(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2011-0054555
(43) 공개일자 2011년05월25일

(51) Int. Cl.
G06F 19/00 (2011.01) *G06F 15/18* (2006.01)
(21) 출원번호 10-2009-0111244
(22) 출원일자 2009년11월18일
심사청구일자 2009년11월18일

(71) 출원인
서강대학교산학협력단
서울 마포구 신수동 1-1 서강대학교
(72) 발명자
김동선
서울특별시 마포구 신수동 1 서강대학교
박수용
서울특별시 마포구 신수동 1 서강대학교
(74) 대리인
특허법인충현

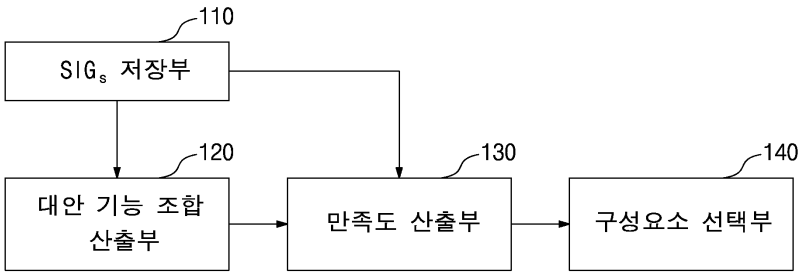
전체 청구항 수 : 총 13 항

(54) 애플리케이션 구성요소 선택 방법 및 장치

(57) 요 약

본 발명은 애플리케이션 구성요소 선택 방법 및 장치에 관한 것으로서 대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출하고, 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 대안 기능 조합들 각각의 만족도를 산출한 다음, 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 애플리케이션 내의 구성요소를 선택하는 것을 특징으로 하며, 상황 변화에 대응하여 최적의 애플리케이션 구성요소를 선택함으로써, 애플리케이션 사용자의 만족도를 최대한 높일 수 있다.

대 표 도 - 도1



이 발명을 지원한 국가연구개발사업
과제고유번호 200950004.01
부처명 정보통신산업진흥원
연구관리전문기관
연구사업명 대학 IT연구센터 육성.지원사업
연구과제명 융합 소프트웨어를 위한 요구 및 품질검증 기술 개발
기여율
주관기관 서강대학교 산학협력단
연구기간 2009년 03월 01일 ~ 2009년 12월 31일

특허청구의 범위

청구항 1

대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출하는 단계;

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계; 및

상기 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 애플리케이션 내의 구성요소를 선택하는 단계를 포함하는 것을 특징으로 하는 애플리케이션 구성요소 선택 방법.

청구항 2

제 1 항에 있어서,

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도는 상기 애플리케이션 주변 환경에 의해 결정되는 것을 특징으로 하는 애플리케이션 구성요소 선택 방법.

청구항 3

제 1 항에 있어서,

상기 대안 기능 조합들 각각의 만족도를 산출하는 단계는,

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 복수의 품질 속성들 각각에 미치는 영향도와 상기 복수의 품질 속성들 각각이 갖는 가중치를 고려하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계인 것을 특징으로 하는 애플리케이션 구성요소 선택 방법.

청구항 4

제 1 항에 있어서,

상기 대안 기능 조합들 각각의 만족도를 산출하는 단계는,

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 복수의 품질 속성들 각각에 미치는 영향도를 합산하여 각 품질 속성별 스코어를 산출하고, 상기 산출된 스코어에 상기 가중치를 적용하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계인 것을 특징으로 하는 애플리케이션 구성요소 선택 방법.

청구항 5

제 1 항에 있어서,

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도는 대안 기능과 품질 속성에 의해 결정되는 상황 평가 함수에 의해 산출되며, 주변 상황을 수치적으로 표현하는 상황 변수를 상기 상황 평가 함수의 입력으로 할 때, 상기 상황 평가 함수의 출력값을 대안 기능과 품질 속성 간의 영향도로 결정하는 것을 특징으로 하는 애플리케이션 구성요소 선택 방법.

청구항 6

제 1 항에 있어서,

유전 알고리즘을 이용하여 상기 대안 기능 조합들 중 가장 만족도가 큰 대안 기능 조합을 선택하는 것을 특징으로 하는 애플리케이션 구성요소 선택 방법.

청구항 7

제 1 항 내지 제 6 항 중에 어느 한 항의 방법을 컴퓨터에서 실행시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체.

청구항 8

대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출하는 대안 기능 조합 산출부;

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 만족도 산출부; 및

상기 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 애플리케이션 내의 구성요소를 선택하는 구성요소 선택부를 포함하는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치.

청구항 9

제 8 항에 있어서,

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도는 상기 애플리케이션 주변 환경에 의해 결정되는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치.

청구항 10

제 8 항에 있어서,

상기 만족도 산출부는 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 복수의 품질 속성들 각각에 미치는 영향도와 상기 복수의 품질 속성들 각각이 갖는 가중치를 고려하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치.

청구항 11

제 8 항에 있어서,

상기 만족도 산출부는 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 복수의 품질 속성들 각각에 미치는 영향도를 합산하여 각 품질 속성별 스코어를 산출하고, 상기 산출된 스코어에 상기 가중치를 적용하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치.

청구항 12

제 8 항에 있어서,

상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도는 대안 기능과 품질 속성에 의해 결정되는 상황 평가 함수에 의해 산출되며, 주변 상황을 수치적으로 표현하는 상황 변수를 상기 상황 평가 함수의 입력으로 할 때, 상기 상황 평가 함수의 출력값을 대안 기능과 품질 속성 간의 영향도로 결정하는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치.

청구항 13

제 8 항에 있어서,

상기 만족도 산출부는 유전 알고리즘을 이용하여 상기 대안 기능 조합들 중 가장 만족도가 큰 대안 기능 조합을 선택하는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치.

명 세 서

발명의 상세한 설명

기 술 분 야

[0001] 본 발명은 애플리케이션 구성요소 선택 방법에 관한 것으로서, 더욱 상세하게는 상황 변화에 대응하여 최적의 애플리케이션 구성요소를 선택함으로써, 애플리케이션 사용자의 만족도가 높은 애플리케이션 구성요소 선택 방법 및 장치에 관한 것이다.

배 경 기 술

[0002] 소프트웨어 산업에서는 모바일 기기를 통해 사용자들에게 이동의 편의성을 제공하는데 중점을 두고 있다. 이러

한 모바일 기기 내에 포함되는 모바일 애플리케이션들은 많은 상황 변화에 점점 더 많이 노출된다. 이러한 상황 변화는 모바일 애플리케이션이 제공하는 서비스의 질에 영향을 주게 된다. 지금까지는 개발자들이 수동으로 아키텍처 구조를 재구성하거나 휴리스틱 기법을 이용하여 상황 변화에 대처하였다. 그러나 이러한 대처 방법은 애플리케이션이 다양한 상황들에 노출되고, 많은 아키텍처 구성 요소가 있는 경우에 효과적이지 못하다. 따라서 상황 변화에 대응하여 애플리케이션이 만족할만한 성능을 유지하기 위해서 애플리케이션의 구성이 동적으로 결정될 필요가 있다.

발명의 내용

해결 하고자하는 과제

[0003] 따라서, 본 발명이 해결하고자 하는 첫 번째 과제는 상황 변화에 대응하여 최적의 애플리케이션 구성요소를 선택함으로써, 애플리케이션 사용자의 만족도가 높은 애플리케이션 구성요소 선택 방법을 제공하는 것이다.

[0004] 본 발명이 해결하고자 하는 두 번째 과제는 상황 변화에 대응하여 최적의 애플리케이션 구성요소를 선택함으로써, 애플리케이션 사용자의 만족도가 높은 애플리케이션 구성요소 선택 장치를 제공하는 것이다.

과제 해결수단

[0005] 본 발명은 상기 첫 번째 과제를 달성하기 위하여, 대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출하는 단계; 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계; 및 상기 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 애플리케이션 내의 구성요소를 선택하는 단계를 포함하는 것을 특징으로 하는 애플리케이션 구성요소 선택 방법을 제공한다.

[0006] 본 발명의 일 실시예에 의하면, 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도는 상기 애플리케이션 주변 환경에 의해 결정될 수 있다.

[0007] 또한, 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계는, 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 복수의 품질 속성들 각각에 미치는 영향도와 상기 복수의 품질 속성들 각각이 갖는 가중치를 고려하여 상기 대안 기능 조합들 각각의 만족도를 산출할 수 있다.

[0008] 본 발명의 다른 실시예에 의하면, 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계는, 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 복수의 품질 속성들 각각에 미치는 영향도를 합산하여 각 품질 속성별 스코어를 산출하고, 상기 산출된 스코어에 상기 가중치를 적용하여 상기 대안 기능 조합들 각각의 만족도를 산출할 수 있다.

