떻게 수정되어야만 하는지를 지정하는 레이아웃 관리자(118)에 대한 지시를 템플릿(113) 또는 템플릿 파일에 포함시킴으로써 제어될 수 있다. 이어서, 렌더러(111)는 레이아웃 관리자(118)에 의해 준비된 대로 문자열을 디스플레이한다. 레이아웃 관리자(118)는 텍스트 스타일의 데이터 요소가 지정되고 동적으로 제어될 수 있게 해준다.

[0067] 도 3d에 개략적으로 나타낸 바와 같이, 레이아웃 관리자는 템플릿에 기술된 동적 레이아웃에 따라 데이터 요소를 변경할 수 있으며, 이들 변경이 템플릿(113)에 직접 저장될 수 있다. 다른 대안으로서, 데이터 요소는 애플리케이션(105)으로부터 제공되는 템플릿(113)에 (또는 템플릿(113)과 따로) 저장될 수 있다. 레이아웃 관리자(118)는 데이터 요소가 렌더러(111)에 제공될 때 이들을 수정할 수 있다. 레이아웃 관리자는 데이터 요소의 배치, 크기, 가시성, 기타 등등의 데이터 요소의 디스플레이 특성을 수정할 수 있다. 이와 같이, 렌더러는 애플리케이션(105)에 의해 제공되고 템플릿(113)에 기술된 방식으로 레이아웃 관리자(118)에 의해 수정되는 데이터 요소를 기술하는 문자열로서 데이터 요소를 수신한다.

[0068] 레이아웃 관리자는 텍스트 요소 이외의 요소들의 디스플레이를 제어하는 데 사용될 수 있다. 레이아웃 관리자는 그래픽 정보의 디스플레이 특성도 동적으로 제어하는 데 사용될 수 있다. 그래픽 인터페이스 설계자는 그래픽 또는 이미지의 배치에 의해 덮여지는 텍스트가 없는 경우에만 어떤 위치에 그래픽 또는 이미지를 디스플레이하고자 할 수 있다. 레이아웃 관리자(118)는 요소의 디스플레이를 제어하는 수단을 제공하는 데 사용될 수 있다. 예를 들어, 그래픽 인터페이스(120)가 텍스트가 없는 위치에 아이콘이 디스플레이되어야 하는 것으로 기술하고 있는 경우, 그 아이콘 요소가 레이아웃 요건을 지정하는 레이아웃 관리자 제어 정보와 함께 레이아웃 관리자(118)로 전달될 수 있다. 레이아웃 관리자는 이어서 요소가 렌더링되기 이전에 템플릿(113) 내의 요소의 디스플레이 정보를 동적으로 변경할 수 있다. 레이아웃 관리자(118)는 아이콘의 배치를 지정하는 좌표를 변경할 수 있거나, 그래픽 인터페이스(120) 상에 이용가능한 공간이 없는 것으로 판정할 수 있으며, 따라서 아이콘이 디스플레이되어서는 안된다. 동적 디스플레이 특성을 갖는 수정된 요소는 이어서 그래픽 인터페이스(120)에 렌더링하기 위해 렌더러(111)로 전달될 수 있다.

[0069] 레이아웃 관리자(118)는 렌더러(111)에 의해 렌더링되기 이전에 데이터 요소들을 조작하는 수단을 제공할 수 있는 것은 물론 렌더러(111)가 적절히 렌더링할 수 없을지도 모르는 요소들의 디스플레이 특성을 동적으로 제어하는 수단을 제공할 수 있다. 템플릿(113)은 레이아웃 관리자(118)를 제어하는 정보를 지정할 수 있다.

[0070] 도 4는 예시적인 애플리케이션(400)의 리스트를 의사 코드로 나타낸 것이다. 예시적인 애플리케이션(400)은 본 발명의 시스템 및 방법의 기능을 설명하기 위한 것이다. 예시적인 애플리케이션(400)은 2개의 데이터 요소를 정의한다. 첫번째 데이터 요소(405)는 'data_int'라는 이름의 정수이고, 두번째 데이터 요소(407)는 'data_str'이라는 이름의 문자열이다. 예시적인 애플리케이션(400)은, 'scroll_wheel' 사용자 입력을 수신할 때마다, data_int의 값(초기값은 0임)에 1을 가산한다. 예시적인 애플리케이션(400)은 오퍼레이팅 시스템(103)으로부터 이 정보를 수신할 수 있다. 그에 부가하여 또는 다른 대안으로서, 애플리케이션은 앞서 기술한 이벤트 엔진(117)의 사용자 입력 기능을 사용하도록 설계될 수 있다. data_int의 값이 5일 때, 예시적인 애플리케이션(400)은 data_str의 값을 'Big Int'로 변경하고 'big_int_event'라는 이름의 커스텀 이벤트를 생성한다. data_int의 값이 10일 때, 이 값은 0으로 리셋되고, data_str의 값은 'Small Int'로 변경되며, 'small_int_event'라는 이름의 커스텀 이벤트가 생성된다.

[0071] 예시적인 애플리케이션(400)은 'update_data_element'라는 이름의 미디어 엔진 API 호출을 사용하여 데이터 요소(405, 407)의 값에 대한 어떤 변경이라도 미디어 엔진(110a, 110b, 110c, 110d)에 통지한다. update_date_element는 데이터 요소의 ID를 식별해주는 문자열을 기대하고 있다. 이것은 애플리케이션 data_element의 이름과 동일하도록 선택된다. 이것은 시스템의 요건이 아니지만, 데이터 요소를 갱신할 때 일관성을 보장해주는 용이한 방법이다. 데이터 요소들의 어레이 내의 요소를 나타내는 data_element_ID에 대한 문자열은 어레이 이름에 특정의 요소의 인덱스 번호를 첨부함으로써 구성될 수 있다. update_data_element는 또한 데이터 요소(405, 407)의 갱신된 값을 나타내는 값을 기대하고 있다. 데이터 요소에 대한 갱신을 미디어

엔진에 통지하는 데 사용되는 API의 예시적인 정의는 다음과 같다.

[0072] Int update_data_element (string data_element_ID, string value)

[0073] 상기 정의된 update_data_element는 에러 검출을 위해 사용될 수 있는 정수를 반환한다. 이는 데이터 요소(107)를 찾을 수 없는 경우 또는 값을 변경할 수 없는 경우 또는 값이 적절히 갱신된 경우, 기타 등등의 여러가지 조건을 검출하는 데 사용될 수 있다. 이 호출은 데이터 요소(107)의 ID를 나타내는 문자열을 기대하고 있다. 이 ID는 데이터 요소(107)를 어떻게 디스플레이할지를 기술하기 위해 템플릿 파일이 사용하는 문자열이다. 이는, 다른 대안으로서, 템플릿 파일이 데이터 요소(107)를 식별하기 위해 동일한 유형을 사용하는 한, 정수 등의 다른 유형일 수 있다. 데이터 요소(107)의 값은 문자열로서 정의되며, 다시 말하면 이것은, 템플릿 파일이 유형을 어떻게 디스플레이할지를 기술하고 있는 한, 다른 유형으로 변경될 수 있다.

[0074] 예시적인 애플리케이션(400)은 'notify_custom_event'라는 이름의 미디어 엔진 API 호출을 사용하여 어떤 커스텀 이벤트의 발생이라도 미디어 엔진(110a)에 통지한다. notify_custom_event 호출은 커스텀 이벤트의 ID를 식별해주는 문자열을 기대하고 있다. 이 문자열은, 템플릿 파일이 이벤트를 식별하기 위해 동일한 유형을 사용하는 한, 정수 또는 다른 유형으로 대체될 수 있다. 이벤트의 발생을 미디어 엔진에 통지하기 위해 사용되는 API의 일례는 다음과 같다.

[0075] Int notify_custom_event (string event_ID,)

[0076] 이상에 정의된 notify_custom_event는 에러 검출을 위해 사용될 수 있는 정수를 반환한다. 이 호출은 커스텀 이벤트의 ID를 나타내는 문자열을 기대하고 있다. 이 ID는 이벤트를 식별하기 위해 템플릿 파일이 사용하는 문자열이다. 이는, 다른 대안으로서, 템플릿 파일이 이벤트를 식별하기 위해 동일한 유형을 사용하는 한, 정수 등의 다른 유형일 수 있다.

[0077] 도 4의 예시적인 애플리케이션(400)은 정보가 어떻게 또는 언제 디스플레이되는지를 제어하는 어떤 코드도 포함하지 않는다. 이는 사용자 입력을 제어하고 데이터 요소(405, 407)에 대한 갱신 및 이벤트의 발생을 미디어 엔진(110a, 110b, 110c, 110d)에 통지하는 코드를 포함한다. 미디어 엔진(110a, 110b, 110c, 110d)은 템플릿(113)을 사용하여 정보를 디스플레이에 어떻게 렌더링할지를 결정한다.

[0078] 예시적인 프로그램에 대한 그래픽 인터페이스(120)를 지정하는 템플릿 파일이 마크업 언어를 사용하여 생성될 수 있다. 이하는 예시적인 애플리케이션(400) 정보를 어떻게 디스플레이할지를 기술하는 가능한 템플릿 파일의 일례이다.

[0079] 예시적인 GUI 1:

[0080] <text id ="data_int" x="5" y="5" font-family="myfont"

[0081] font-size="15" fill="black">not set</text>

[0082] <set xlink:href="data_int" attributeName="fill"

[0083] to="green" begin=" custom (Big_Int) "

[0084] end="custom (Small_Int)"/>

[0085] <set xlink : href="data_int" attributeName="fill"

[0086] to="red" begin="custom (Small_Int) "

[0087] end="custom (Big_Int) " />

[0088] 이상에서는 예시적인 애플리케이션에서 사용될 수 있는 템플릿 파일로부터의 exert를 나타낸 것이다. 이는 'data_int'의 ID를 갖는 <text> 엘리먼트를 정의한다. 이것은 애플리케이션이 미디어 엔진(110a, 110b, 110c, 110d)으로 전송하는 동일한 ID이다. <text> 엘리먼트는 어떻게 텍스트를 형식화하고 이를 디스플레이(101) 상에 배치할지를 지정한다. exert는 또한 2개의 <set> 엘리먼트를 정의한다. 이들 엘리먼트는 렌더러(111)가 애플리케이션 이벤트에 기초하여 정보를 렌더링하는 방법을 변경하는 데 사용된다. 'Big_Int' 커스텀 이벤트가 예시적인 애플리케이션(400)으로부터 미디어 엔진(110a, 110b, 110c, 110d)에 도착할 때, data_int 요소의 채움 컬러(fill colour)가 녹색으로 변경된다. set 엘리먼트는 또한 연관된 <set> 특성을 사용하여 언제 중단해야 하는지를 정의하며, 상기 예에서, 이것은 커스텀 이벤트 'Small_Int'가 가 미디어 엔진(110a, 110b, 110c, 110d)에 수신되는 때이다. 두번째 set 엘리먼트는 커스텀 이벤트 'Small_Int'가 수신될 때 data_int의 디스플

레이 속성을 기술한다. 이는 요소의 채움 컬러를 적색으로 설정하는 것을 기술한다.

[0089] 상기 템플릿은 2개의 데이터 요소(405, 407)를 정의하는 예시적인 애플리케이션(400)에 대한 하나의 데이터 요소(405)를 디스플레이하는 것만을 기술하고 있다. 애플리케이션 데이터 요소 data_str이 디스플레이되지 않는다. 미디어 엔진(110a, 110b, 110c, 110d)은 예시적인 애플리케이션(400)으로부터 수신하는 이 데이터 요소(407)의 어떤 갱신도 폐기 또는 무시할 수 있다.

[0090] 동일한 예시적인 애플리케이션에 대한 다른 그래픽 인터페이스(120)를 기술하는 다른 가능한 템플릿 파일은 다음과 같다.

[0091] 예시적인 GUI 2:

[0092] <text id ="data_int" x="5" y="5" font-family="myfont"

[0093] font-size="15" fill="black">not set</text>

[0094] <text id ="data_str" x="5" y="25" font-family="myfont2"

[0095] font-size="10" fill="grey">not set</text>

[0096] 상기한 예시적인 그래픽 인터페이스(120)는 예시적인 애플리케이션(400)의 데이터 요소(405, 407) 둘다의 위치를 기술하고 있다. 이 인터페이스는 예시적인 애플리케이션(400)으로부터 전송된 커스텀 이벤트를 사용하지 않는다. 이 미디어 엔진은 예시적인 애플리케이션(400)으로부터의 커스텀 이벤트의 통지를 폐기 또는 무시할 수 있다.

[0097] 도 5a는 본 발명에 따른 애플리케이션(105)과 연관된 그래픽 인터페이스(120)를 생성하는 방법을 플로우차트(500)로 나타낸 것이다. 애플리케이션(105)이 장치 상에서 시동되어 미디어 엔진(110a, 110b, 110c, 110d)에 그의 ID를 통지한다. 미디어 엔진(110a)은 이 ID를 사용하여 템플릿 라이브러리에서 템플릿 파일을 찾는다(501). 템플릿 파일을 애플리케이션과 연관시키는 다른 방법은 당업자에게는 명백할 것이다. 미디어 엔진(110a, 110b, 110c, 110d)이 템플릿 파일이 발견되지 않은 것으로 판정하는 경우(510), 에러가 형성된다(512). 이 에러는 애플리케이션(105)으로 다시 보고될 수 있다.

[0098] 미디어 엔진(110a, 110b, 110c, 110d)은 또한 화면 상에 에러를 제공할 수 있다. 이것을 하는 한 옵션이 나타내어져 있다. 애플리케이션(105)은 에러에 응답하여 종료된다(513). 미디어 엔진(110a, 110b, 110c, 110d)은 라이브러리로부터 에러 템플릿을 로드하고 파싱한다(514). 이 에러 템플릿 파일은 정적 객체만을 기술하는 표준 그래픽 인터페이스일 수 있다, 즉 애플리케이션 데이터 요소(107) 또는 애플리케이션 커스텀 이벤트가 없다. 템플릿(113)은 렌더러(111)로 전달되고(540), 이 렌더러(111)는 이어서 템플릿(113)을 디스플레이(101)로 렌더링한다. 미디어 엔진은 애플리케이션이 종료된 것으로 판정하고(545) 단계(550)에서 처리를 종료한다.