[0009] 또한, 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도는 대안 기능과 품질 속성에 의해 결정되는 상황 평가 함수에 의해 산출되며, 주변 상황을 수치적으로 표현하는 상황 변수를 상기 상황 평가 함수의 입력으로 할 때, 상기 상황 평가 함수의 출력값을 대안 기능과 품질 속성 간의 영향도로 결정할 수 있다.

[0010] 본 발명의 또 다른 실시예에 의하면, 유전 알고리즘을 이용하여 상기 대안 기능 조합들 중 가장 만족도가 큰 대안 기능 조합을 선택할 수 있다.

[0011] 본 발명은 상기 두 번째 과제를 달성하기 위하여, 대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출하는 대안 기능 조합 산출부; 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 만족도 산출부; 및 상기 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 애플리케이션 내의 구성요소를 선택하는 구성요소 선택부를 포함하는 것을 특징으로 하는 애플리케이션 구성요소 선택 장치를 제공한다.

효 과

[0012] 본 발명에 따르면, 상황 변화에 대응하여 최적의 애플리케이션 구성요소를 선택함으로써, 애플리케이션 사용자의 만족도가 높은 효과가 있다. 또한, 본 발명에 따르면, 실시간으로 환경과 사용자 품질 요구의 변화에 응답하

여 많은 양의 후보군으로부터 적절한 애플리케이션 구성요소를 동적으로 신속하게 선택할 수 있다.

발명의 실시를 위한 구체적인 내용

[0013] 이하, 본 발명을 상세하게 설명한다.

[0014] 본 발명의 일 실시예에 따른 애플리케이션 구성요소 선택 방법은 대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출하는 단계; 상기 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 상기 대안 기능 조합들 각각의 만족도를 산출하는 단계; 및 상기 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 애플리케이션 내의 구성요소를 선택하는 단계를 포함한다.

[0015] 이하, 바람직한 실시예를 들어 본 발명을 더욱 상세하게 설명한다. 그러나 이들 실시예는 본 발명을 보다 구체적으로 설명하기 위한 것으로, 본 발명의 범위가 이에 의하여 제한되지 않는다는 것은 당업계의 통상의 지식을 가진 자에게 자명할 것이다.

[0016] 애플리케이션에 대한 사용자 품질 요구가 변화하거나 애플리케이션이 동작하는 주변 상황이 변화함에 따라 모바일 애플리케이션에 대한 성능 평가는 달라진다.

[0017] 사용자 요구는 기능적 요구와 비기능적 요구 두 가지 형태가 존재한다. 기능적 요구(functional requirements, FRs)는 소프트웨어 시스템에서 관찰되는 시스템 행동을 의미한다. 반면 비기능적 요구(nonfunctional requirements, NFRs)는 소프트웨어 시스템의 품질과 관련이 있다. 사용자의 비기능적 요구 즉, 품질 요구(quality requirement)는 품질 속성(quality attribute)의 우선 순위를 나타내는 가중치를 이용하여 표현할 수 있다. 가중치가 높은 품질 속성일수록 사용자가 만족도를 평가할 때 중요하게 생각하는 품질 속성이다.

[0018] 따라서 사용자 품질 요구의 변화는 품질 속성(quality attribute)에 부여되는 가중치를 변경함으로써 수치화할 수 있다. 품질 속성은 품질 변수(quality variable)로 표현되며, 품질 속성이란 사용자가 관심을 갖는 속성으로 애플리케이션에 대한 사용자 만족도 평가 항목을 의미한다. 모바일 애플리케이션의 품질 속성의 예로는 응답성, 사용성, 내구성, 및 가독성 등을 들 수 있다.

[0019] 주변 상황은 상황 변수(situation variable)로 표현할 수 있다. 상황 변수는 주변 환경에서의 상황을 수치적으로 표시하기 위해 사용되는데, RSSI(received signal strength indication) 레벨, 배터리 레벨, 밝기 등을 예로 들 수 있다. 또한, 상황 변수는 명목값과 수치값 중 어느 하나를 가질 수 있으며, RSSI 레벨과 배터리 레벨은 [낮음,중간,높음]과 같은 명목값을 갖고, 밝기 레벨은 0과 255 사이의 수치값을 가질 수 있다.

[0020] 상황 변수의 변화는 환경의 변화를 의미한다. 예를 들어 상황 변수 "RSSI level"의 현재 값이 "낮음"에서 "높음"으로 변화할 경우, 애플리케이션의 주변 환경이 잡음이 많은 환경으로 변화하였음을 알 수 있다. 환경 변화를 전체적으로 나타내기 위해 상황 변수를 벡터로 표시할 수 있다. 예를 들면, 애플리케이션 A1이 RSSI 레벨, 배터리 레벨, 및 밝기 레벨과 관련이 있다고 하면, 벡터<(rssi),(battery),(brightness)>는 전체적인 환경 상태를 나타낸다. 즉, <0,1,220>은 RSSI 레벨=0, 배터리 레벨=1, 밝기 레벨=220인 것을 나타낸다.

[0021] 이상에서 살펴본 상황의 변화와 사용자의 품질 요구 변화에 대응하여 가장 품질 성능이 우수하게 소프트웨어 구조를 형성하도록 하기 위해서 애플리케이션은 다양한 대안 기능들(alternative functions)을 갖고 있다.

[0022] 도 1은 본 발명의 일 실시예에 따른 애플리케이션 구성요소 선택 장치의 블록도이다.

[0023] 도 1을 참조하면, 본 발명의 일 실시예에 따른 애플리케이션 구성요소 선택 장치는 SIGs 저장부(110), 대안 기능 조합 산출부(120), 만족도 산출부(130), 및 구성요소 선택부(140)로 구성된다.

[0024] SIGs 저장부(110)는 SIGs(softgoal interdependency graphs)를 저장한다. SIGs는 품질 속성들 간의 관계 또는 품질 속성과 대안 기능과의 관계를 나타내는 그래프이다. SIGs에는 대안 타입별로 속해 있는 대안 기능들에 대한 정보와 대안 기능들이 품질 속성에 미치는 영향도에 대한 정보를 갖고 있다.

[0025] 대안 기능 조합 산출부(120)는 대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출한다.

[0026] 만족도 산출부(130)는 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 대안 기능 조합들 각각의 만족도를 산출한다.

[0027] 구성요소 선택부(140)는 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 어플리케이션 내

의 구성요소를 선택한다.

[0028] 도 2는 SIGs 저장부(110)가 저장하고 있는 SIGs의 일 실시예를 나타낸 것이다.

[0029] SIGs에서 소프트골(softgoal)은 품질 속성에 대응하며, 구름 형태로 표시된다. 두 개의 소프트골 간의 상호 의존성(interdenpendency)은 두 개의 소프트골을 연결하는 선으로 표시된다. SIGs는 가는 선의 구름모양들로 표현되고, 동작 목표(operationalizing goal)는 굵은 선의 구름모양들로 표현되며, 대안 타입들은 직사각형으로 구분된다. 동작 목표는 대안 기능에 상응하는 개념이다.

[0030] SIGs는 가독성(readability), 성능(performance), 내구성(durability), 사용성(usability)과 같은 추상적인 품질 속성으로부터 시작한다. 이러한 추상적인 품질 속성은 소프트골로 표현되고, 이러한 초기 소프트골들로부터 보다 세부적인 서브 소프트골들을 생성한다. 예를 들어, 모바일 애플리케이션의 "성능" 소프트골을 분해하면, 응답성(responsiveness), 메모리 사용량(memory usage), 지연(latency) 소프트골과 같은 서브 소프트골들로 분해된다. 하나 이상의 자식 소프트골이 하나의 부모 소프트골로부터 분해될 때, 소프트골 간의 특정 관계를 정할 수 있다.

[0031] 부모 소프트골을 만족시키기 위해서 모든 자식 소프트골들이 만족되어야 한다면, 이러한 관계를 "AND" 관계로 정의하고, 상호 의존성을 표시하는 선 사이에 아크로 표시한다. 또한, 하나 이상의 자식 소프트골이 만족될 때, 부모 소프트골도 만족되는 경우 "OR" 관계로 정의하고, 상호 의존성을 표시하는 선 사이에 두 개의 아크로 표시한다. 소프트골이 만족되는 경우란 사용자의 만족도가 올라간다고 판단되는 경우를 말한다.

[0032] 한편, 지연(latency) 소프트골과 응답성(responsiveness) 소프트골은 네트워크 지연(network latency)과 리프레시율(refresh rate) 소프트골과 같은 더 세부적인 서브 소프트골들로 세분화된다. 이러한 소프트골 세분화 과정은 충분한 소프트골이 생성될 때까지 계속된다. 도 2에 도시된 SIGs를 이용하여 품질 속성(소프트골)을 표현하는 것은 개발자가 서브 품질 속성(서브 소프트골)을 쉽게 도출하도록 하는 장점이 있다.