[0099] 템플릿 파일이 발견되는 경우, 이는 파서(114)로 전달되고(515), 이 파서(114)는 템플릿 파일을 파싱하고 템플릿 정보를 템플릿(113)에 저장한다(520). 미디어 엔진(110a)은 이어서 데이터 요소(107)에 대한 갱신 또는 커스텀 이벤트의 통지를 수신하기 위해 기다린다(525). 데이터 요소(107) 갱신 또는 커스텀 이벤트 통지가 도착할 때, ID가 검사된다(527). 데이터 요소 또는 이벤트 ID가 템플릿(113)에서 발견되면, 처리는 단계(530)로 계속된다. ID가 템플릿(113)에서 발견되지 않는 경우에는, 처리가 단계(525)로 되돌아가서, 사실상 갱신 또는 통지를 폐기한다. 갱신된 정보 또는 통지 정보는 템플릿(113)을 갱신하는 데 사용된다(530). 이 경우에, 미디어 엔진(110a, 110b, 110c, 110d)은 그 정보를 템플릿(113)에 저장한다. 데이터 요소(107) 및 이벤트 정보를 템플릿(113)과 별도로 저장하는 것이 바람직할 수 있다. 이 경우에, 템플릿(113)을 갱신하지 않고, 미디어 엔진(110a, 110b, 110c, 110d)은 그 정보를 저장한다. 미디어 엔진(110a, 110b, 110c, 110d)이 애플리케이션(105)으로부터 수신된 모든 정보를 저장하는 경우, ID를 알기 위해 템플릿을 검사하는 단계(527)을 건너뛸 수 있다. 다른 대안으로서, 미디어 엔진이 템플릿(113)에 의해 사용되는 ID를 위해 데이터 요소 또는 이벤트만을 수신하는 경우 ID가 검사될 필요가 없을 수 있다. 애플리케이션(105)이 모든 정보를 다시 전달하게 할 필요없이 그래픽 인터페이스(120)의 전환을 가능하게 해주기 위해 이것이 사용될 수 있다. 미디어 엔진(110a, 110b, 110c, 110d)은 그래픽 인터페이스(120)가 갱신되어야만 하는지를 판정한다(535). 갱신되지 않아도 되는 경우, 처리는 단계(525)로 복귀한다. 갱신되어야만 하는 경우, 템플릿(113)이 렌더러로 전달된다(540). 앞서 기술한 바와 같이, 디스플레이 정보를 렌더러(111)로 전달하기 위한 기타 옵션들이 존재한다. 정보가 렌더러(111)로 전달된 후에, 미디어 엔진(110a, 110b, 110c, 110d)은 애플리케이션이 종료되었는지를 판정하고(545), 종료되지 않은 경우, 처리는 단계(525)로 되돌아간다. 애플리케이션이 종료된 경우, 처리가 종료된다(550).

[0100] 도 5b는 본 발명에 따른 애플리케이션(105)과 연관된 그래픽 인터페이스(120)를 생성하는 다른 방법을 플로우차트로 나타낸 것이다. 파서(114)는 템플릿 파일로부터 템플릿 정보를 파싱하고(565) 이를 템플릿(113)에 저장한다(570). 갱신된 데이터 요소 정보가 컨텐츠 상호작용 API를 사용함으로써 애플리케이션(105)으로부터 수신된다(575). 수신된 갱신 데이터 요소 정보로 템플릿(113)이 갱신된다(580). 데이터 요소 정보의 레이아웃이 레이아웃 관리자(118)를 사용하여 제어된다(585). 이것은 데이터 요소 문자열을 절단하는 것, 데이터 요소에 다른 문자들을 첨부하는 것, 요소의 디스플레이 특성을 변경하는 것, 또는 기타 수정을 포함할 수 있다. 템플릿(113)은 렌더러(111)로 전달되고(590) 렌더러는 그래픽 인터페이스(120)를 렌더링한다(595).

[0101] 스킨화 시스템은 장치(100) 상에 SVG 문서(115)를 렌더링하는 미디어 엔진(110a, 110b, 110c, 110d)을 포함한다. 미디어 엔진(110a, 110b, 110c, 110d)은 스킨화 API(또는 컨텐츠 상호작용 API)(112)를 포함한다. 스킨화 시스템은 또한 SVG 언어 문서(115) 및 SVG 트랜스코더 또는 파서(114)를 포함한다. 컨텐츠 개발자는 SVG 언어 및 SVG 트랜스코더를 사용하여 스킨화 문서를 템플릿 파일로서 생성한다. 애플리케이션 개발자는 미디어 엔진(110a, 110b, 110c, 110d) 및 스킨화 API(112)를 사용한다. 홈 스크린 애플리케이션 등의 장치 애플리케이션(105)은 그의 스킨 또는 그래픽 인터페이스(120)를 위해 미디어 엔진(110a) 및 스킨화 API(112)를 사용한다.

[0102] 본 발명의 일 실시예에서의 스킨화 시스템 컴포넌트들의 일례에 대해 이제부터 기술한다. 템플릿 파일의 구조를 설명하기 위해 홈 스크린 애플리케이션에 대한 스킨이 기술되어 있다. 이 예들은 미디어 엔진의 기능과 함께 템플릿 문서의 형식을 설명하는 데 사용된다.

[0103] 애플리케이션 및 시스템 데이터를 홈 스크린(Home Screen) 스킨에 통합

[0104] 투데이 스타일 홈 스크린은 애플리케이션 및 시스템 정보를 하나의 뷰에 통합한다. 이 정보는 홈 스크린 SVG 파일에 규정된 테마 설계자의 레이아웃에 따라 구성된다.

[0105] 투데이 홈 스크린 엘리먼트

[0106] 도 6은 본 발명의 일 실시예에 따른, 투데이 스타일 홈 스크린 뷰의 일례의 스크린샷을 나타낸 것이다.

[0107] 홈 스크린의 사용자 구성

[0108] 일련의 애플리케이션(및 연관된 데이터)이 테마에 포함되어 있는 SVG 파일의 요소들에 의해 결정된다. 양호하게는, 투데이 스타일 홈 스크린 상에 나타내어지는 고정된 일련의 애플리케이션의 사용자 구성은 이하의 것들을 포함한다.

[0109] • 도 6의 항목(26)의 표현은 메시지 상태 옵션(Message Status Options)으로부터 사용자 구성가능하다. 즉, 읽지 않은 것인지, 새로운 것인지 또는 아무것도 아닌지가 디스플레이된다.

[0110] • 도 6의 항목(13, 15, 19, 22)의 표현은 현재장소의 날짜 표현 및 시간 형식(Time Format)의 사용자 선택에 의존한다.

[0111] 애플리케이션 데이터를 SVG에 통합

[0112] 홈 스크린은 특수한 "데이터 요소"를 스킨 파일에 포함시키는 기능을 지원한다. 데이터 요소는 사용자 데이터 또는 시스템 정보를 위한 스킨에서의 플레이스 홀더(place holder)이다. 이것은 애플리케이션이 적절한 정보로 컨텐츠를 채울 수 있게 해주면서 스킨 설계자가 홈 스크린의 템플릿을 지정할 수 있게 해준다. 특정의 애플리케이션에 대한 데이터 요소들이 각각의 서브 기능(sub feature) 내에 정의된다. 스킨 설계자는 임의의 지원되는 SVG 구문을 통해 데이터 요소의 모습 및 거동을 제어할 수 있다. 홈 스크린 설계자가 스킨 내에 포함시킬 수 있는 일련의 데이터 요소는 총칭하여 "데이터 팔레트(data palette)"라고 한다. 예시적인 데이터 요소는 이하의 것을 포함한다.

[0113] <g id="hsl" font-family="BBMillbank" font-size="10">

[0114] <text id="emailltime" x="30" y="200"></text>

[0115] <text id="emaillFrom" x="50" y="200" font- style="Bold" ></text>

[0116] <text id="emaillSubject" x="100" y="200"x/text>

[0117] <set xlink : href="caret" attributeName="y"

[0118] begin="focusin" to="200" fill="freeze"/>

[0119] <loadScene begin= "activate" xlink : href="x- exec :

[0120] //LaunchInternal?id=email1 " />

[0121] </g>

[0122] email1Time, email1From, email1Subject는 홈 스크린 애플리케이션에 의해 애플리케이션(이 경우에 메시지) 관련 정보로 채워지는 데이터 요소들의 예이다. 각각의 애플리케이션의 데이터 요소들은 이 문서의 그들 각자의 서브섹션 내에 기술되어 있다.

[0123] 내비게이션

[0124] 홈 스크린 SVG 레이아웃은 사용자 선택가능한 핫스폿의 생성을 가능하게 해준다. 예를 들어, 스킨 설계자는 새로 수신된 마지막 이메일을 포함하는 행(도 6의 라벨(27, 28, 29))을 하이라이트하는 핫스폿을 생성할 수 있다. 사용자는 트랙휠을 굴림으로써 이 핫스폿에 포커싱할 수 있다. SVG 구문은 홈 스크린 설계자가 포커스 인(focus in), 포커스 아웃(focus out) 및 활성화(activate)(즉, 썸휠(thumbwheel)을 클릭하는 것)와 연관된 동작을 지정할 수 있게 해준다. 내비게이션 순서는 SVG에서 핫스폿이 정의되는 순서이다. SVG 핫스폿의 일례는 이하의 것을 포함한다.

[0125] <image id="caret" x="30" y="180" width="200"

[0126] height="30" xlink:href="caretBar.png"</image>

[0127] <g id="hsl" font-family="BBMillbank" font-size="10">

[0128] <text id="emailltime" x="30" y="200"></text>

[0129] <text id="email1Sender" x="50" y="200" font- style="Bold"></text>

[0130] <text id="email1Subject" x="100" y="200" </text>

[0131] <set xlink: href="caret" attributeName="y"

[0132] begin="focusin" to="200" fill="freeze"/>

[0133] <loadScene begin="activate" xlink:href="x-

[0134] exec : //LaunchInternal?email1"/>

[0135] </g>

[0136] 상기 예에서, focusin 및 activate 동작을 갖는 핫스폿이 제공된다. focusin <set>은 ^(caret) 이미지를 정확한 위치로 변환한다. <loadScene> 엘리먼트는 email1 데이터 요소에 대응하는 메시지를 연다.

[0137] 동적 레이아웃

[0138] 레이아웃은 변하는 길이를 가질 수 있는 텍스트를 포함하는 홈 스크린을 구현할 때 문제점을 제기한다. 이러한 문자열의 일례는 이메일 주제/보낸사람, 일정표 주제/위치, 기타 등등을 포함한다. 레이아웃 관리자는 홈 스크린 템플릿을 채우게 되는 데이터 요소의 동적 길이를 처리하는 데 사용된다.

[0139] 안타깝게도, SVG는 원래 이 기능을 제공하지 않는다. 데이터 요소들이 서로에 대해 어떻게 레이아웃되는지를 지정하는 것을 가능하게 해주는 이 언어에 대한 확장이 제공된다. 투데이 스타일 뷰에서, 이들 요건은 데이터 요소들의 열별 레이아웃(column-wise layout)을 요구한다. 예를 들어, 일정표 통합에 대한 요건은 다음과 같다.

[0140] • 주제(subject). 주제가 너무 길어서 이용가능한 공간에 맞지 않는 경우, 이는 "..."로 끝나게 된다.

[0141] • 장소(location). 장소가 너무 커서 이용가능한 공간에 맞지 않는 경우, 이는 "..."으로 끝나게 된다.

[0142] 양호하게는, 각각의 요소는 독립된 테마를 갖는 텍스트 스타일(independently themed text style)을 갖는다. 시간 및 주제 필드는 열 내에서 좌측 정렬된다(left aligned). 이 주제 필드는 최대폭을 부여받게 되며, 장소 필드는 주제 필드 바로 다음에 온다.

[0143] 상기한 것은 홈 스크린이 열별 레이아웃 관리자를 제공하는 것을 포함하며, 여기서 각각의 열은 이용가능한/최대 공간 및 너무 길어서 그 공간에 맞지 않는 경우 문자열을 절단하는 규칙을 지정한다. 필드가 "none"으로 설정되는 디스플레이 속성을 갖는 경우, 이는 차후의 텍스트 요소들을 레이아웃하기 위해 어떤 공간도 차지하지 않는다. 홈 스크린 스킨은 스킨 내의 데이터 요소들의 위치를 제어하는 레이아웃 관리자를 지정하는 기능을 갖는다. 예를 들어, <foreignObject> 엘리먼트는 스킨 파일 내의 레이아웃 규칙(layout rule) 및 참조 엘리먼트(references element)를 지정할 수 있다. 예를 들어,

[0144] <foreignObject>

[0145] <image x="81" y="216" width="230" height="20"

[0146] xlink:href="x-

[0147] object:/layout?ids=calendar2subject;calendar2location

[0148] &spacing=5&ellipsis=…"/>

[0149] </foreignObject/>

[0150] 다른 가능한 방법은 복합 문서 html 테이블 + svg를 사용하는 것이다.

[0151] 디폴트 애플리케이션 표현

[0152] 홈 스크린은 <image> 엘리먼트 또는 <text> 엘리먼트를 사용하여 홈 스크린 SVG 파일 내의 애플리케이션 기술(application description)을 참조하는 기능을 제공한다. 이들은 애플리케이션 이름을 지정하는 다른 방법이다. <text> 방법이 일반적으로 더 나은데 그 이유는 SVG 구문 및 특징을 이용할 수 있기 때문이다. <image> 방법은 레거시이다. 애플리케이션은 그의 모듈 이름 및 진입점을 xlink:href 속성의 "id" 파라미터의 값으로서 지정함으로써 식별된다. xlink:href 속성을 문자열 "x-object:/EntryDescription"으로 프리픽싱함으로써 애플리케이션 기술이 참조된다. 애플리케이션 기술이 ApplicationEntry 클래스로부터 검색된다. Phone 애플리케이션에 대한 애플리케이션 기술의 일례는 이하의 것을 포함한다.

[0153] <image x="32" y="65" width="276" height="24"

[0154] xlink:href="x-bject:/EntryDescription?font-

[0155] family=BBClarity&font-size=15&font-

[0156] style=bold&id=net_rim_bb_phone_app.Phone&align

[0157] =left&width=276&height=24&enclosing=()

[0158] &showname&showinfo">

[0159] <set attributeName= "visibility"

[0160] begin="hsl.focusin" to="visible" fill="freeze"/>

[0161] <set attributeName="visibility"

[0162] begin="hsl.focusout" to= "hidden" fill="freeze"/>

[0163] </image>

[0164] "calendar_app"라고 하는 애플리케이션 기술에 대한 <text> "데이터 엘리먼트"를 사용하는 대안의 표현은 다음과 같다.

[0165] <text id="calendar_app" x="32" y="180" style="font-

[0166] family:BBCondensed; font-size:12; font-weight :bold;

[0167] fill:#394142">

[0168] <set attributeName="fill" to="#FFFFFF"

[0169] begin="hs5.focusin" end="hs5. focusout"/>

[0170] </text>

[0171] 상기 예는 메시지 애플리케이션 기술의 포커싱된 상태인 이미지를 지정한다. 유의할 점은 포커스를 받고 해제하는 때 <set> 애니메이션이 <image> 엘리먼트의 가시성 속성(visibility attribute)을 토글시킨다는 것이다.

[0172] 이하는 EntryDescription에 대한 xlink:href url 파라미터의 요약 테이블의 일례이다. 이 요약 테이블은 애플리케이션 기술을 지정하는 <image> 방법에 적용된다. <text> 엘리먼트는 SVG의 모든 속성을 지원한다. <image> 메카니즘은 중첩하는 세트를 지원하지만 한 구현에서 <text>에 의해 지원되지 않고 <image>에 의해 지원되는 어떤 속성(즉, showInfo)이 있을 수 있다.

[0173]

이름	유형	값	디폴트
font-family	선택적	폰트 이름	시스템 디폴트
font-style	선택적	PLAIN, BOLD, ITALIC	PLAIN
font-size	필수적	정수	none
Foreground-color	선택적	HEX 형태의 정수	0=BLACK
id	필수적	애플리케이션 엔트리 이름	none
Align	선택적	좌측, 중앙, 우측	left
Width	필수적	정수	none
Height	필수적	정수	none
Enclosing	선택적	() 또는 {} 또는 []	공백 = 포함(enclosing) 없음
Showinfo	선택적	부울값	존재하는 경우, 애플리케이션 이름이 디스플레이됨
Showname	선택적	부울값	존재하는 경우, 읽지 않은 카운트(또는 새로운 카운트)가 디스플레이됨

[0174] 애플리케이션 아이콘

[0175] 홈 스크린은 <image> 엘리먼트를 사용하여 홈 스크린 SVG 파일 내에 애플리케이션 아이콘을 참조하는 기능을 제공한다. 애플리케이션 아이콘은 xlink:href 속성을 문자열 "x-object:/EntryIcon"으로 프리픽싱함으로써 참조된다. 이 애플리케이션은 그의 모듈 이름 및 진입점을 xlink:href 속성의 "id" 파라미터의 값으로서 지정함으로써 식별된다. 이 아이콘은 한 구현에서 RibbonIconField 클래스로부터 검색된다. SVG는 (하드코딩된) 스킨에 의해 제공되는 애플리케이션 아이콘을 참조할 수 있거나 애플리케이션 그자체에 의해 제공되는 아이콘을 참조할 수 있다. x-object:/EntryIcon xlink:href 애플리케이션은 애플리케이션에 의해 제공되는 애플리케이션 아이콘을 참조하기 위한 메카니즘을 제공한다. non x-object 메카니즘은 이미지 파일을 직접 참조한다, 즉 xlink:href="messageIcon.png"이다.

[0176] 포커싱된 표현

[0177] 애플리케이션 아이콘은 포커싱된(focused) 및 언포커싱된(unfocused) 표현을 가질 수 있다. 이것은 xlink:href 속성의 "focus" 파라미터의 존재에 의해 지정된다. 예를 들어,

[0178] <image x="-4" y="206 " width="48" height="36"

[0179] xlink:href="x-

[0180] object:/EntryIcon?focus& id=net.rim.ProfileHomeScreenApp&

[0181] size=36&width=48&height=36 "/>

[0182] "focus" 파라미터를 포함하지 않는 EntryIcon은 언포커싱된 상태로 렌더링된다. 예를 들어,

[0183] <image x="-4" y="206 " width="48" height="36"

[0184] xlink:href="x-

[0185] object:/EntryIcon?id=net.rim.ProfileHomeScreenApp&

[0186] size=36&width=48&height=36 "/>

[0187] 양호하게는, 포커싱된 및 언포커싱된 아이콘 표현의 가시도를 토글시키는 (즉, <animate> 또는 <set> 엘리먼트를 사용하여) 애니메이션을 정의하는 것은 SVG의 책임이다. 가시도, 불투명도 및 x,y 위치는 SVG를 통해 애니

메이트가능한 이러한 종류의 외래 객체 요소(foreign object element)의 속성이다.

[0188] 모든 애플리케이션은 그 애플리케이션에 사용되는 최소의 디폴트 시각 표현을 갖는다. 이 표현에 부가하여, 어떤 애플리케이션은 홈 스크린 등의 특정의 상황에서 또는, 예를 들어, 배너에서 사용하기 위한 대안의 표현을 정의한다. 메시지의 경우, 최소한, 이는 아이콘 및 이름을 제공한다. 이 디폴트 표현은 아이콘 격자를 사용하여 아이콘 테마에서 또 (BlackBerry Applications List 등의) 애플리케이션 리스트에서 사용된다. 한 테마에서, 메시지(Messages)는 홈 스크린 상에 카운트와 함께 표현될 수 있다. 배너에서, 이는 더 작은 엔벨로프 및 카운트를 제공하지만, 이름을 제공하지는 않는다. 다른 테마의 스킨에서, 이는 가장 최근의 새 이메일의 리스트를 제공할 수 있다. 이들 부가적인 표현이 아이콘 및 이름의 최소의 디폴트 표현을 넘어 정의된다. 디폴트 표현의 요소들은 테마에 의해 정의될 수 있거나 애플리케이션 자체 내에 리소스로서 정의될 수 있다. 디폴트 표현의 요소들의 일례는 다음과 같다.

[0189] • 이름: 테마가 있는 이름(themed name)은 애플리케이션에 의해 제공된 이름을 오버라이드(override)한다.

[0190] • 아이콘(선택적): 테마가 있는 이름은 애플리케이션에 의해 제공된 이름을 오버라이드한다. 어느 아이콘도 이용가능하지 않은 경우, 테마에 의해 제공된 디폴트 애플리케이션 아이콘이 사용된다.

[0191] • 아이콘 - 인-포커스(in-focus)(선택적): 이 아이콘은 아이콘이 포커싱되어 있을 때 사용된다. 어떤 아이콘도 이용가능하지 않은 경우, 통상의 아이콘이 사용된다.

[0192] • 아이콘 - 디스에이블됨(선택적): 이 아이콘은 애플리케이션이 가시적이지만 디스에이블되어 있을 때 사용된다. 예를 들어, 브라우저 서비스 북이 없는 경우, 디폴트 브라우저가 디스에이블된다. 이용가능한 아이콘이 없는 경우, 통상의 아이콘이 사용된다.

[0193] 읽지않은 카운트 표현

[0194] 홈 스크린 SVG 구문에 읽지않은 카운트를 지정하는 다른 방법들이 있다. 한 방법(방법 1)에서, foreignObject 메카니즘이 사용된다.

[0195] <image x="99" y="40" width="35" height="13"

[0196] xlink:href="x-object:/UnreadCound?type=email

[0197] &align=left&width=35&height="13""/>

[0198] 방법 1은 홈 스크린이 이메일에 대한 읽지않은 카운트를 내장할 수 있게 해준다. 애플리케이션 엔트리를 읽지않은 카운트와 연관시키면 읽지않은 카운트가 EntryDescription 또는 EntryIcon(열린 슬롯(open slot)임)과 연관되지 않도록 한다. 방법 1이 슬롯 유형 홈 스크린 설계에서 사용되지 않도록 한다. 슬롯 설계는 (Blackberry Applications List 등의) 애플리케이션 리스트에서의 애플리케이션들의 순서에 기초하여 동적으로 채워지는 애플리케이션 플레이스 홀더를 지정하는 기능을 참조한다.

[0199] 애플리케이션 이름에 읽지않은 카운트를 포함시키는 다른 외래 객체 방법(방법 2)에서,

[0200] <image x="32" y="65" width="276" heigth="24"

[0201] xlink:href="x-object:/EntryDescription?font-

[0202] family=BBClarity&font-size=15&font-

[0203] style=bold&id=net_rim_bb_phone_app.Phone&align

[0204] =left&width=276&heigth=24&enclosing=()

[0205] &showname&showinfo "></image>

[0206] 방법 2는 이러한 info가 애플리케이션에 의해 제공되는 경우 추가의 info를 렌더링하게 되는 "showinfo" 파라미터를 선택적으로 지정할 수 있다(3.12.1.6 참조). 애플리케이션이 제공하는 추가의 info는 테마 또는 SVG 스킨의 제어를 받지 않는다. 이 추가의 info는 통상적으로 읽지않은 또는 새로운 카운트이다. "showinfo" 파라미터가 지정되어 있는 경우, info는 설명의 바로 오른쪽에 디스플레이된다.

[0207] 예를 들어, 메시지 (1): 읽지않은 카운트가 0인 경우, 카운트 및 브레이스(brace)는 디스플레이되지 않는다.

어떤 상황에서, 읽지않은 카운트가 이름 옆에 디스플레이된다는 사실은 어떤 옵션을 통해 사용자에 의해 구성가능할 수 있다. 예를 들어, 특징 3.10의 구성(새 메시지 상태 옵션)은 카운트가 디스플레이되는지 여부는 물론 그의 의미(즉, 새로운 또는 읽지않은)를 결정한다.

[0208] 공지의 ID, 즉 데이터 요소를 갖는 <text> 및 <tspan> 엘리먼트를 사용하여 읽지않은 카운트를 스킨 내에 통합시키는 다른 가능한 방법이 있다. 이것은 우수한 방법일 수 있는데 그 이유는 읽지않은 카운트가 어떻게 렌더링되는지보다 홈 스크린 스킨에 더 많은 제어를 제공하기 때문이다. 이는 또한 카운트가 애니메이트될 수 있게 해준다. 예를 들어,

[0209] <text id="messagesName" font-name="BBCasual" font-

[0210] size="10" font-style="bold">

[0211] <tspan font-style="plain"> [</tspan>

[0212] <tspan id="messageslnfo" font-style="plain"></tspan>

[0213] <tspan font-style="plain">] </tspan>

[0214] </text>

[0215] 이것은 데이터 요소만(즉, id="messagesName" 및 id="messagesInfo")을 사용하여 애플리케이션 이름 및 읽지않은 카운트를 지정하는 비외래 객체(Non foreign object) 방법이고 SVG의 구문 및 특징들을 이용한다. 상기 예에서, id "messagesName" 및 "messagesInfo"는 홈 스크린 애플리케이션에 의해 인식되고, 적절한 info가 스킨 내에 치환된다.

[0216] 사용자 구성가능한 애플리케이션 슬롯

[0217] 홈 스크린 스킨은 리스트 내의 첫번째 요소인 0에서 시작하는 리스트에 그의 숫자 순서를 지정함으로써 애플리케이션 리스트(즉, Blackberry Applications List) 내의 애플리케이션을 참조할 수 있다. 예를 들어,

[0218] <image x="280" y="104" width="39" height="29"

[0219] xlink:href="x-object:

[0220] /EntryIcon?id=slot0&size=29&width=39&height=29">

[0221] </image>

[0222] 이 경우에, id=slot0는 스킨이 애플리케이션 리스트에서의 첫번째 애플리케이션을 참조하고 있다는 것을 지정한다.

[0223] 홈 스크린은 스킨 설계자가 애플리케이션 리스트로부터의 항목들을 홈 스크린 레이아웃 내에 통합시킬 수 있게 해준다. 스킨 설계자는 메뉴 항목 슬롯의 장소 및 화면의 내비게이션 순서 내에서의 그의 위치를 지정할 수 있게 해준다. 이들 슬롯을 차지하는 애플리케이션들은 사용자에 의해 구성가능하다. 디폴트 표현(default representation)을 확장하는 애플리케이션들의 경우, 디폴트 확장(default extension)도 역시 보여진다.

[0224] 홈 스크린 스킨에의 일정표 통합

[0225] 이 특징은 일정표 정보를 홈 스크린 SVG 스킨에 통합시키는 기능을 제공한다. 도 7a, 도 7b 및 도 7c는 본 발명의 일 실시예에 따른, 홈 스크린 스킨에의 일정표 통합의 일례의 스크린샷을 나타낸 것이다.

[0226] 일정표 이벤트 순서(Calendar Event Order)

[0227] 일정표에 대한 데이터 요소는 이하의 기준에 의해 정렬된 리스트에서의 그의 순서 위치(ordinal position)에 따라 일정표 이벤트를 참조한다.

[0228] 1: 시작 시간(가장 빠른 때부터 가장 늦은 때까지)

[0229] 2: 생성 시간(가장 빠른 때부터 가장 늦은 때까지)

[0230] 그에 부가하여, 리스트는 이하의 기준에 따라 필터링된다.

[0231] 1: 종료된 일정표 이벤트를 제외시킨다.