[0033] 도 2를 참조하면, 굵은 선으로 표시된 대안 기능들은 동작 목표들로 표현된다. 동작 목표는 재구성 가능한 애플리케이션 구조를 설계시 어떠한 구성요소를 결정할 것인지를 나타내기 위해 사용한다. 예를 들면, 대안 타입 t_3에 포함된 Simple Messaging, Compressed Messaging, Videotelephony 대안 기능들 중 어느 것을 사용할 것인지에 따라 애플리케이션의 구성요소가 결정된다.

[0034] 한편, 동작 목표와 소프트골 간의 이러한 관계는 기여도(contribution)에 의해 표현된다. 동작 목표는 하나 이상의 소프트골에 기여하며, 기여하는 정도에 따라 정량화하여 "+++" 또는 "-"로 표현할 수 있다. 동작 목표가 소프트골에 기여하는 정도를 나타내기 위해서 상황 평가 함수(situation evaluation function)를 사용하기도 한다.

[0035] 비슷한 특징을 공유하는 대안 기능들은 하나의 대안 타입으로 그룹화된다. 대안 타입 t_i를 구성하는 대안 기능들 중에 하나의 대안 기능이 선택되거나 어떠한 대안 기능도 선택되지 않을 수 있다. 동시에 선택될 수 없는 대안 기능들이 별개의 대안 타입으로 그룹핑되고, 대안 기능들은 어느 하나의 대안 타입에는 속해야 한다. 이러한 대안 기능들과 대안 타입 간의 관계를 다음의 수학식 1과 같이 나타낼 수 있다.

수학식 1

[0036]
$$\forall t_i \in T \quad \text{and} \quad i_j \in I,$$
$$t_i \in 2^I,$$
$$\bigcap_i^n t_i = \emptyset, \quad \bigcup_i^n t_i = I,$$
$$M_{t_i}(k) = i_j, \quad i_j \in t_i$$

[0037] 여기서, T는 대안 타입들의 집합이고, I는 대안 기능들의 집합이다. n은 대안 타입들의 개수이고, $M_{t_i}(k)$는 시간 k에서 대안 타입 t_i에 속한 대안 기능을 선택하는 함수이다. $M_{t_i}(k)$는 시간 k에 t_i로부터 단 하나의 대안 기능을 선택하는 함수이다.

[0038] 도 3 내지 도 5는 만족도 산출부(130)가 선택된 대안 기능들의 사용자 만족도를 산출하는 방법을 나타낸

것이다.

[0039] 도 3은 주변의 상황을 고려하여 선택된 대안 기능들의 사용자 만족도를 평가하는 일 실시예를 도시한 것이다. 상황 변수를 살펴보면, RSSI 레벨=5, 배터리 레벨=3, 밝기 레벨=120이다.

[0040] "RichGUI", "NormalGUI", 및 "SimpleGUI"는 GUI 대안 타입에 속하는 대안 기능들의 일 예이다. 이러한 대안 기능들은 상황과 품질 요구가 변할 때 애플리케이션이 선택할 수 있으며, 대체 가능한 기능들을 나타낸다.

[0041] 이러한 대안 기능들은 타입별로 그룹화될 수 있다. 예를 들면, "HighContrastDisplay"와 "NormalDisplay" 대안 기능들은 같은 대안 타입인 "Display 대안 타입"에 속한다. 각 대안 타입 내에서는 단 하나의 대안 기능만이 선택되어 소프트웨어 구조를 형성할 수 있다. 즉 "NormalDisplay"와 "HighContrastDisplay"가 동시에 선택될 수는 없다.

[0042] 각 대안 기능은 하나 이상의 품질 속성과 서로 연결된다. 예를 들면, GUI(graphical user interface) 대안 타입 내에 속한 여러 대안 기능들은 애플리케이션의 사용성(usability)과 내구성(durability)에 영향을 미친다. 풍부한 GUI는 더 나은 사용자 편리성을 제공하지만, 배터리 소모도 많기 때문에 GUI 대안 타입은 사용성과 내구성과 관련이 있다.

[0043] 이와 같은 대안 기능과 품질 속성 간의 관계를 정량적으로 분석하는 방법은 다음과 같다. 예를 들면, "High Contrast Display" 대안 기능은 가독성에는 긍정적인 영향을 미치지만, 내구성에는 부정적인 영향을 미칠 것이다. 이때 사용자의 만족도에 긍정정인 영향을 미치는 경우에는 "+"로 표시하고, 부정적인 영향을 미치는 경우에는 "-"로 표시하며, 영향의 정도는 "+" 또는 "-"의 표시 개수로 표시하기로 한다.

[0044] 각 품질 속성에 대안 기능들이 미치는 영향을 더하면 도 3의 각 품질 속성 위에 표시된 스코어가 연산된다. "+"와 "-"는 각각 "+1"과 "-1"을 의미한다. 각 품질 속성별로 연산된 스코어는 사용자가 각 품질 속성에 대하여 얼마나 만족하고 있는지를 나타내는 척도가 된다.

[0045] 도 3에 도시된 품질 속성에 대응하는 스코어들에 가중치를 적용한 후 합함으로써, 사용자의 품질 요구에 대한 전체적인 만족도를 측정할 수 있다. 사용자가 정한 각 품질 속성에 대한 가중치가 있는 경우에는 이 가중치와 스코어들을 이용하여 만족도를 측정한다. 가중치란 사용자에 의해 결정되는 각 품질 속성에 대한 우선도를 의미한다. 도 3에서 "반응성(Responsiveness)"은 0.2, "사용성(Usability)"는 0.5, "내구성(Durability)"는 0.1, "가독성(Readability)"는 0.2의 가중치를 갖고 있으며, 가중치로부터 판단하면 사용자는 "사용성"를 가장 우선적으로 고려하는 것을 알 수 있다.

[0046] 각 품질 속성의 스코어에 사용자가 결정한 가중치를 적용하여 전체 품질 속성의 스코어를 산출하면, 0.8의 만족도가 된다. 이 만족도는 다수의 대안 기능들 중에서 선택된 대안 기능들(RichGUI, High Contrast Display, Videotelephony)에 의해 결정된 만족도이다.

[0047] 이와 같이 결정된 만족도는 현재 상황과 사용자의 품질 요구에 따라 선택된 대안 기능들을 평가하는 기준이 된다. 모든 가능한 대안 기능들의 조합들의 만족도를 연산하고 비교함으로써, 가장 만족도가 높은 대안 기능들의 조합을 찾을 수 있다.

[0048] 도 3에 도시된 대안 기능들의 조합은 각 대안 타입별로 하나의 대안 기능들을 선택할 수 있다고 할 때 총 18개(3×2×3)가 가능하고, 총 18개의 조합의 만족도를 계산하여 비교함으로써, 가장 높은 만족도를 갖는 대안 기능의 조합을 찾을 수 있다. 또한, 가장 높은 만족도를 갖는 대안 기능의 조합은 사용자에게 가장 좋은 만족을 제공한다는 것을 유추할 수 있다.

[0049] 도 4는 도 3의 상황 변수가 변화된 경우에 대안 기능들의 만족도를 평가하는 다른 실시예를 도시한 것이다.

[0050] 상황 변수를 살펴보면, RSSI 레벨=1, 배터리 레벨=1, 밝기 레벨=50으로 도 3에 도시된 상황 변수의 값과는 다르다는 것을 알 수 있다. 상황 변수가 다르므로, 각 대안 기능들이 품질 속성에 미치는 영향도 달라졌음을 확인할 수 있다. 그 결과 도 3에서 선택한 대안 기능들과 동일한 대안 기능들을 선택하여도 만족도는 0.8에서 -1.4로 떨어졌음을 알 수 있다. 보다 상세히 살펴보면, 도 4의 RSSI 레벨은 도 3의 RSSI 레벨보다 낮은데, 낮은 RSSI 레벨에서 "Videotelephony" 대안 기능은 다른 대안 기능들보다도 더 많은 네트워크 리소스를 사용하기 때문에, "반응성" 품질 속성에 더 부정적인 영향을 미치게 되어 만족도가 떨어지게 된다.

[0051] 따라서 주변 환경이 변화하는데 적응하고, 사용자의 품질 요구를 최대한 만족시키기 위해서는 선택되는 각 대안 기능들의 조합을 변경하여야 한다는 것을 알 수 있다. 즉, 상황이 변화할 때마다 애플리케이션은 실시간으로 대

안 기능들의 모든 조합에 대하여 만족도를 재계산하고, 가장 만족도가 높은 조합을 선택한다. 다만, 많은 대안 기능들이 존재하는 경우에는 대안 기능들의 조합에 대응하는 만족도를 계산하는데 시간이 많이 소요되어, 이로 인해 사용자가 불만족을 느끼게 될 가능성이 커질 것이다.

[0052] 도 5는 사용자 품질 요구가 변화된 경우에 대안 기능들의 만족도를 평가하는 또 다른 실시예를 도시한 것이다.

[0053] 상황 변수를 살펴보면, 도 3의 상황변수의 값과 동일한 RSSI 레벨=5, 배터리 레벨=3, 밝기 레벨=120이다.

[0054] 사용자 품질 요구가 변화된 경우는 각각의 품질 속성에 대응하는 가중치가 변경된다는 것을 의미한다. 도 5에 도시된 상황 변수들의 값들은 도 3에 도시된 상황 변수들의 값들과 같지만, 각 품질 속성에 대응하는 가중치는 다르다. 그 결과 동일하게 선택된 대안 기능들의 사용자 만족도는 0.8에서 -0.1로 낮아졌다. 사용자 품질 요구가 변화된 경우도 상황 변수가 변화한 경우와 마찬가지로 모든 대안 기능들의 조합에 대하여 만족도를 재계산하여 가장 높은 만족도를 갖는 조합을 찾아야 한다.

[0055] 한편, 가중치와 상황 변수가 가질 수 있는 값들이 유한하지 않아 미리 모든 만족도를 계산하는 것이 불가능하므로, 실시간으로 가장 최적의 대안 기능들의 조합을 찾기 위해서는 동적으로 대안 기능들의 조합에 대응하는 만족도를 재계산하여야 한다.

[0056] 이상에서와 같이 가장 최적의 대안 기능들의 조합을 찾은 후, 애플리케이션은 최적의 조합에 따라 애플리케이션의 구성요소를 재구성한다. 즉 상황이나 사용자 품질 요구가 변화하면, 애플리케이션은 현재의 상황 변수 값과 품질 요구에 대응한 가중치에 적절한 대안 기능들의 조합을 찾고, 그 조합에 따라 소프트웨어 또는 하드웨어 구성요소를 재구성한다. 애플리케이션 구성요소란 애플리케이션 내에 있고 기능적으로 구분되는 단위 하드웨어 또는 소프트웨어를 포함하는 의미이다.

[0057] 이상에서 개시된 내용으로부터 모바일 애플리케이션에서 동적으로 애플리케이션 내의 구성요소를 선택하는 방법을 보다 상세하게 설명하기로 한다.

[0058] 대안 기능들은 아키텍처 결정 변수(architectural decision variable)의 값을 정의하는데 사용되며, 도 6을 참조하여 아키텍처 결정 변수에 대하여 살펴보기로 한다.

[0059] 도 6은 아키텍처 결정 변수들, 대안 타입, 상황 변수들, 및 품질 속성들 간의 상호 관계를 나타내는 도면이다. 도 6을 참조하면, 구성요소 선택부(140)가 아키텍처 결정 변수에 대응하는 애플리케이션 내의 구성요소를 선택하는 방법을 알 수 있다.

[0060] 아키텍처 결정 변수는 대안 타입을 사용하여 아키텍처 구성(configuration)의 일부를 결정한다. 즉, 하나의 대안 타입은 하나의 아키텍처 결정 변수에 대응한다. 대안 타입 내의 대안 기능들은 하나의 아키텍처 결정 값(NORM, HCONST)에 대응하기 때문에 대안 타입과는 달리 아키텍처 결정 변수는 애플리케이션의 아키텍처 구조의 일부를 나타낼 수 있다. 아키텍처 결정 값은 구성요소들의 부분집합을 나타내고, 구성요소들 간의 관계를 나타낼 수 있다. a_i가 아키텍처 결정 변수라고 하면, a_i는 아키텍처 결정 값 v_i를 가질 수 있다. 이때 $v_i \in V$이고, V는 아키텍처 결정 값들의 집합이다. v_i는 대안 기능 i_j($i_j \in I$)에 일대일 대응한다. 따라서 아키텍처 결정 값들의 개수($|V|$), 대안 기능들의 개수($|I|$)는 같아야 한다. 대안 타입 t_i($t_i \in T$)는 아키텍처 결정 변수 a_i에 대응하므로, 그 개수는 서로 같아야 한다.

[0061] 아키텍처 결정 변수들의 조합은 아키텍처 인스턴스들로 표현된다. e_i를 아키텍처 인스턴스라고 하면, e_i는 아키텍처 결정 변수들의 벡터로 표현될 수 있다. 예를 들면, e_i=<a_1=HCONST, a_2=RichGUI, a_3=0, ..., a_n=SIMPLEMSG>로 표현할 수 있다. E를 가능한 아키텍처 인스턴스들의 집합이라고 하면, 다음의 수학식 2와 같이 표현될 수 있다.

수학식 2

$$|E| = |a_1| \times |a_2| \times \cdots \times |a_n| = \prod_{i}^{n} |a_i|$$

[0062]

[0063] 여기서, $|a_i|$는 a_i이 가질 수 있는 아키텍처 결정 값들의 수이고, $|E|$는 애플리케이션이 가질 수 있는 아키텍처 인스턴스들의 수이다. n은 아키텍처 결정 변수들의 수($|A|=n$)이다.

[0064] 수학식 2에서 볼 수 있는 바와 같이 가능한 아키텍처 인스턴스들의 수 $|E|$가 아키텍처 선택 문제의 복잡도를 결정하게 된다.

[0065] 한편, 품질 변수들은 선택된 아키텍처 인스턴스에 대한 사용자 만족도를 평가하는데 사용된다. 아키텍처 결정 변수들은 아키텍처 인스턴스에 포함된 대안 기능들을 나타내는데 사용된다. 상황 변수를 변수로 갖는 상황 평가 함수는 아키텍처 인스턴스가 품질 속성에 미치는 영향을 결정하는데 사용된다.

[0066] 도 7 내지 도 10은 대안 기능들이 품질 속성에 미치는 영향도를 고려하여 본 발명의 일 실시예에 따른 만족도 산출부(130)가 만족도를 산출하는 방법을 나타낸 것이다.

[0067] 도 7은 본 발명의 일 실시예에 따라 동적으로 품질 속성에 대응하는 스코어들을 연산하는 방법을 개념적으로 나타낸 것이다.

[0068] 스코어들은 품질 속성에 대응하는 가중치와 함께 만족도를 산출하는데 사용되는 값이다. 특히, 이하에서는 SIGs(softgoal interdependency graphs)를 이용하여 만족도를 평가하는 방법을 살펴보기로 한다.

[0069] 추상적인 초기 소프트골들은 품질 속성에 대응하는 표현이며, 품질 속성들은 품질 변수(quality variable) q_i에 의해 그 만족의 정도가 표시된다. SIGs의 소프트골들 중에서 가장 높은 위치의 품질 속성들만이 품질 변수들로 고려된다. 도 2를 참조하면, SIGs에서 가장 높은 품질 속성들은 가독성(Readability), 성능(performance), 내구성(Durability), 사용성(Usability)이다.

[0070] 품질 변수는 품질 속성을 애플리케이션이 얼마나 만족시키는 지를 나타내며, 실수값을 갖는다. 품질 변수들의 집합 Q는 사용자의 전체 만족도를 나타낸다. 품질 변수들의 집합 Q는 품질 변수들을 결합한 하나의 만족도를 나타내기 위해 필요하다. 전체 품질 변수들에 대하여 전체적인 만족도를 산출하는 전체 만족도 함수 U는 다음의 수학식 3과 같이 계산될 수 있다.

수학식 3

$$U : Q_1 \times Q_2 \times \cdots \times Q_n \times W_1 \times W_2 \times \cdots \times W_n \rightarrow \mathbb{R}$$

[0071]
$$\begin{aligned} U(Q,W) &= U(q_1, q_2, \cdots, q_n, w_1, w_2, \cdots, w_n) \\ &= q_1 \cdot w_1 + q_2 \cdot w_2 + \cdots + q_n \cdot w_n \\ &= \sum_i^n q_i \cdot w_i \end{aligned}$$

[0072] 여기서, Q는 품질 변수들의 집합이고, Q_i는 품질 변수 q_i가 가질 수 있는 값들의 집합이다. W는 가중치들의 집합이고, w_i는 품질 변수 q_i의 가중치이다. n은 품질 변수의 총 개수이다. 가중치들의 합은 1($\sum_i^n w_i = 1$)이어야 하므로, W_i는 [0,1] 사이에 있는 실수 집합을 나타낸다. 품질 변수의 값은 주변 상황과 선택된 대안 기능들에 의해 결정되는 값이다. 가중치는 사용자에 의해 정의되는 값이고, 품질 속성의 우선도를 나타낸다. 이러한 가중치는 실시간으로 사용자에 의해 변화될 수 있는 값이다.

[0073] 다음으로, 주변 상황 변화에 기초하여 소프트골과 동작 목표(operationalizing goal) 간의 관계를 보다 상세하게 살펴보기로 한다.