[0232] 리스트가 수정될 때마다 스킨이 갱신된다. 이하의 동작들 중 하나가 일어날 때 리스트가 갱신된다.

[0233] • 장치 날짜/시간이 사용자에 의해 수정될 때

[0234] • 장치 시간대가 변경될 때

[0235] • 장치가 부팅할 때

[0236] • 이벤트가 다른 일정표 이벤트에 의해 대체될 때

[0237] • 이벤트가 삭제될 때

[0238] • 이벤트가 갱신될 때

[0239] • 이벤트가 종료될 때

[0240] • 데이터 요소 테이블(Data Element Table)

[0241] 이하의 테이블은 홈 스크린 SVG 스킨에 포함될 수 있는 일정표에 관련된 데이터 요소들을 기술한다. 열에 대한 설명은 다음과 같다.

[0242] 엘리먼트 이름(element name): "id" 속성의 이름. 이것은 홈 스크린 스킨에 대한 엘리먼트를 식별해준다. 이들 이름은 대소문자구별이며 정확히 테이블에 나타낸 대로 지정되어야만 한다.

[0243] 엘리먼트 설명(element description): 엘리먼트를 채우게 되는 애플리케이션 데이터에 대한 설명.

[0244] 엘리먼트 유형(element type): 데이터 요소의 SVG 엘리먼트 유형. 데이터 요소는 지정된 유형으로 되어 있어야만 한다.

[0245]

엘리먼트 이름	엘리먼트 설명	엘리먼트 유형
calendar<n>time	일정표 이벤트의 리스트에서 n번째 일정표 이벤트의 시간/날짜. 시간은 날짜/시간 옵션 화면에서 "시간 형식" 설정에 따라 형식화된다. 이벤트가 (로컬 시간대에서) 오늘의 일정표에 있는 경우, 시간만이 보여진다. 이벤트가 (현재의 시간대에서) 장래의 일정표에 있는 경우, 날짜만이 보여진다. 날짜는 Ribbon.rrh 내의 DATE_FORMAT_STRING 키에 따라 net.rim.device.api.i18n.SimpleDateFormat으로 형식화된다.	<text>
calendar<n>subject	일정표 이벤트의 리스트에서 n번째 일정표 이벤트의 주제 라인. 너무 길어서 레이아웃 관리자에 의해 제공된 이용가능한 공간 내에 맞지 않는 경우, 텍스트가 생략부호로 절단된다.	<text>
calendar<n>location	리스트에서 n번째 일정표 이벤트의 장소. 너무 길어서 레이아웃 관리자에 의해 제공된 이용가능한 공간 내에 맞지 않는 경우, 텍스트가 생략부호로 절단된다.	<text>

[0246] 애플리케이션 진입점

[0247] 홈 스크린 SVG 구문은 <loadScene> 엘리먼트를 통해 일정표 이벤트에서의 n번째 일정표 항목을 기동시키는 기능을 제공한다. 예를 들어,

[0248] <loadScene begin="activate" xlink:href="x-

[0249] exec://LaunchInternal?calendar1/>

[0250] 상기 예는 일정표 이벤트 리스트에서 첫번째 일정표 이벤트를 연다. 홈 스크린 SVG 구문은 <loadScene> 엘리먼트를 통해 일정표 애플리케이션을 기동시키는 기능을 제공한다. 예를 들어,

[0251] <loadScene begin="activate" xlink:href="x-

[0252] exec://net_rim_bb_calendar_app.Calendar/>

[0253] 상기 예는 일정표 애플리케이션을 기동시킨다. 디폴트 뷰(default view)가 디스플레이된다.

[0254] 커스텀 이벤트

[0255] 홈 스크린 스킨은 데이터 팔레트에 기술된 데이터 요소에 관련된 커스텀 이벤트에 대한 애니메이션을 트리거하는 기능을 갖는다. 이것은 정보/상태가 갱신될 때 애니메이션 효과를 트리거하는 데 사용될 수 있다. 이하는 일정표에 대한 커스텀 이벤트의 예를 기술하는 테이블이다.

[0256]

이벤트 이름	이벤트 설명
Calendar<n>begin	n번째 일정표 이벤트가 시작되었음
Calendar<n>endpre	n번째 일정표 이벤트가 종료중임. 이 이벤트는 스킨 내의 데이터 요소가 n번째 일정표 이벤트의 종료의 결과로서 갱신되기 이전에 발생된다. 이 스킨은 전환 사라짐 효과(transition out effect)(예를 들어, 페이드 아웃(fade out) 또는 애니메이션 화면에서 사라짐(animation off the screen)) 등의 커스텀 애니메이션을 트리거할 기회를 갖는다. calendar<n>endpre 시에 스킨이 애니메이션을 정의하지 않는 경우, calendar<n>endpost(이하 참조)가 즉각적으로 발생된다. calendar<n>endpre 시에 스킨이 애니메이션을 정의하는 경우, 이는 calendar<n>endpost를 발생하기 이전에 이 이벤트 시에 시작하는 모든 애니메이션이 완료될 때까지 기다린다. 양호하게는, dur="indefinite"를 갖는 애니메이션이 정의되지 않으며, 그렇지 않은 경우 calendar<n>endpost는 결코 발생되지 않는다.
calendar<n>endpost	n번째 일정표 이벤트가 시작되었다. 이 이벤트는 일정표(데이터 요소 테이블 참조)에 관련된 모든 데이터 요소가 갱신된 후에 발생된다.

[0257] 이하의 SVG는 calendar1 이벤트에 대해 데이터가 갱신되기 전후에 페이드 아웃 및 페이드 인 효과를 어떻게 달성하는지를 나타낸 것이다.

[0258] <g id="hsl" font-famiIy="BBMillbank" font-size="10">

[0259] <text id="calendarltime" x="30" y="200"></text>

[0260] <text id="calendarlSubject" x="50" y="200" font-

[0261] style="Bold" ></text>

[0262] <text id="calendarlLocation" x="100" y="200" </text>

[0263] <animate attibuteName="fill-opacity" to="0"

[0264] dur="2s" begin="calendarlEndPre" fill="freeze" />

[0265] <animate attributeName="fill-opacity" to="1"

[0266] dur="2s" begin="calendarlEndPost" fill="freeze"/>

[0267] </g>

[0268] 양호하게는, 이벤트가 일어나게 되어 있었을 때 어떤 이유로 장치가 오프된 경우 스킨을 올바른 상태로 되게 하는 메카니즘도 있다.

[0269] 일정표 상태는 오늘 날짜의 앞으로 있게 될 일정표 이벤트 중 1개 내지 n개를 나타내는 기능을 포함하게 된다. 이들 이벤트를 클릭하면 그 이벤트를 열거나 요건(10)이 지원되는 경우 컨텍스트 관련 메뉴가 뜨게 한다. 이들 이벤트는 이하의 것을 포함한다.

[0270] • 이벤트가 오늘 예정되어 있는 경우, 로컬화된 시간 형식을 사용하여 시간이 보여진다.

[0271] • 이벤트가 오늘을 지나 예정되어 있는 경우, 로컬화된 MM/DD 형식을 사용하여 날짜가 보여진다.

[0272] • 주제. 주제가 너무 길어 이용가능한 공간에 맞지 않는 경우, 이는 "..."으로 끝나게 된다.

[0273] • 장소. 장소가 너무 길어 이용가능한 공간에 맞지 않는 경우, 이는 "..."으로 끝나게 된다.

[0274] 각각의 요소는 독립적인 테마가 있는 텍스트 스타일을 갖는다. 시간 및 주제 필드는 열 내에서 왼쪽 정렬된다. 주제 필드는 최대 폭을 부여받으며, 장소 필드는 주제 필드 바로 다음에 온다.

[0275] 양호하게는, 회의가 끝날 때 일정표 이벤트가 리스트로부터 제거된다. 예외는 하루 종일 걸리는 이벤트(all

day event)이며, 이 이벤트를 리스트로부터 제거하게 될 회의의 시작 1시간 전에 그 이벤트가 제거된다.

[0276] 홈 스크린 스킨에의 전화 통합

[0277] 도 8a 및 도 8b는 본 발명의 일 실시예에 따른, 홈 스크린 스킨에의 전화 통합의 예들의 스크린샷을 나타낸 것이다.

[0278] 데이터 요소 테이블

[0279]

엘리먼트 이름	엘리먼트 설명	홈 스크린 스킨 SVG 엘리먼트 유형
missedcalls<n>time	n번째 마지막 연결실패된 호출의 시간. 이 시간은 날짜/시간 옵션 화면에서 "시간 형식" 설정에 따라 형식화된다. 이벤트가 (로컬 시간대에서) 오늘 일정표에 있는 경우, 시간만이 보여진다. 이벤트가 (현재 시간대에서) 과거 일정표에 있는 경우, 날짜만이 보여진다. 양호하게는, 날짜는 Ribbon.rrh 내의 DATE_FORMAT_STRING 키에 따라 net.rim.device.api.i18n.SimpleDateFormat로 형식화된다.	<text>
missedcalls<n>info	n번째 연결실패된 호출의 정보. 예를 들어(이름 또는 전화번호 또는 미지의 번호), 전화 번호에 대해 주소 엔트리가 있는 경우, 이는 호출자의 이름을 보여준다. 이름이 너무 길어 레이아웃 관리자에 의해 정의된 이용가능한 공간에 맞지 않는 경우, 이는 "..."으로 끝난다.	<text>

[0280] 대안의 애플리케이션 표현

[0281] 홈 스크린 SVG는 애플리케이션에 대한 아이콘 및 텍스트 표현의 통합을 지원한다. 이것은 EntryIcon 및 EntryDescription <image>를 지정함으로써 달성된다. 전화에 대한 EntryIcon 및 EntryDescription은 새로운 연결실패된 호출이 있는지 여부에 따라 대안의 표현을 디스플레이한다. 새로운 연결실패된 호출이 있는 경우, EntryDescription은 "연결실패된 호출"을 디스플레이하는 반면, 새로운 연결실패된 호출이 없는 경우, 이는 "전화"를 디스플레이한다. EntryIcon은 연결실패된 호출이 있는지 여부에 따라 다른 아이콘을 디스플레이한다.

[0282] 애플리케이션 진입점

[0283] 홈 스크린 SVG 구문은 또한 전화 애플리케이션을 직접 <loadScene> 엘리먼트를 통해 기동시키는 기동도 제공한다. 예를 들어,

[0284] <loadScene begin="activate" xlink:href="x-

[0285] exec://net_rim_bb_phone_app.Phone/>

[0286] 이것은 상기한 요건 a), 즉 애플리케이션이 GUI로부터의 사용자 입력에 반응할 수 있어야 한다는 것에 관련되어 있다. 이 기능은 애플리케이션이 SVG 엘리먼트를 클릭함으로써 이벤트를 등록할 수 있게 해준다. 애플리케이션은 그 자신을 미디어 엔진 API에 MediaListener로서 등록한다. 이벤트가 <loadScene> 엘리먼트에 의해 발생될 때, 등록된 MediaListener 구현으로 통지가 이루어진다. 이 구현은 어느 엘리먼트가 클릭되었는지를 판정하고 적절한 조치를 취하기 위해 코드가 사용할 수 있는 xlink:href URI를 포함하는 통지를 수신한다.

[0287] SVG가 통지를 등록하는 메카니즘의 다른 예는 <loadScene> 엘리먼트를 사용하지 않는다. 애플리케이션은 activate 이벤트(activate event)로 기동되는 <loadScene> 엘리먼트보다 직접 엘리먼트에 대해 activate 이벤트가 있는지 리스닝한다. 이 엘리먼트가 포커싱가능한 한, activate 이벤트가 발생된다. 포커싱가능한 엘리먼트는 그 엘리먼트에 대해 정의된 infocus, outfocus 또는 activate <animate>를 갖는 것이다. 예를 들어,

[0288] <image id="myImage" x="110" y="19" width="20"

[0289] height="18" xlink:href="myImage.png">

[0290] <!-- dummy <set>: 이 노드의 begin 조건은 외부 리스너가 activate 이벤트를 리스닝할 수 있게 해준다. (SVG 내의 어느 것도 activate 이벤트에 의존하지 않는 경우, 이들이 처리되거나 외부 리스너로 발생되지 않는다) -->

[0291] <set attributeName="visibility"

[0292] to="visible" begin="myImage.activate"/>

[0293] </image>

[0294] 임의적인 키 이벤트(key event)에 연계되어 있는 전화 기동 동작(Launch Phone action)의 지정: 홈 스크린은 액세스 키(access key)에 연계되어 있는 진입점을 정의하는 기능을 제공한다. 예를 들어,

[0295] <loadScene begin="foo.accessKey("send")"

[0296] xlink:href="x-exec://net_rim_bb_phone_app.Phone/>

[0297] 이 예에서, "foo"는 loadScene 이벤트를 트리거하기 위해 포커싱되어 있어야만 하는 엘리먼트의 이름이다. accessKey("send")는 동작을 트리거 또는 시작하는 키 이벤트이다. 이 예는 기본적으로 ""foo"라는 이름의 엘리먼트가 포커싱되어 있고 send 키가 눌러질 때 전화 애플리케이션을 기동시킨다"는 것이다.

[0298] 선택적인 다이얼 번호 파라미터로 전화 기동(Launch Phone) 동작을 지정: 전화 애플리케이션 진입점은 다이얼할 명령(command to dial) 및 그에 뒤따르는 다이얼할 번호(number to dial)인 파라미터를 받을 수 있다. 예를 들어,

[0299] <loadScene begin="foo.accessKey("send")" xlink:href="x-

[0300] exec://net_rim_bb_phone_app.Phone?command=dial:<miss edcallsl>/>

[0301] <loadScene begin="foo.accessKey("send")" xlink:href="x-

[0302] exec://dialselected"/>

[0303] 이 예에서, command=dial:<number>은 loadScene 동작이 시작할 때 전화 애플리케이션으로 전달되는 동작이다. 홈 스크린 애플리케이션은 전화 애플리케이션으로 전달되기 이전에 <missedcallsl>을 적절한 번호로 치환해야만 하는데, 그 이유는 항 "missedcallsl"이 전화 애플리케이션이 모르는 홈 스크린 구성자(home screen construct)이기 때문이다. 이 예는 "foo"라는 이름의 엘리먼트가 포커싱되어 있고 send 키 이벤트가 눌러질 때 전화 애플리케이션을 기동시키고 <missedcallsl>와 일치하는 번호를 다이얼링함"을 말하고 있다.