[0074] 상황 변수(situation variable) s_i는 주변 환경 변화에 대한 정보를 갖고 있다. 상황 변수들은 대안 기능들이 소프트골에 미치는 영향을 결정하며, 이를 표현하면 수학식 4와 같이 표현될 수 있다.

수학식 4

$$f : V \times G \times S_1 \times S_2 \times \cdots \times S_n \rightarrow \mathbb{R}$$

$$\begin{aligned} f_{v_1}^{g_1}(S) &= f_{v_1}^{g_1}(s_1, s_2, \cdots, s_n) \\ f_{v_2}^{g_2}(S) &= f_{v_2}^{g_2}(s_1, s_2, \cdots, s_n) \\ &\cdots \\ f_{v_m}^{g_k}(S) &= f_{v_m}^{g_k}(s_1, s_2, \cdots, s_n) \end{aligned}$$

[0075]

[0076] f는 상황 평가 함수(situation evaluation function)이고, V는 아키텍처 결정 값들의 집합이며, G는 SIGs에 있는 소프트골들의 집합이다. S_i는 상황 변수 s_i가 가질 수 있는 값들의 집합, S는 상황 변수들의 집합이며, R은 실수 집합이다. s_i는 i번째 상황 변수, v_j는 j번째 아키텍처 결정 값, g_k는 k번째 소프트골이다. k는 소프트골들의 개수가 아니라 소프트골을 나타내는 인덱스이다.

[0077] Q_i는 i번째 품질 변수, g_i는 i번째 소프트골, v_i는 i번째 대안 기능, a_i는 i번째 아키텍처 결정 변수, $f_{v_i}^{g_j}(S)$는 대안 기능 v_i와 소프트골 g_j에 대한 상황 평가 함수를 나타낸다. 상황 평가 함수는 동작 목표와 소프트골 간의 직접적인 상호 의존성(interdenpendency)을 정의하는 함수이다.

[0078] 상황 평가 함수들의 개수는 동작 목표들과 소프트골들 간의 직접적인 상호 의존성(interdenpendency)의 개수에 의해 결정된다. 상호 의존성은 동작 목표가 소프트골에 미치는 영향이 있는 경우에 발생한다. 모든 동작 목표들은 대응하는 아키텍처 결정 값을 갖고 있으며, 소프트골과의 내적 관계와 이 내적 관계를 표현하는 상황 평가 함수를 적어도 하나 이상을 갖고 있다.

[0079] 도 7을 참조하면, |F|는 상황 평가 함수들의 개수이고, |V|는 아키텍처 결정 값들의 개수일 때 $|F| \geq |V|$임을 알 수 있다. 이러한 상황 평가 함수들은 아키텍처 결정 값들이 품질 속성에 미치는 영향도를 계산하는데 사용된다.

[0080] 아키텍처 구성요소를 선택하는 문제는 최적화된 아키텍처 인스턴스 또는 아키텍처 결정 값들의 조합을 찾아 해결할 수 있다. 최적화된 조합을 결정하기 위해서는 수학식 2의 만족도 함수 U를 이용할 수 있다. 만족도 함수 U를 계산하기 위해서는 SIGs와 상황 평가 함수가 필요하다.

[0081] 최적화된 아키텍처 인스턴스 또는 아키텍처 결정 변수들의 조합을 찾는 목적은 현재 상황과 사용자 품질 요구에 기반하여 최적의 아키텍처 인스턴스 조합을 찾는 데 있다. 이때 현재 상황은 상황 변수에 의해 표현할 수 있고, 사용자 품질 요구는 품질 속성에 부여된 가중치로 표현할 수 있다.

[0082] 최적의 아키텍처 인스턴스의 조합은 다음의 수학식 5에 의해 표현된다.

수학식 5

$$\begin{aligned} e^* &= \arg\max_{e_i \in E} U(Q, W) \\ &= \arg\max_{e_i \in E} U(Q_{SIG}(A), W) \\ &= \arg\max_{e_i \in E} U(Q_{SIG}(< a_1, a_2, \cdots, a_n >), W) \\ &= \arg\max_{e_i \in E} U(Q_{SIG}(e_i), W) \end{aligned}$$

[0083]

[0084] Q_{SIG}는 SIGs에 기반한 평가 함수이고, $Q_{SIG} : A \rightarrow \hat{Q}$과 같이 정의된다. 여기서 $\hat{Q}$는 <q_1, q_2, ..., q_m>과 같은 모든 품질 변수 q_i를 포함하는 벡터 집합이고, m은 품질 변수들의 개수, A는 <a_1, a_2, ..., a_m>로 표현되는 아키텍처 결정 변수들의 벡터이다. 아키텍처 결정 변수들의 벡터 A는 아키텍처 인스턴스 e_i를 나타낸다. e^*는 가중치들로 표현되는 품질 요구와 현재 상황 하에서 최적의 아키텍처 인스턴스를 의미한다.

[0085] 최적의 아키텍처 인스턴스 e^*를 찾기 위해서는 만족도 함수 U(Q,W)를 SIGs를 이용하여 연산하여야 한다. 만족도 함수 U(Q,W)를 연산하는 방법은 아키텍처 결정 변수들로부터 품질 변수에 이르는 바텀-업(bottom-up) 방식으로 수행된다.

[0086] 아키텍처 인스턴스 e^*는 아키텍처 결정 변수들의 벡터로 표현된다. 또한, 각 아키텍처 인스턴스에 대응하는 동작 목표들의 집합을 생성할 수 있다.

[0087] 도 8은 상황 평가 함수들에 적용하는 결합 규칙을 나타낸 것이다.

[0088] 선택된 동작 목표들로부터 시작하는 모든 소프트골들에 대해서 현재의 상황 변수들의 값에 기초하여 상황 평가 함수를 연산한다. 상황 평가 함수들의 연산 결과는 도 8에 도시된 바와 같이 관련 소프트골들의 상황 평가 함수들의 결과가 더해져 부모 소프트골의 상황 평가 함수의 결과가 된다.

[0089] 상황 평가 함수들이 연산되면, 연산 결과는 부모 소프트골로 전달된다. 자식 소프트골들과 부모 소프트골들 사이의 관계는 세 가지 유형이 있을 수 있다. 세 가지 유형은 다이렉트(direct), AND, OR 관계이다.

[0090] 다이렉트 관계는 자식 소프트골의 값이 부모 소프트골로 바로 전달되는 관계이다. AND 관계는 적어도 2개 이상의 자식 소프트골들이 부모 소프트골과 연관이 있을 경우에 적용이 가능하며, 도 9를 참조하여 AND 관계를 설명하기로 한다.

[0091] 도 9는 AND 관계 결합 규칙을 나타낸 것이다.

[0092] AND 관계는 모든 자식 소프트골들이 만족되는 경우에 부모 소프트골이 만족되는 관계이다. AND 관계에서 모든 자식 소프트골들의 값이 양의 값을 가지는 경우 자식 소프트골들의 모든 값이 결합되어 더해진 후 부모 소프트골의 값이 된다. 또는 하나의 자식 소프트골이 음의 값을 갖거나 0이면, 부모 소프트골의 값은 0이 된다. 그러나 모든 자식 소프트골들의 값이 음의 값이거나 0이면 부모 소프트골의 값은 자식 소프트골들의 모든 값이 결합되어 더해진 값이다. 종래의 NFR 프레임워크에서는 소프트골이 만족되었는지 여부만을 사용한다는 점에서 본 발명과 차이가 있다. 본 발명에서 사용되는 AND 결합 관계는 동적으로 아키텍처를 선택하는 문제에 있어서, 비교 가능한 숫자를 이용함으로써, 사용자가 얼마나 만족하는지를 측정할 수 있다는 점에서 종래 기술과 중요한 차이를 갖고 있다.

[0093] 도 10은 OR 관계 결합 규칙을 나타낸 것이다.

[0094] OR 관계는 적어도 하나의 자식 소프트골이 만족되면 부모 소프트골이 만족되는 관계이다. OR 관계에서 하나 이상의 자식 소프트골이 양의 값을 가지면, 부모 소프트골은 자식 소프트골들의 값들 중에서 최대값을 갖는다. 그러나 모든 자식 소프트골들이 음의 값을 갖거나 0이면, 부모 소프트골은 확정적으로 만족되지 않으므로, 부모 소프트골은 자식 소프트골들 중에서 최소값을 갖는다. 또한, 단 하나의 자식 소프트골이 양의 값을 갖는 경우 부모 소프트골은 자식 소프트골의 양의 값을 갖는다. 이러한 관계 결합 규칙들에 근거하여 최종적으로 품질 변수들의 값이 결정되고, 모든 아키텍처 인스턴스들의 만족도 함수의 결과값인 만족도를 얻을 수 있다.