[0304] 호출 로그 엔트리 컨텍스트 메뉴 기동(Launch Call Log Entry Context Menu) 동작을 지정: 홈 스크린 SVG 구문은 연결실패된 호출 데이터 요소에 대응하는 호출 로그 엔트리(call log entry)에 대한 팝업 메뉴를 기동시키는 기능을 제공한다. 이 거동은 홈 스크린 SVG 파일 내의 <loadScene> 엘리먼트를 통해 지정될 수 있다. 예를 들어,

[0305] <loadScene begin="activate" xlink:href="x-

[0306] exec://LaunchInternal?missedcallsl />

[0307] 홈 스크린에의 SMS 및 MMS 통합

[0308] 도 9는 본 발명의 일 실시예에 따른, 홈 스크린에의 SMS 및 MMS 통합의 일례의 스크린샷을 나타낸 것이다.

[0309] 데이터 요소 테이블

[0310]

엘리먼트 이름	엘리먼트 설명	엘리먼트 유형
sms<n>time	n번째 마지막 새로운 sms 또는 mms 메시지의 시간. 이 시간은 날짜/시간 옵션 화면에서의 "시간 형식" 설정에 따라 형식화된다. (로컬 시간대에서) 오늘 일정표에 메시지가 수신된 경우, 시간만이 보여진다. (현재 시간대에서) 과거 일정표에서 이메일이 수신된 경우, 날짜만이 보여진다. 날짜는 Ribbon.rrh에서의 DATA_FORMAT_STRING 키에 따라 net.rim.device.api.i18n.SimpleDateFormat으로 형식화된다. "Display Time"이 Messages Options->General Options에서 no로 설정되어 있는 경우, 이 필드는 디스플레이되지 않는다. 즉, 그의 "display" 속성이 "none"으로 설정되어 있다.	<text>

sms<n>from	n번째 마지막 새로운 sms 또는 mms 메시지의 보낸사람. 이메일 주소에 대해 주소록 엔트리가 존재하는 경우, 연락처 이름이 그 대신에 보여진다. 연락처 이름 또는 이메일 주소가 너무 커서 이용가능한 공간에 맞지 않는 경우, 이는 "..."으로 끝난다. "Display Name"이 Messages Options->General Options에서 no로 설정되어 있는 경우, 이 필드는 디스플레이되지 않는다. 즉, 그의 "display" 속성이 "none"으로 설정되어 있다.	<text>
sms<n>body	n번째 마지막 새로운 이메일의 본문(body). 본문이 너무 커서 이용가능한 공간에 맞지 않는 경우, 이는 "..."으로 끝난다. 본문이 비어 있는 경우, 아무것도 디스플레이되지 않는다.	<text>

[0311] 애플리케이션 진입점

[0312] 홈 스크린 SVG 구문이 데이터 요소와 연관된 메시지를 여는 <loadScene> 엘리먼트를 정의하는 기능을 제공한다. 메시지를 여는 SVG 구문:

[0313] <loadScene begins="activate" xlink:href="x-

[0314] exec://LaunchInternal?sms1/>

[0315] 상기 예는 데이터 요소 테이블로부터의 sms1과 연관된 sms 또는 mms 메시지를 연다.

[0316] 홈 스크린 SVG 구문은 결합된 sms 및 mms 받은편지함을 여는 <loadScene> 엘리먼트를 정의하는 기능을 제공한다. 이는 모듈 이름 및 진입점 이름을 지정함으로써 이것을 한다. 메시지 애플리케이션을 여는 SVG 구문:

[0317] <loadScene begin="activate" xlink:href="x-exec: //

[0318] net_rim_bb_messaging_app.sms_and_mms"/>

[0319] 새로운/읽지않은 메시지 카운트

[0320] 결합된 SMS 및 MMS 진입점과 연관된 카운트는 메시지 상태 옵션 화면을 통해 구성가능하다. 텍스트 스타일 및 닫기 괄호(enclosing embrace)는 홈 스크린 SVG에서 구성가능하다.

[0321] 애플리케이션 구성의 홈 스크린 영역

[0322] 홈 스크린 SVG 스킨 파일은 기술된 바와 같이 애플리케이션에 대한 "슬롯"을 지정할 수 있다. 몇개의 슬롯이 이용가능하고 어느 애플리케이션이 이들을 차지하는지를 나타내는 (테마 영향을 받는(themeable)) 영역이 애플리케이션 구성(Organize Applications) 화면 상단에 렌더링된다. 이 영역은 "홈 스크린(Home Screen)"이라는 타이틀을 갖는다. 이 영역의 크기는 홈 스크린 스킨에서 이용가능한 슬롯의 수에 의존한다. 사용자는 애플리케이션 구성에서 통상의 이동(move) 동작을 통해 애플리케이션들을 이 영역 안으로 또는 그 밖으로 이동시킬 수 있다. 홈 스크린 스킨에서 고정된 애플리케이션에 할당된 슬롯은 리스트에 나타나지 않는다. 양호하게는, 테마 개발자는 고정된 슬롯에 할당되는 애플리케이션이 홈 스크린(Home Screen) 영역 내의 위치를 차지하도록 하는 방식으로 이들에 우선순위를 부여하지 않도록 주의한다. 예를 들어, "메시지들"이 홈 스크린 상의 고정된 위치를 차지하는 경우, 테마는 비어 있는 슬롯 중 어느 것도 차지하지 않도록 메시지의 위치를 설정해야만 한다. 이것은 메시지가 홈 스크린 상에 두번 나타나는 것을 방지한다.

[0323] 도 10은 본 발명의 일 실시예에 따른, 애플리케이션 구성(Organize Applications) 내부의 홈 스크린(Home Screen) 영역의 개념을 나타낸 것이다. 연한 회색의 영역은 홈 스크린 상에서 이용가능한 슬롯들 및 현재 이들을 차지하는 애플리케이션을 나타낸다.

[0324] 홈 스크린(Home Screen) 영역 내의 위치들을 차지하는 애플리케이션은 또한 애플리케이션 리스트 팝업에서 상단 위치들을 차지한다. 홈 스크린 내의 슬롯들을 차지하는 애플리케이션들은 또한 애플리케이션 리스트로부터 액세스될 수 있어야만 하는데, 그 이유는 이 팝업이 편의키 매핑(convenience key mapping)을 통해 임의의 애플리케이션 내로부터 기동될 수 있기 때문이다.

[0325] 양호하게는, 숨겨져 있는 애플리케이션은 애플리케이션 구성(Organize Applications)의 홈 스크린(Home Screen) 영역 내의 위치로 이동가능하지 않다. 이것은 숨겨져 있는 애플리케이션이 홈 스크린 상의 슬롯을 차지하는 것을 방지한다. 이와 마찬가지로, 애플리케이션 구성(Organize Applications)의 홈 스크린(Home Screen) 영역

내의 위치를 차지하는 애플리케이션은 숨겨질 수 없다.

[0326] 홈 스크린 스킨이 기술한 바와 같이 구성가능한 애플리케이션 슬롯들을 포함하는 경우, 이들은 애플리케이션 구성 화면을 통해 구성된다. 홈 스크린 애플리케이션 슬롯들의 컨텐츠는 애플리케이션 메뉴 순서의 상단에 있는 항목들에 기초한다. 5개의 슬롯이 있는 경우, 리스트의 상단에 있는 5개의 애플리케이션이 이들 슬롯 내에 채워진다. 애플리케이션 구성 화면은 이용가능한 슬롯들을 채우는 아이콘들 아래에 테마 영향을 받는 영역을 드로잉함으로써 애플리케이션들 중 몇개가 홈 스크린 상에 나타나는지를 나타낸다. 이 영역은 상단에 "홈 스크린(Home Screen)"이라고 되어 있는 읽기 전용 라벨을 갖는다. 따라서, 스킨이 2개의 슬롯을 갖는 경우, 상단의 2개의 아이콘이 이 영역 상부에 나타난다. 그 스킨이 5개의 슬롯을 갖는 경우, 상위 5개가 이 영역 상부에 나타난다.

[0327] 도 11은 기술된 바와 같은 메시지 및 구성 시스템 함수 및 홈 스크린 GUI를 갖는 전자 장치로서 사용될 수 있는, 일 실시예에 따라 구성된 양호한 핸드헬드 장치(1002)의 일례를 상세하게 나타낸 블록도이다. 핸드헬드 장치(1002)는 양호하게는, 다른 컴퓨터 시스템과 통신하는 기능을 비롯한, 적어도 음성 및 향상된 데이터 통신 기능을 갖는 양방향 통신 장치이다. 핸드헬드 장치(1002)에 의해 제공되는 기능에 따라, 이는 데이터 메시징 장치, 양방향 페이저, 데이터 메시징 기능을 갖는 셀룰러 전화, 무선 인터넷 가전 기기, 또는 (전화 기능을 갖거나 갖지 않는) 데이터 통신 장치라고 할 수 있다. 핸드헬드 장치(1002)는 그의 지리적 통화권 영역 내의 복수의 기지국 송수신기 시스템(도시 생략) 중 임의의 하나와 통신할 수 있다.

[0328] 핸드헬드 장치(1002)는 통상적으로 수신기(1012), 송신기(1014), 및 하나 이상의 (양호하게는 매립형(embedded) 또는 내장형(internal)) 안테나 요소(1016, 1018) 등의 관련 컴포넌트를 포함하는 통신 서브시스템(1011)을 포함한다. 통신 분야의 당업자에게는 명백한 바와 같이, 통신 서브시스템(1011)의 특정의 설계는 핸드헬드 장치(1002)가 동작하기로 되어 있는 통신 네트워크에 의존한다.

[0329] 핸드헬드 장치(1002)는, 요구된 네트워크 등록 또는 활성화 절차가 완료된 후에, 네트워크를 통해 통신 신호를 전송 및 수신할 수 있다. 네트워크를 통해 안테나(1016)에 의해 수신된 신호는 수신기(1012)에 입력되며, 수신기(1012)는 신호 증폭, 주파수 다운 컨버전, 필터링, 채널 선택, 및 아날로그-디지털(A/D) 변환 등의 통상의 수신기 기능을 수행할 수 있다. 수신된 신호의 A/D 변환은 복조 및 디코딩 등의 더 복잡한 통신 기능이 DSP(1020)에서 수행될 수 있게 해준다. 유사한 방식으로, 예를 들어, DSP(1020)에 의해 변조 및 인코딩을 비롯하여 전송될 신호가 처리된다. 이들 DSP-처리된 신호들은 디지털-아날로그(D/A) 변환, 주파수 업 컨버전, 필터링, 증폭 및 안테나(1018)를 거쳐 통신 네트워크를 통한 전송을 위해 송신기(1014)에 입력된다. DSP(1020)는 통신 신호를 처리할 뿐만 아니라, 수신기 및 송신기 제어를 제공한다. 예를 들어, 수신기(1012) 및 송신기(1014)에서 통신 신호에 적용되는 이득은 DSP(1020)에 구현되는 자동 이득 제어 알고리즘을 통해 적응적으로 제어될 수 있다.

[0330] 네트워크 액세스는 핸드헬드 장치(1002)의 가입자 또는 사용자와 연관되어 있으며, 따라서 핸드헬드 장치(1002)는 메모리 모듈(1062), 메모리 모듈 카드 또는 네트워크에서 동작하기 위해 인터페이스(1064)에 삽입되거나 연결되는 R-UIM(Removable User Identity Module)을 포함한다. 다른 대안으로서, 메모리 모듈(1062)은 이동국(1002)이 네트워크에서 동작할 수 있도록 서비스 제공자에 의해 구성 데이터로 프로그램되는 비휘발성 메모리일 수 있다. 핸드헬드 장치(1002)가 모바일 배터리-전원 장치이기 때문에, 이는 또한 하나 이상의 충전가능 배터리(1056)를 받아들이는 배터리 인터페이스(1054)를 포함한다. 이러한 배터리(1056)는 핸드헬드 장치(1002) 내의 대부분의 전기 회로(전부가 아닐 수 있음)에 전기 전력을 제공하며, 배터리 인터페이스(1054)는 그에 대한 기계적 및 전기적 연결을 제공한다. 배터리 인터페이스(1054)는 그 회로 전부에 전력 V+를 제공하는 레귤레이터(regulator)에 연결되어 있다.

[0331] 핸드헬드 장치(1002)는 이동국(1002)의 전체적인 동작을 제어하는 마이크로프로세서(1038)를 포함한다. 적어도 데이터 및 음성 통신을 비롯한 통신 기능은 통신 서브시스템(1011)을 통해 수행된다. 마이크로프로세서(1038)는 또한 디스플레이(1022), 플래쉬 메모리(1024), 랜덤 액세스 메모리(RAM)(1026), 보조 입/출력(I/O) 서브시스템(1028), 직렬 포트(1030), 키보드(1032), 스피커(1034), 마이크로폰(1036), 단거리 통신 서브시스템(1040), 및 전체적으로 1042로 표시되어 있는 임의의 다른 장치 서브시스템 등의 부가적인 장치 서브시스템과 상호작용한다. 도시된 서브시스템들 중 어떤 것은 통신-관련 기능을 수행하는 반면, 다른 서브시스템들은 "상주(resident)" 또는 온-디바이스(on-device) 기능을 제공할 수 있다. 주목할 만한 것은, 예를 들어, 키보드(1032) 및 디스플레이(1022) 등의 어떤 서브시스템들은 통신 네트워크를 통해 전송하기 위한 텍스트 메시지를 입력하는 것 등의 통신-관련 기능 및 계산기 또는 작업 리스트 등의 장치-상주 기능 둘다를 위해 사용될 수 있

다. 마이크로프로세서(1038)에 의해 사용되는 오퍼레이팅 시스템 소프트웨어는 양호하게는 플래쉬 메모리(1024) 등의 영구적 저장소(다른 대안으로서, 판독 전용 메모리(ROM) 또는 유사한 저장 요소(도시 생략)일 수 있음)에 저장된다. 당업자라면 오퍼레이팅 시스템, 특정의 장치 애플리케이션, 또는 그의 일부분이 RAM(1026) 등의 휘발성 저장소 내에 일시적으로 로드될 수 있다는 것을 잘 알 것이다.

[0332] 마이크로프로세서(1038)는, 그의 오퍼레이팅 시스템 기능에 부가하여, 양호하게는 핸드헬드 장치(1002) 상에서 소프트웨어 애플리케이션을 실행하는 것을 가능하게 해준다. 적어도 데이터 및 음성 통신 애플리케이션을 비롯한, 기본적인 장치 동작을 제어하는 미리 정해진 일련의 애플리케이션은 통상적으로 제조 동안에 핸드헬드 장치(1002) 상에 설치된다. 핸드헬드 장치(1002) 상에 로드될 수 있는 양호한 애플리케이션은 이메일, 일정표 이벤트, 음성 메일, 약속, 및 작업 항목(이에 한정되지 않음) 등의 사용자에 관한 데이터 항목을 구성 및 관리하는 기능을 갖는 PIM(personal information manager, 개인 정보 관리자) 애플리케이션일 수 있다. 자연스럽게, PIM 데이터 항목 및 기타 정보의 저장을 용이하게 해주기 위해 핸드헬드 장치(1002) 및 메모리 모듈(1062)에서 하나 이상의 메모리 저장소가 이용가능하다.

[0333] PIM 애플리케이션은 양호하게는 무선 네트워크를 통해 데이터 항목을 전송 및 수신하는 기능을 가지고 있다. 양호한 실시예에서, PIM 데이터 항목은 무선 네트워크를 통해 매끄럽게 통합, 동기화 및 갱신되고, 이동국 사용자의 대응하는 데이터 항목이 저장되고 및/또는 호스트 컴퓨터 시스템과 연관됨으로써 이러한 항목들과 관련하여 핸드헬드 장치(1002) 상에 미러링된 호스트 컴퓨터를 생성한다. 이것은 호스트 컴퓨터 시스템이 이동국 사용자의 사무실 또는 기업 컴퓨터 시스템인 경우에 특히 유익하다. 부가적인 애플리케이션이 또한 네트워크, 보조 I/O 서브시스템(1028), 직렬 포트(1030), 단거리 통신 서브시스템(1040), 또는 임의의 다른 적당한 서브시스템(1042)을 통해 핸드헬드 장치(1002) 상에 로드되고 마이크로프로세서(1038)에 의해 실행하기 위해 RAM(1026) 또는 양호하게는 비휘발성 저장소(도시 생략)에 설치될 수 있다. 애플리케이션 설치에 있어서의 이러한 유연성은 핸드헬드 장치(1002)의 기능을 향상시키고 향상된 온-디바이스 기능, 통신-관련 기능, 또는 둘다를 제공할 수 있다. 예를 들어, 보안 통신 애플리케이션은 전자 상거래 기능 및 기타 이러한 금융 거래가 핸드헬드 장치(1002)를 사용하여 수행될 수 있게 해준다.

[0334] 데이터 통신 모드에서, 텍스트 메시지, 이메일 메시지, 또는 웹 페이지 다운로드 등의 수신된 신호는 통신 서브시스템(1011)에 의해 처리되어 마이크로프로세서(1038)에 입력된다. 마이크로프로세서(1038)는 양호하게는 디스플레이(1022)로 또는 다른 대안으로서 보조 I/O 장치(1028)로 출력하기 위해 신호를 추가적으로 처리한다. 핸드헬드 장치(1002)의 사용자는 또한 디스플레이(1022) 및 아마도 보조 I/O 장치(1028)와 함께 키보드(1032)를 사용하여, 예를 들어, 이메일 메시지 등의 데이터 항목을 작성할 수 있다. 키보드(1032)는 양호하게는 완전한 영숫자 키보드 및/또는 전화형 키패드이다. 이들 작성된 항목은 통신 서브시스템(1011)을 통해 통신 네트워크를 거쳐 전송될 수 있다.

[0335] 음성 통신의 경우, 핸드헬드 장치(1002)의 전체적인 동작은, 수신된 신호가 스피커(1034)로 출력되고 전송을 위한 신호가 마이크로폰(1036)에 의해 발생되는 것을 제외하고는, 실질적으로 유사하다. 음성 메시지 레코딩 서브시스템 등의 대안의 음성 또는 오디오 I/O 서브시스템도 구현될 수 있다. 음성 또는 오디오 신호 출력이 양호하게는 주로 스피커(1034)를 통해 달성되지만, 어떤 예로서, 호출하는 당사자의 ID의 표시, 음성 통화의 기간, 또는 기타 음성 통화 관련 정보를 제공하기 위해 디스플레이(1022)도 사용될 수 있다.

[0336] 도 11의 직렬 포트(1030)는 통상 선택적이지만 바람직한 컴포넌트로서 사용자의 데스크톱 컴퓨터와 동기화하기 위해 PDA(personal digital assistant)-유형 통신 장치에서 구현된다. 직렬 포트(1030)는 사용자가 외부 장치 또는 소프트웨어 애플리케이션을 통해 환경 설정(preference)을 설정할 수 있게 해주며 무선 통신 네트워크 이외의 것을 통해 핸드헬드 장치(1002)로의 정보 또는 소프트웨어 다운로드를 제공함으로써 핸드헬드 장치(1002)의 기능을 확장시킨다. 대안의 다운로드 경로가, 예를 들어, 직접 연결, 따라서 신뢰성있는 신뢰 연결을 통해 암호화 키를 핸드헬드 장치(1002)로 로드함으로써 안전한 장치 통신을 제공하는 데 사용될 수 있다.

[0337] 단거리 통신 서브시스템(1040)은 핸드헬드 장치(1002)와 다른 시스템 또는 장치(꼭 유사한 장치일 필요는 없음) 간의 통신을 제공하는 부가의 선택적인 컴포넌트이다. 예를 들어, 서브시스템(1040)은 유사한 기능의 시스템 및 장치와의 통신을 제공하기 위해 적외선 장치 및 관련 회로와 컴포넌트, 또는 블루투스™ 통신 모듈을 포함할 수 있다. 블루투스™는 Bluetooth SIG, Inc.의 등록 상표이다.

[0338] 핸드헬드 장치(1002)는 상기한 바와 같이 GUI에 홈 스크린 통합된 정보 제공(home screen integrated presentation of information)을 제공함으로써 소프트웨어(명령어 및 데이터) 등을 통해 구성될 수 있다.

[0339] 본 발명에 따른 시스템 및 방법은 상기한 기능들을 갖는 임의의 하드웨어, 소프트웨어 또는 하드웨어와 소프트웨어의 조합에 의해 구현될 수 있다. 소프트웨어 코드는 그 전체가 또는 그 일부분이 컴퓨터 판독가능 메모리에 저장될 수 있다. 게다가, 반송파에 임베딩될 수 있는 소프트웨어 코드를 나타내는 컴퓨터 데이터 신호가 통신 네트워크를 통해 전송될 수 있다. 이러한 컴퓨터-판독가능 메모리 및 컴퓨터 데이터 신호는 물론 하드웨어, 소프트웨어 및 이들의 조합도 본 특허 내용의 범위 내에 속한다.

산업상 이용 가능성

[0340] 본 발명의 특정의 실시예들이 도시되고 기술되어 있지만, 본 발명의 진정한 범위를 벗어나지 않고 이러한 실시예들에 여러 변경 및 수정이 행해질 수 있다.

도면의 간단한 설명

[0013] 도 1은 본 발명의 일 실시예에 따른, 모바일 장치의 일례의 기능을 개략적으로 나타낸 도면.

[0014] 도 2는 본 발명의 일 실시예에 따른, 애플리케이션의 일례의 기능을 개략적으로 나타낸 도면.

[0015] 도 3a는 본 발명의 일 실시예에 따른, 미디어 엔진의 일례의 기능을 개략적으로 나타낸 도면.

[0016] 도 3b는 본 발명의 일 실시예에 따른, 미디어 엔진의 다른 예의 기능을 개략적으로 나타낸 도면.

[0017] 도 3c는 본 발명의 일 실시예에 따른, 미디어 엔진의 다른 예의 기능을 개략적으로 나타낸 도면.

[0018] 도 3d는 본 발명의 일 실시예에 따른, 미디어 엔진의 다른 예의 기능을 개략적으로 나타낸 도면.

[0019] 도 4는 본 발명의 일 실시예에 따른, 예시적인 애플리케이션을 의사 코드 리스트(pseudo code listing)로 나타낸 도면.

[0020] 도 5a는 본 발명의 일 실시예에 따른, 애플리케이션과 연관된 그래픽 인터페이스를 생성하는 방법의 일례를 나타낸 플로우차트.

[0021] 도 5b는 본 발명의 일 실시예에 따른, 애플리케이션과 연관된 그래픽 인터페이스를 생성하는 방법의 다른 예를 나타낸 플로우차트.

[0022] 도 6은 본 발명의 일 실시예에 따른, 투데이 스타일 홈 스크린 뷰(Today style home screen view)의 일례의 스크린샷을 나타낸 도면.

[0023] 도 7a, 도 7b 및 도 7c는 본 발명의 일 실시예에 따른, 홈 스크린 스킨(home screen shot)에의 일정표 통합의 일례의 스크린샷을 나타낸 도면.

[0024] 도 8a 및 도 8b는 본 발명의 일 실시예에 따른, 홈 스크린 스킨에의 전화 통합의 일례의 스크린샷을 나타낸 도면.

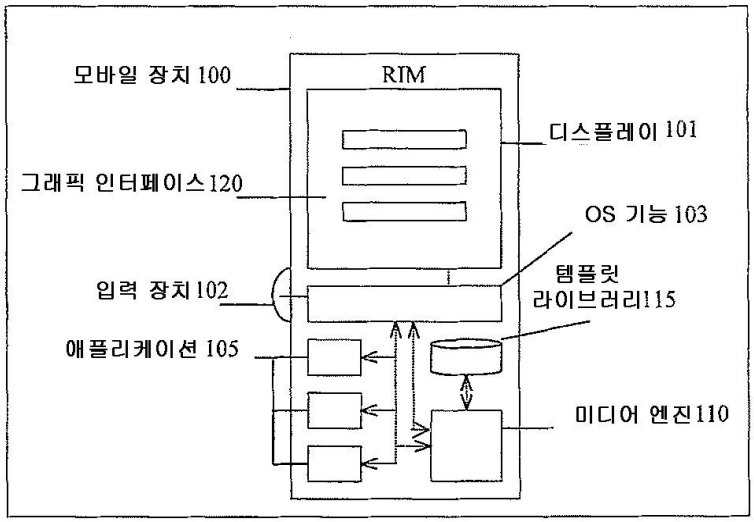
[0025] 도 9는 본 발명의 일 실시예에 따른, 홈 스크린에의 SMS 및 MMS 통합의 일례의 스크린샷을 나타낸 도면.

[0026] 도 10은 본 발명의 일 실시예에 따른, Organize Applications 내부의 홈 스크린(Home Screen) 영역의 개념의 일례를 나타낸 도면.

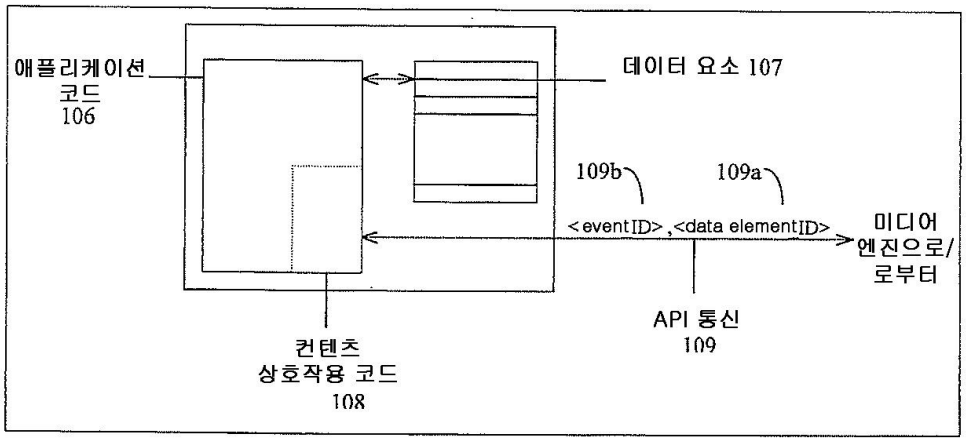
[0027] 도 11은 본 발명의 일 실시예에 따른, 메시지 및 구성 시스템 기능을 갖는 전자 장치 및 홈 스크린 GUI로서 사용될 수 있는 일 실시예에 따라 구성된 양호한 핸드헬드 장치의 일례를 상세히 나타낸 블록도.