[0095] 모든 아키텍처 인스턴스들에 대한 만족도를 산출하는 것은 시간이 많이 소요된다. 최적의 아키텍처 인스턴스를 찾는 문제의 복잡도는 아키텍처 결정 변수들의 개수에 의해 결정된다. 따라서 최적의 아키텍처 인스턴스를 찾을 때의 검색 공간의 크기는 지수적으로 증가하며, $O(|E|)=O(\delta^n)$과 같이 표현할 수 있다. 이때 $|E|$는 아키텍처 인스턴스들의 개수이고, δ는 아키텍처 결정 변수가 가질 수 있는 아키텍처 결정 값들의 평균 개수, n은 아키텍처 결정 변수들의 수이다.

[0096] 따라서, 아키텍처 결정 변수들이 많은 경우 모든 아키텍처 인스턴스들에 대한 만족도를 산출하는 것은 상태 폭발 문제(state explosion problem)를 야기할 수 있으므로 바람직하지 않다. 본 발명의 일 실시예에서는 그리디 알고리즘(greedy algorithm), 다이나믹 프로그래밍(dynamic programming), 유전 알고리즘을 이용할 수 있다.

[0097] 도 11과 도 12는 선택된 대안 기능들과 대안 기능들에 대응하는 애플리케이션 구성요소 간의 관계를 나타낸 것이다.

[0098] 도 11을 참조하면, "RichGUI" 대안 기능과 "High Contrast Display" 대안 기능은 각각 "RichGUI" 구성요소와 "High Contrast Transformer" 구성요소에 대응한다. 또한, "Videotelephony" 대안 기능은 "Image Compressor" 구성요소와 "Audio Compressor" 구성요소에 대응한다.

[0099] 만일 주변 상황이나 사용자 품질 요구가 변화하여 "RichGUI" 대안 기능이 "SimpleGUI" 대안 기능으로, "High

Contrast Display" 대안 기능이 "Normal Display" 대안 기능으로, "Videotelephony" 대안 기능이 "Simple Messaging" 대안 기능으로 변경되면, 각 대안 기능들에 대응하는 구성요소들도 변화하게 되어 도 11에 도시된 구성요소로 재구성된다.

[0100] 도 13은 본 발명의 일 실시예에 따른 애플리케이션 구성요소 선택 방법의 흐름도이다.

[0101] 1310 단계에서 구성요소 선택 장치는 대안 타입별로 구분된 복수의 대안 기능들 중에서 각 대안 타입별로 하나의 대안 기능을 선택함으로써 구성가능한 모든 대안 기능 조합들을 산출한다.

[0102] 1320 단계에서 구성요소 선택 장치는 산출된 대안 기능 조합들에 포함된 대안 기능들이 품질 속성에 미치는 영향도를 결합하여 대안 기능 조합들 각각의 만족도를 산출한다.

[0103] 1330 단계에서 구성요소 선택 장치는 산출된 만족도들 중에서 가장 만족도가 큰 대안 기능 조합에 기초하여 어플리케이션 내의 구성요소를 선택한다.

[0104] 도 14는 그리디 알고리즘을 이용하여 만족도를 산출하는 방법을 개념적으로 도시한 것이다.

[0105] 그리디 알고리즘은 SIGs에서 각 아키텍처 결정 변수에 대응하는 대안 기능들 중에서 가장 큰 대안 기능을 선택하고, 이로부터 최적의 조합을 결정한다. 단 하나의 아키텍처 결정 변수에 대해서만 상황 평가 함수를 연산하므로, 다른 아키텍처 결정 변수에 대한 상황 평가 함수는 연산되지 않는다.

[0106] 모든 각각의 동작 목표들의 값을 결정할 수 있다. 예를 들어 a=-0.5, b=0.0, c=-0.5, d=0.5라고 하면, 각 대안 타입 내에서 가장 최선의 선택을 할 수 있게 되며, t_1의 경우에는 b, t_2의 경우에는 d를 선택하여 값이 0이 된다. 그러나 a와 c를 선택하는 경우 값이 1.5가 되므로 가장 최적의 조합이 된다.

[0107] 본 발명의 다른 실시예에 따르면, 유전 알고리즘을 이용하여 동적으로 아키텍처의 구성요소를 선택할 수 있다.

[0108] 유전 알고리즘은 솔루션 공간에서 근사한 솔루션을 찾는 메타휴리스틱 검색 방법으로 최적의 조합을 찾는 검증된 방법이기도 하다.

[0109] 유전 알고리즘에서 해결하고자 하는 문제는 유전 서열로 표현된다. 이 유전 서열을 염색체(chromosome)라고 한다. 염색체 표현을 사용하면, 유전 알고리즘을 사용할 때 염색체의 초기 개수를 생성할 수 있다.

[0110] 도 15는 유전 알고리즘을 나타낸 것이다.

[0111] 유전 알고리즘의 과정은 (1) 부모 염색체들을 선택하고, 교배(crossover)를 한다. (2) 염색체들을 선택하고, 특정 돌연변이 확률에 기초하여 염색체를 돌연변이 시킨다. (3) 자식 염색체의 적합도(fitness)를 평가한다. (4) 자식 염색체로부터 다음 세대의 염색체를 선택한다. (1)~(4)의 과정을 특정 종료 조건이 만족될 때까지 반복한다. 특정 종료 조건은 일정 세대까지만 상기 과정을 진행하는 조건일 수 있다.

[0112] 이러한 유전 알고리즘을 최적의 아키텍처 인스턴스를 구하는데 사용하기 위해 아키텍처 인스턴스들을 염색체들로 코드화하고, 적합 함수(fitness function)를 디자인하며, 교배와 돌연변이 연산자를 결정한다.

[0113] 아키텍처 인스턴스 집합으로부터 최적의 아키텍처 인스턴스를 찾기 위해 아키텍처 인스턴스들을 염색체로 인코딩한다.

[0114] 도 16은 아키텍처 결정 변수들을 사용하여 아키텍처 인스턴스들을 인코딩하는 방법을 도시한 것이다.

[0115] i번째 아키텍처 결정 변수는 i번째 염색체의 디지트(digit)에 대응한다. 0,1,2,...,n으로 표시된 숫자는 특정 아키텍처 결정 변수에 속한 아키텍처 결정 값들을 구별하는데 사용되는 식별자이다.

[0116] 이하에서는 교배와 돌연변이 연산자를 설계하여 자식 염색체를 생성하는 방법을 살펴보기로 한다. 다양한 교배와 돌연변이 연산자들 중에서 본 실시예에서는 2점 교배와 디지트방향 확률 돌연변이를 사용한다.

[0117] 도 17은 2점 교배의 일 실시예를 나타낸 것이다.

[0118] 2점 교배는 두 개의 부모 염색체를 선택하고, 각 염색체 상에 같은 위치의 두 점을 임의로 선택한다. 그런 다음 두 위치 사이의 염색체 간 디지트를 교환한다. 2점 교배는 다른 교배 방법보다 부모 염색체의 특성을 더 많이 포함한다. 유사한 염색체들 간의 교배는 유사한 값을 가질 것이다. 교배 확률이 P_c=0.5인 두 개의 부모 염색체를 교배하는 경우, 전체 염색체의 반이 부모 염색체로 선택되고, 교배 연산자는 부모 염색체와 같은 수의 자식 염색체를 생성한다.

[0119] 도 18은 돌연변이의 일 실시예를 나타낸 것이다.

[0120] 교배를 수행한 후에 유전 알고리즘은 돌연변이 확률 P_m으로 돌연변이를 수행한다. 돌연변이 확률 P_m^2로 교배과정에서 생성된 자식 염색체의 모든 디지트가 임의의 값으로 바뀌게 된다. 임의의 값은 아키텍처 결정 변수가 가질 수 있는 범위 내이고, 일반적으로 $P_m=1/n$(n은 염색체의 길이)과 같이 연산된다.

[0121] 만일 돌연변이 확률이 너무 높으면 부모 염색체의 특성을 보전하기 힘들고, 돌연변이 확률이 너무 낮으면, 국소적인 최적화에 한정된다. 교배와 돌연변이에 의해서 생성된 자식 염색체는 다음 세대 염색체의 후보군이 된다.

[0122] 교배와 돌연변이가 수행된 후에는 선택과정을 수행한다. 선택 과정에서 유전 알고리즘은 모든 자식 염색체들의 적합도를 평가하고, 최적의 값이 아닌 값을 가진 염색체들을 배제한다. 유전 알고리즘에서는 수학식 1에 제시한 만족도 함수 U가 염색체를 평가하는 적합 함수(fitness function)로 사용된다. 또한, 토너먼트 선택 전략이 사용되며, 이 전략은 이전 과정에서 생성된 새로운 세대의 염색체로부터 가장 적합한 염색체를 선택하는 방법이다. 이러한 선택 전략은 최초 염색체 수와 같은 수의 염색체를 다음 세대에도 보존할 수 있도록 한다. 염색체 개수는 유전 알고리즘의 효율성을 결정하는 요소이다. 염색체 개수가 너무 작으면, 검색 공간을 탐색하는 것이 효율적이지 않고, 너무 큰 염색체 개수도 효율적이지 않다.