도면

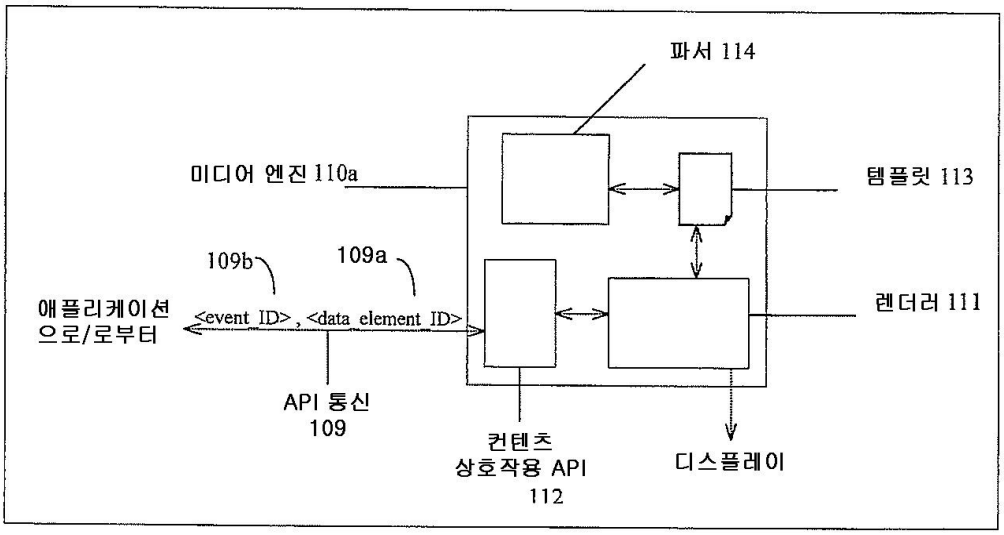
도면1


모바일 장치 100
RIM
디스플레이 101
그래픽 인터페이스 120
OS 기능 103
템플릿
라이브러리115
입력 장치 102
애플리케이션 105
미디어 엔진110

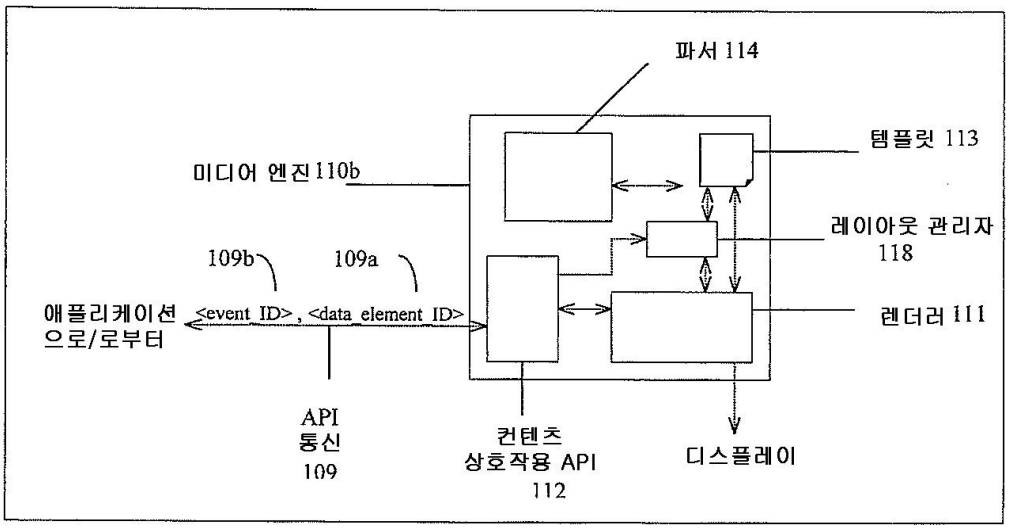
도면2


애플리케이션
코드
106
데이터 요소 107
109b
109a
<eventID>,<data elementID>
미디어
엔진으로/
로부터
API 통신
109
컨텐츠
상호작용 코드
108

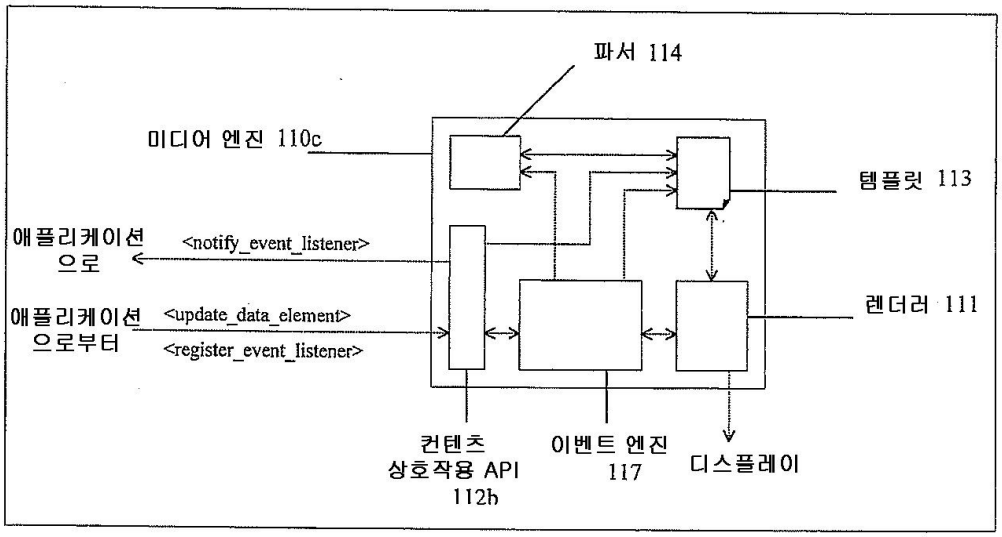
도면3a


파서 114
미디어 엔진 110a
템플릿 113
109b
109a
애플리케이션
으로/로부터
<event_ID>, <data_element_ID>
렌더러 111
API 통신
109
컨텐츠
상호작용 API
112
디스플레이

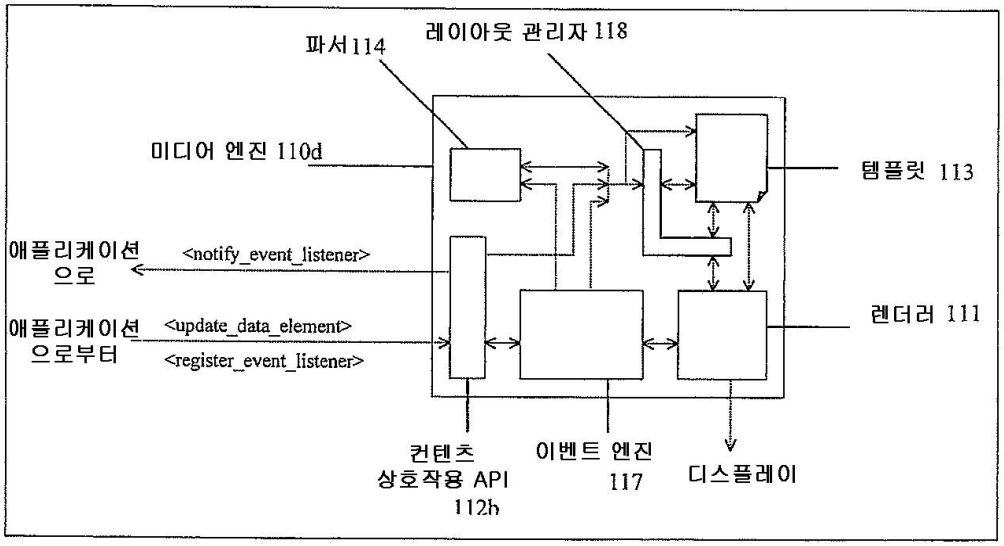
도면3b


파서 114
템플릿 113
미디어 엔진 110b
레이아웃 관리자
118
109b
109a
애플리케이션
으로/로부터
<event_ID>, <data_element_ID>
렌더러 111
API
통신
109
컨텐츠
상호작용 API
112
디스플레이

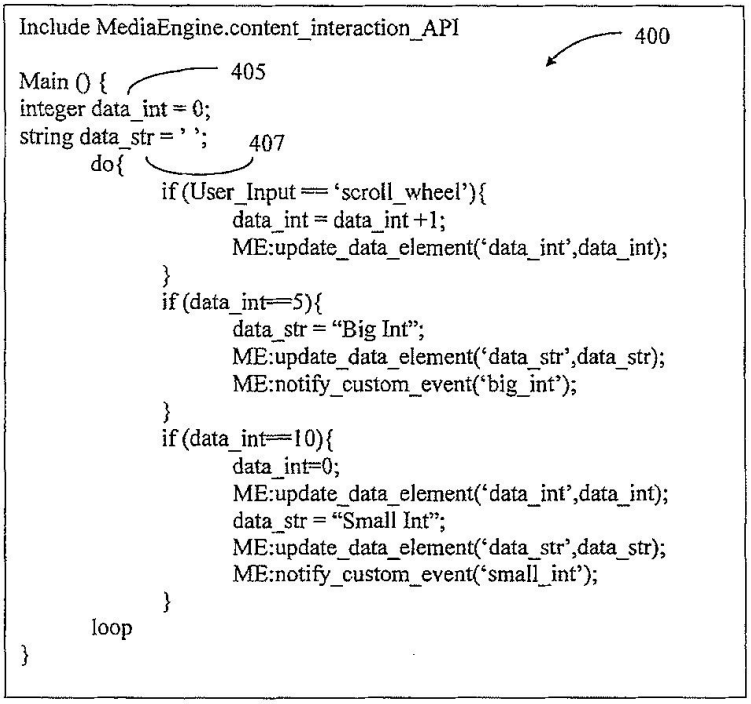
도면3c


파서 114
미디어 엔진 110c
템플릿 113
애플리케이션
으로
<notify_event_listener>
애플리케이션
으로부터
<update_data_element>
<register_event_listener>
렌더러 111
컨텐츠
상호작용 API
112b
이벤트 엔진
117
디스플레이

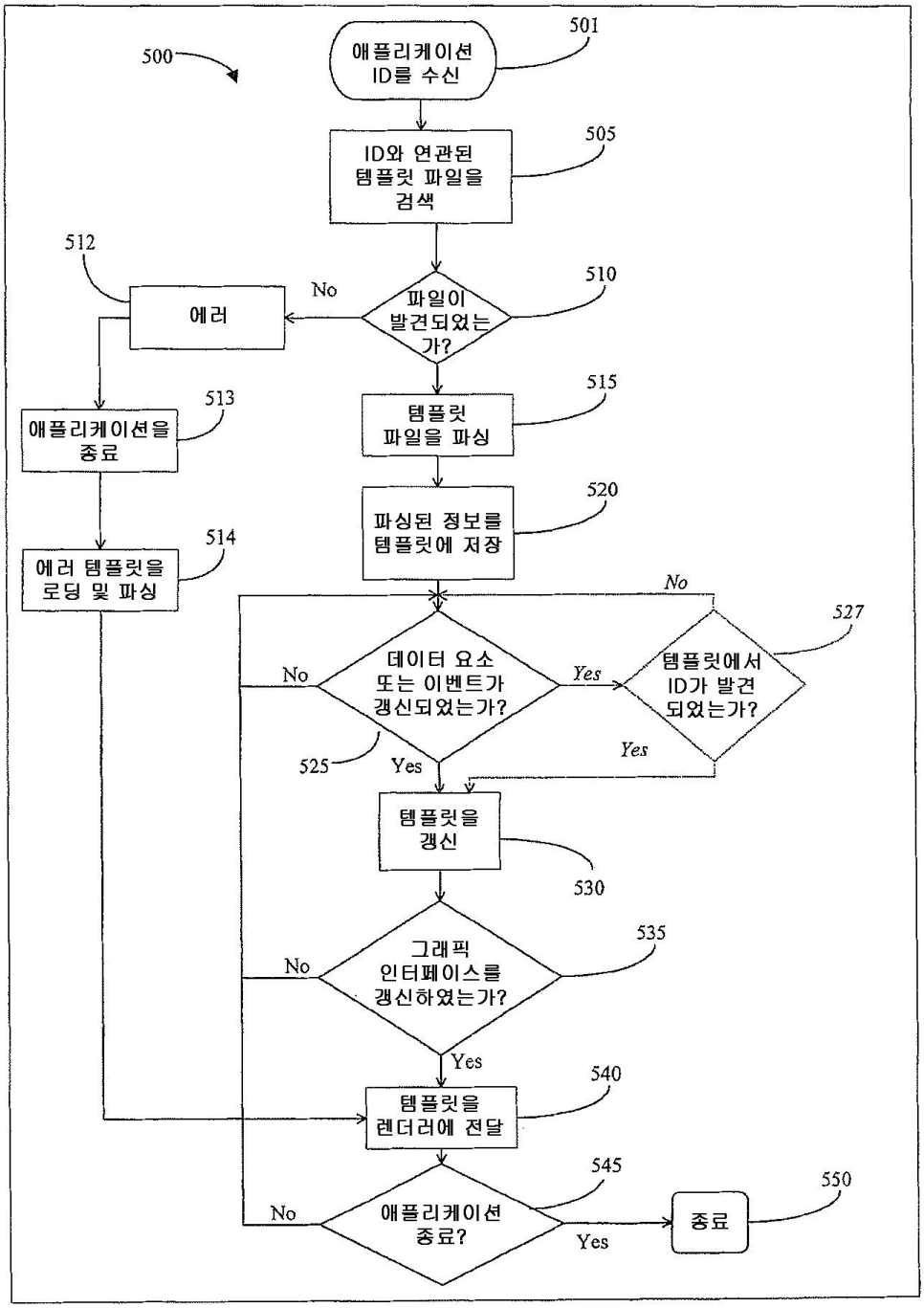
도면3d


파서114
레이아웃 관리자 118
미디어 엔진 110d
템플릿 113
애플리케이션
으로
<notify_event_listener>
애플리케이션
으로부터
<update_data_element>
<register_event_listener>
렌더러 111
컨텐츠
상호작용 API
112b
이벤트 엔진
117
디스플레이