[0123] 본 발명의 일 실시예에 따라 생성되는 VQHQM은 열화 영상의 화질을 평가하는 지수 중의 하나이다. VQHQM은 양자화된 영역에 속하는 변이 벡터의 수를 계산하여 얻은 히스토그램을 이용하여 산출된다. 각각의 화질 평가 지수와 DMOS의 피어슨 상관 계수를 성능 비교에 사용한 결과, 본 발명의 일 실시예에 따라 생성되는 VQHQM은 종래의 화질 평가 지수보다 우수한 성능을 나타낸다는 것을 알 수 있다.

[0124] 본 발명의 실시예들은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기된 하드웨어 장치는 본 발명의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

[0125] 이상과 같이 본 발명에서는 구체적인 구성 요소 등과 같은 특정 사항들과 한정된 실시예 및 도면에 의해 설명되었으나 이는 본 발명의 보다 전반적인 이해를 돕기 위해서 제공된 것일 뿐, 본 발명은 상기의 실시예에 한정되는 것은 아니며, 본 발명이 속하는 분야에서 통상적인 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다. 따라서, 본 발명의 사상은 설명된 실시예에 국한되어 정해져서는 아니되며, 후술하는 특허청구범위뿐 아니라 이 특허청구범위와 균등하거나 등가적 변형이 있는 모든 것들은 본 발명 사상의 범주에 속한다고 할 것이다.

[0126] 이제까지 본 발명에 대하여 그 바람직한 실시예들을 중심으로 살펴보았다. 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 본 발명이 본 발명의 본질적인 특성에서 벗어나지 않는 범위에서 변형된 형태로 구현될 수 있음을 이해할 수 있을 것이다. 그러므로 개시된 실시예들은 한정적인 관점이 아니라 설명적인 관점에서 고려되어야 한다. 본 발명의 범위는 전술한 설명이 아니라 특허청구범위에 나타나 있으며, 그와 동등한 범위 내에 있는 모든 차이점은 본 발명에 포함된 것으로 해석되어야 할 것이다.

도면의 간단한 설명

[0127] 도 1은 본 발명의 일 실시예에 따른 애플리케이션 구성요소 선택 장치의 블록도이다.

[0128] 도 2는 SIGs 저장부(110)가 저장하고 있는 SIGs의 일 실시예를 나타낸 것이다.

[0129] 도 3은 주변의 상황을 고려하여 선택된 대안 기능들의 사용자 만족도를 평가하는 일 실시예를 도시한 것이다.

[0130] 도 4는 도 3의 상황 변수가 변화된 경우에 대안 기능들의 만족도를 평가하는 다른 실시예를 도시한 것이다.

[0131] 도 5는 사용자 품질 요구가 변화된 경우에 대안 기능들의 만족도를 평가하는 또 다른 실시예를 도시한 것이다.

[0132] 도 6은 아키텍처 결정 변수들, 대안 타입, 상황 변수들, 및 품질 속성들 간의 상호 관계를 나타내는 도면이다.

[0133] 도 7은 본 발명의 일 실시예에 따라 동적으로 품질 속성에 대응하는 스코어들을 연산하는 방법을 개념적으로 나타낸 것이다.

[0134] 도 8은 상황 평가 함수들에 적용하는 결합 규칙을 나타낸 것이다.

[0135] 도 9는 AND 관계 결합 규칙을 나타낸 것이다.

[0136] 도 10은 OR 관계 결합 규칙을 나타낸 것이다.

[0137] 도 11과 도 12는 선택된 대안 기능들과 대안 기능들에 대응하는 애플리케이션 구성요소 간의 관계를 나타낸 것이다.

[0138] 도 13은 본 발명의 일 실시예에 따른 애플리케이션 구성요소 선택 방법의 흐름도이다.

[0139] 도 14는 그리디 알고리즘을 이용하여 만족도를 산출하는 방법을 개념적으로 도시한 것이다.

[0140] 도 15는 유전 알고리즘을 나타낸 것이다.

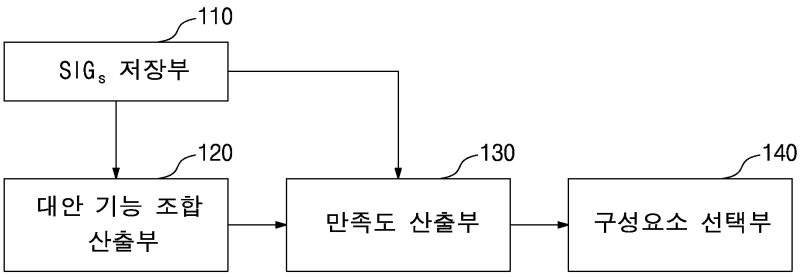
[0141] 도 16은 아키텍처 결정 변수들을 사용하여 아키텍처 인스턴스들을 인코딩하는 방법을 도시한 것이다.

[0142] 도 17은 2점 교배의 일 실시예를 나타낸 것이다.

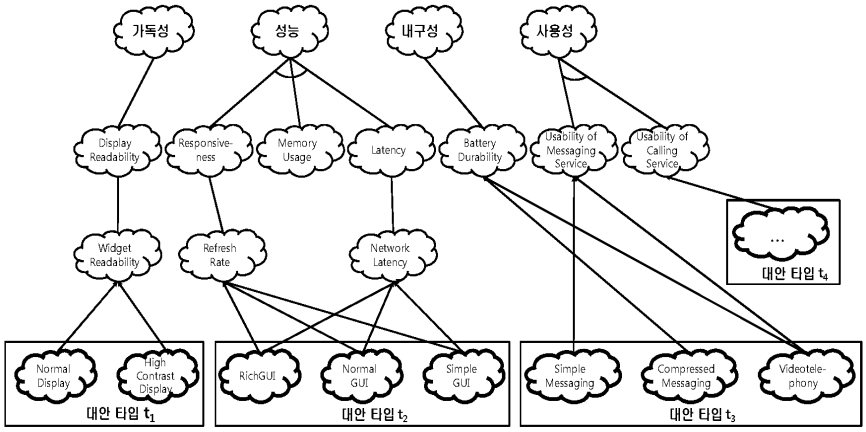
[0143] 도 18은 돌연변이의 일 실시예를 나타낸 것이다.

도면

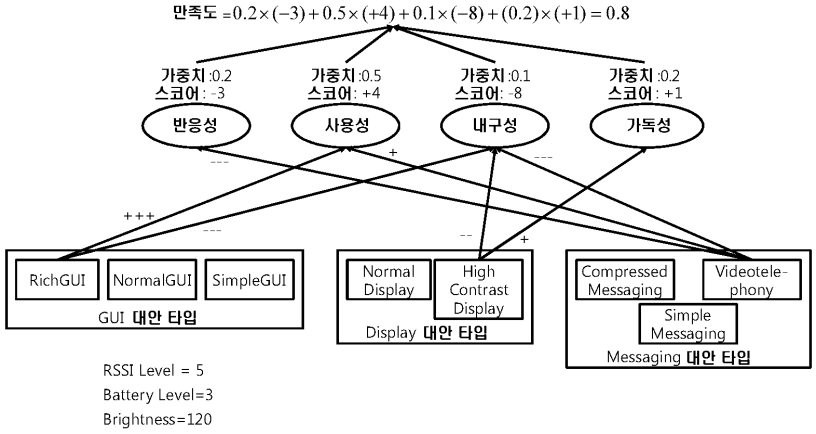
도면1



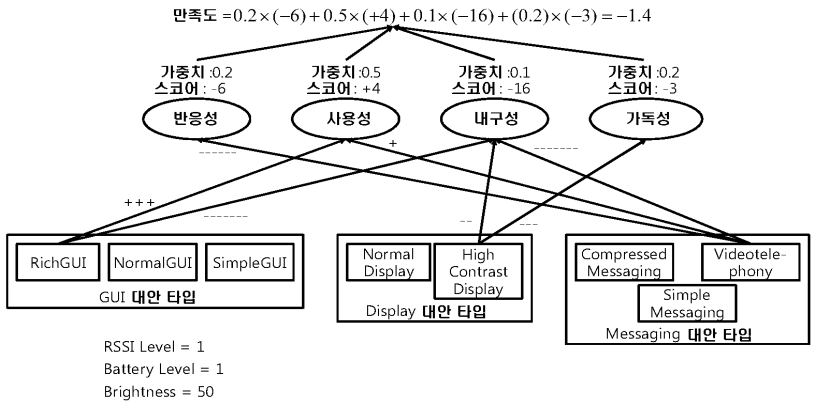
도면2


가독성
성능
내구성
사용성
Display Readability
Responsive-ness
Memory Usage
Latency
Battery Durability
Usability of Messaging Service
Usability of Calling Service
Widget Readability
Refresh Rate
Network Latency
...
대안 타입 t_4
Normal Display
High Contrast Display
대안 타입 t_1
RichGUI
Normal GUI
Simple GUI
대안 타입 t_2
Simple Messaging
Compressed Messaging
Videotele-phony
대안 타입 t_3