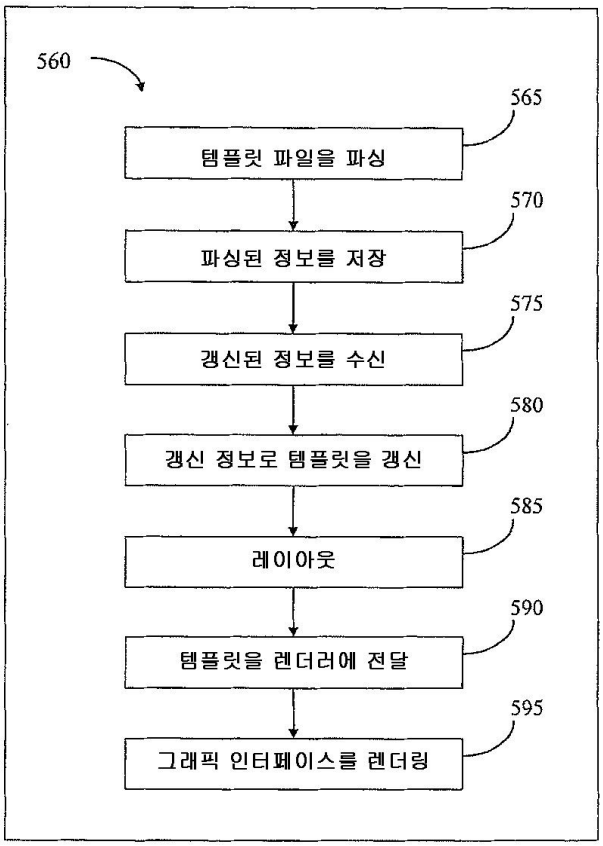
도면4


Include MediaEngine.content_interaction_API
400
Main () {
405
integer data_int = 0;
string data_str = ' ';
407
do{
if (User_Input == 'scroll_wheel'){
data_int = data_int +1;
ME:update_data_element('data_int',data_int);
}
if (data_int==5){
data_str = "Big Int";
ME:update_data_element('data_str',data_str);
ME:notify_custom_event('big_int');
}
if (data_int==10){
data_int=0;
ME:update_data_element('data_int',data_int);
data_str = "Small Int";
ME:update_data_element('data_str',data_str);
ME:notify_custom_event('small_int');
}
loop
}

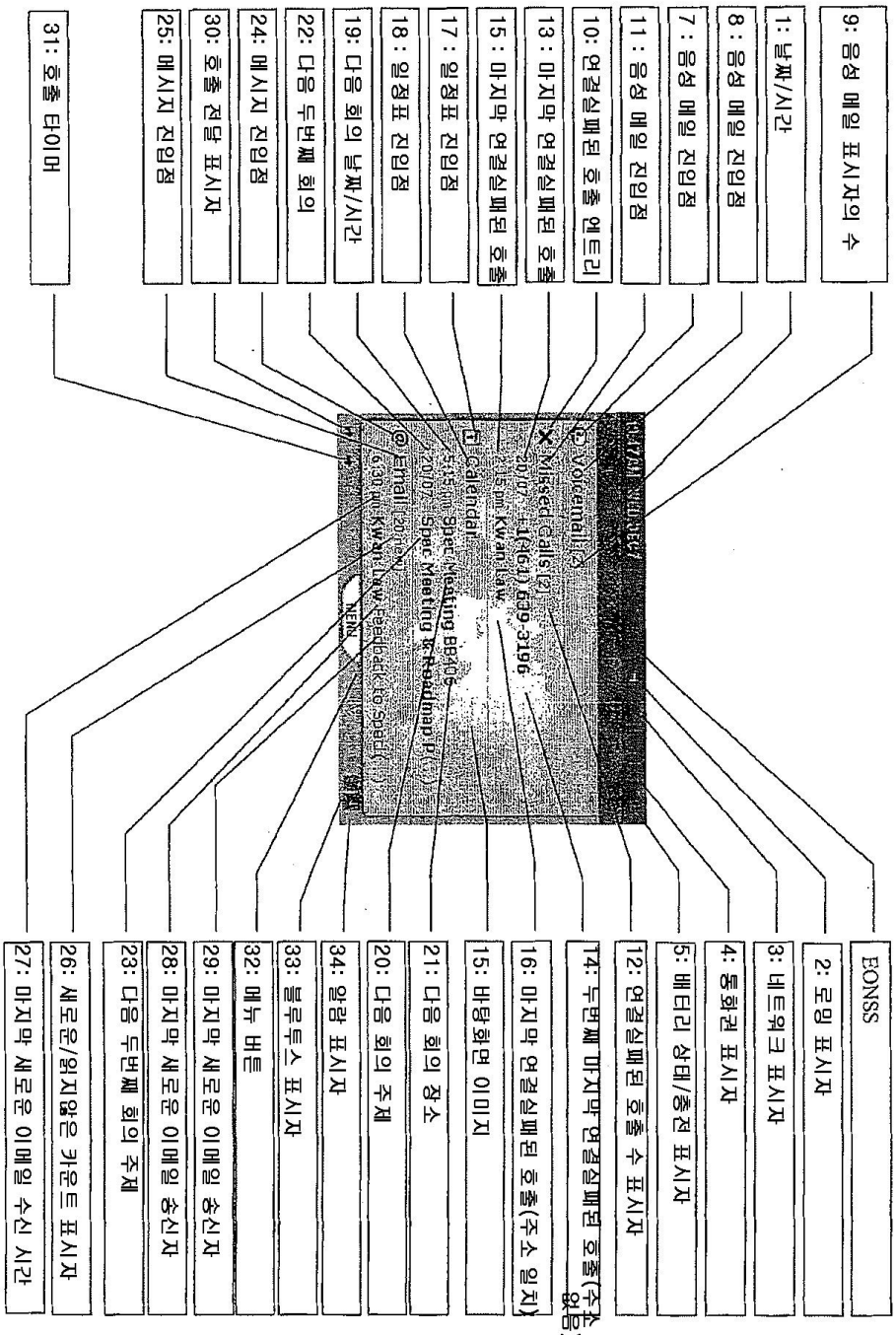
도면5a


500
애플리케이션 ID를 수신
501
ID와 연관된 템플릿 파일을 검색
505
파일이 발견되었는가?
510
No
에러
512
애플리케이션을 종료
513
에러 템플릿을 로딩 및 파싱
514
템플릿 파일을 파싱
515
파싱된 정보를 템플릿에 저장
520
데이터 요소 또는 이벤트가 갱신되었는가?
No
Yes
525
템플릿에서 ID가 발견되었는가?
527
템플릿을 갱신
530
그래픽 인터페이스를 갱신하였는가?
535
템플릿을 렌더러에 전달
540
애플리케이션 종료?
545
종료
550

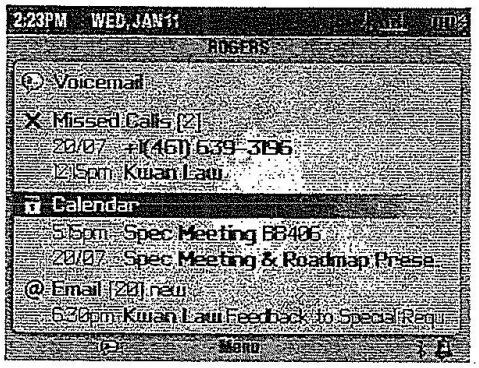
도면5b


560
565
템플릿 파일을 파싱
570
파싱된 정보를 저장
575
갱신된 정보를 수신
580
갱신 정보로 템플릿을 갱신
585
레이아웃
590
템플릿을 렌더러에 전달
595
그래픽 인터페이스를 렌더링

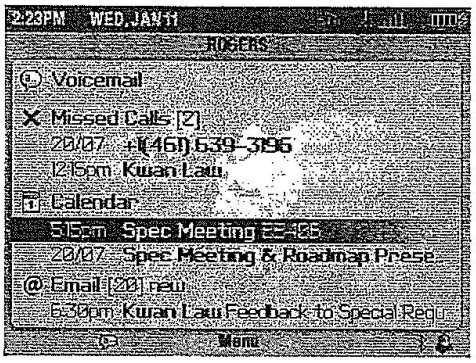
도면6


9: 음성 메일 표시자의 수
1: 날짜/시간
8 : 음성 메일 진입점
7 : 음성 메일 진입점
11 : 음성 메일 진입점
10: 연결실패된 호출 엔트리
13 : 마지막 연결실패된 호출
15 : 마지막 연결실패된 호출
17 : 일정표 진입점
18 : 일정표 진입점
19: 다음 회의 날짜/시간
22: 다음 두번째 회의
24: 메시지 진입점
30: 호출 전달 표시자
25: 메시지 진입점
31: 호출 타이머
EONSS
2: 로밍 표시자
3: 네트워크 표시자
4: 통화권 표시자
5: 배터리 상태/충전 표시자
12: 연결실패된 호출 수 표시자
14: 두번째 마지막 연결실패된 호출(주소 없음)
16: 마지막 연결실패된 호출(주소 일치)
15: 바탕화면 이미지
21: 다음 회의 장소
20: 다음 회의 주제
34: 알람 표시자
33: 블루투스 표시자
32: 메뉴 버튼
29: 마지막 새로운 이메일 송신자
28: 마지막 새로운 이메일 송신자
23: 다음 두번째 회의 주제
26: 새로운/읽지않은 카운트 표시자
27: 마지막 새로운 이메일 수신 시간

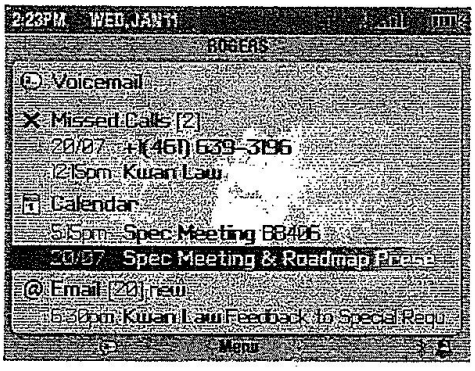
도면7a



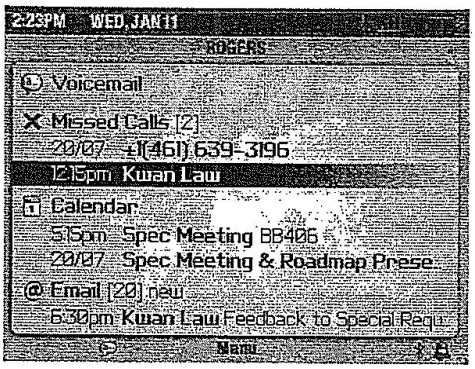
도면7b



도면7c

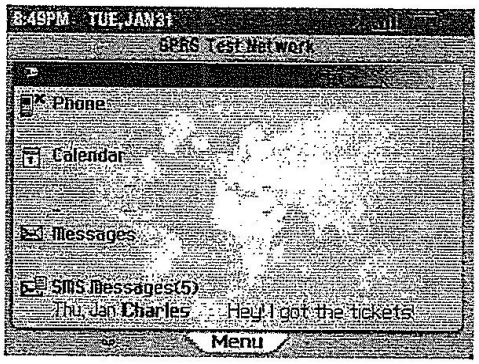


도면8a

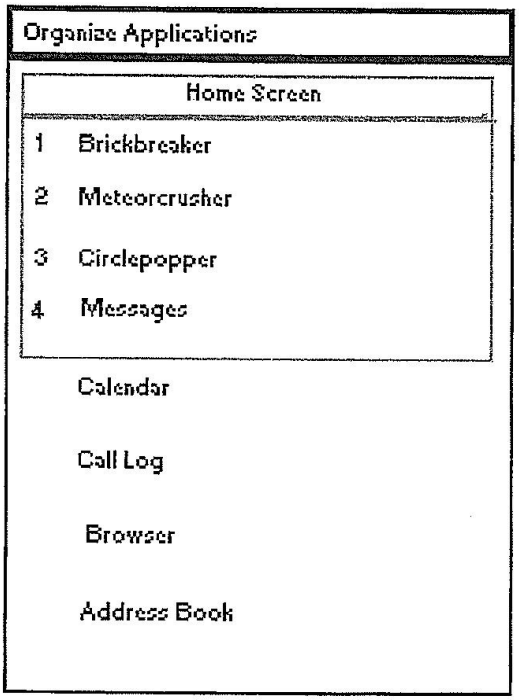


도면8b

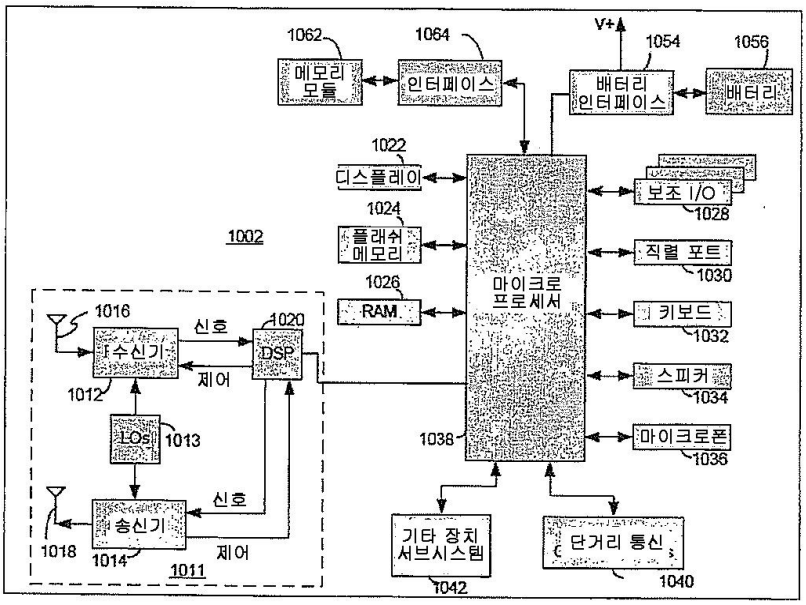
도면9



도면10


Organize Applications
Home Screen
1 Brickbreaker
2 Meteorcrusher
3 Circlepopper
4 Messages
Calendar
Call Log
Browser
Address Book

도면11


1062
메모리 모듈
1064
인터페이스
V+
1054
배터리 인터페이스
1056
배터리
1022
디스플레이
1024
플래쉬 메모리
1026
RAM
1002
마이크로 프로세서
보조 I/O
1028
직렬 포트
1030
키보드
1032
스피커
1034
마이크로폰
1036
1038
기타 장치 서브시스템
1042
단거리 통신
1040
1016
수신기
1012
신호
제어
1020
DSP
LOs
1013
송신기
1018
1014
1011