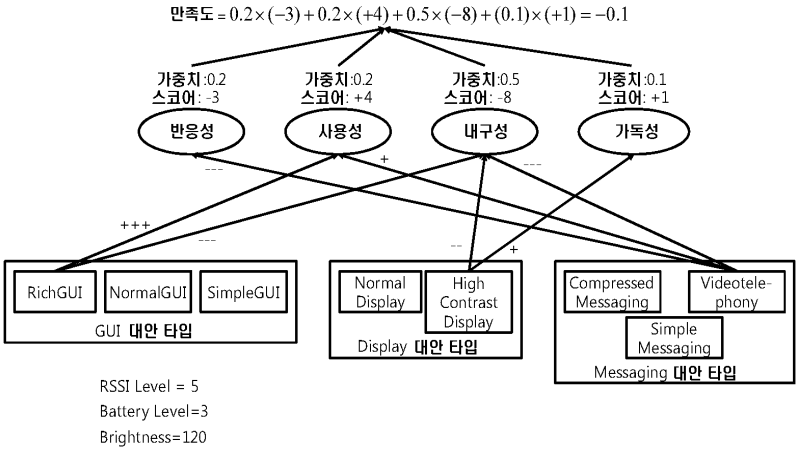
도면3


만족도 $=0.2\times(-3)+0.5\times(+4)+0.1\times(-8)+(0.2)\times(+1)=0.8$
가중치 :0.2 스코어 : -3
가중치 :0.5 스코어 : +4
가중치 :0.1 스코어 : -8
가중치 :0.2 스코어 : +1
반응성
사용성
내구성
가독성
RichGUI
NormalGUI
SimpleGUI
GUI 대안 타입
Normal Display
High Contrast Display
Display 대안 타입
Compressed Messaging
Videotele-phony
Simple Messaging
Messaging 대안 타입
RSSI Level = 5
Battery Level=3
Brightness=120

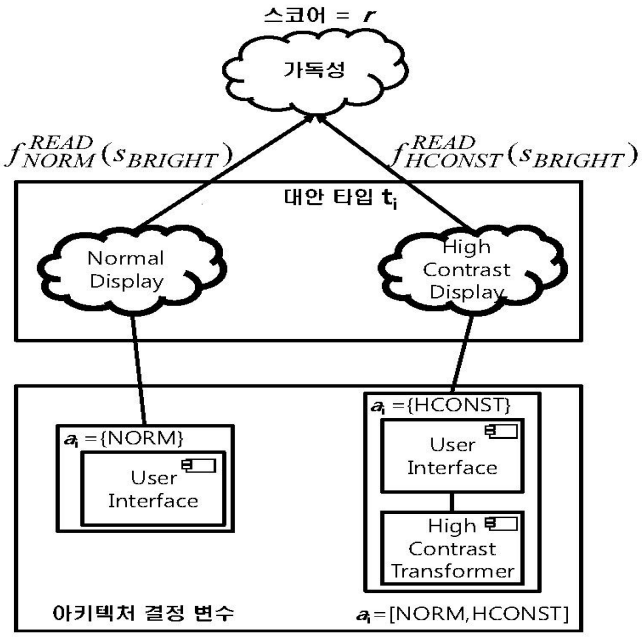
도면4


만족도 $=0.2\times(-6)+0.5\times(+4)+0.1\times(-16)+(0.2)\times(-3)=-1.4$
가중치 :0.2 스코어 : -6
가중치 :0.5 스코어 : +4
가중치 :0.1 스코어 : -16
가중치 :0.2 스코어 : -3
반응성
사용성
내구성
가독성
RichGUI
NormalGUI
SimpleGUI
GUI 대안 타입
Normal Display
High Contrast Display
Display 대안 타입
Compressed Messaging
Videotele-phony
Simple Messaging
Messaging 대안 타입
RSSI Level = 1
Battery Level = 1
Brightness = 50

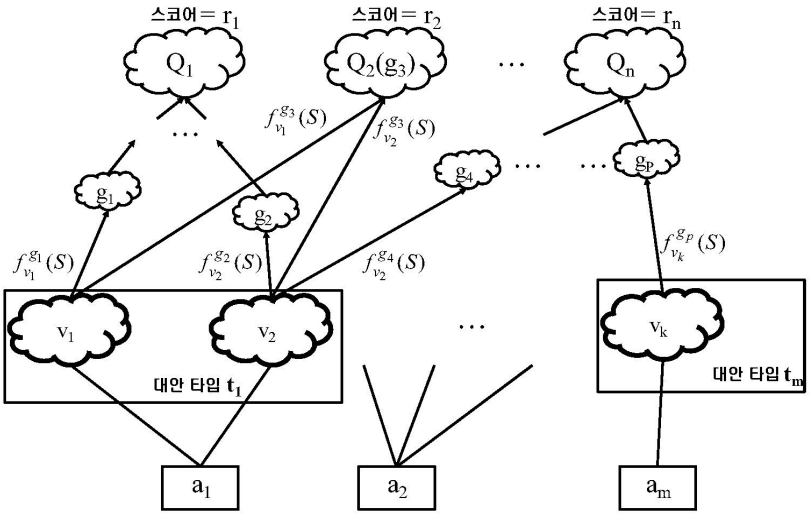
도면5


만족도 = 0.2×(−3) + 0.2×(+4) + 0.5×(−8) + (0.1)×(+1) = −0.1
가중치:0.2 스코어: -3
가중치:0.2 스코어: +4
가중치:0.5 스코어: -8
가중치:0.1 스코어: +1
반응성
사용성
내구성
가독성
RichGUI
NormalGUI
SimpleGUI
GUI 대안 타입
Normal Display
High Contrast Display
Display 대안 타입
Compressed Messaging
Videotele-phony
Simple Messaging
Messaging 대안 타입
RSSI Level = 5
Battery Level=3
Brightness=120

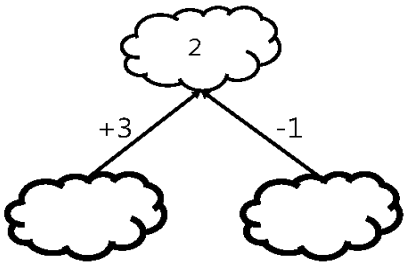
도면6


스코어 = r
가독성
$f^{READ}_{NORM}(s_{BRIGHT})$
$f^{READ}_{HCONST}(s_{BRIGHT})$
대안 타입 t_i
Normal Display
High Contrast Display
a_i ={NORM}
User Interface
a_i ={HCONST}
User Interface
High Contrast Transformer
아키텍처 결정 변수
a_i=[NORM,HCONST]

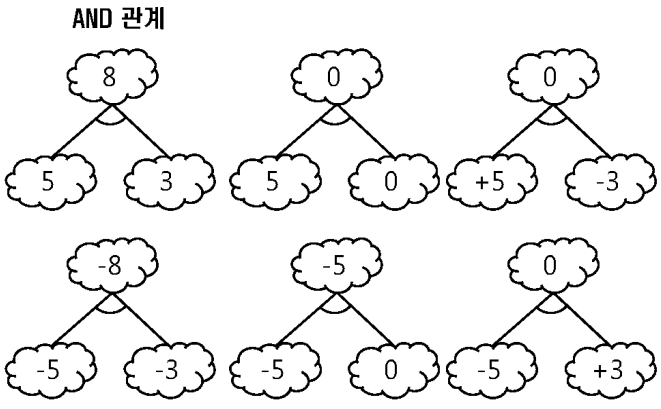
도면7


스코어= r1
스코어= r2
스코어= rn
Q1
Q2(g3)
Qn
g1
g2
g4
gp
v1
v2
vk
대안 타입 t1
대안 타입 tm
a1
a2
am

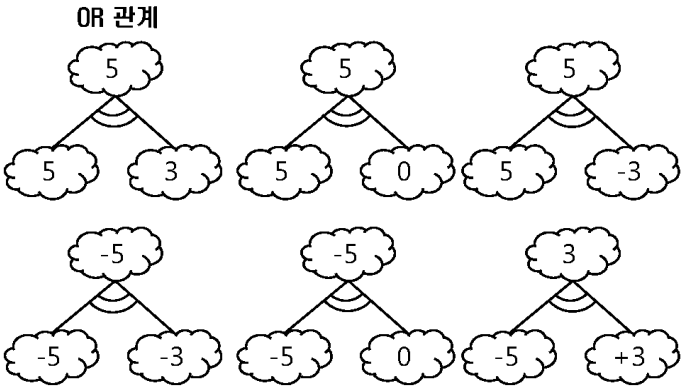
도면8


2
+3
-1

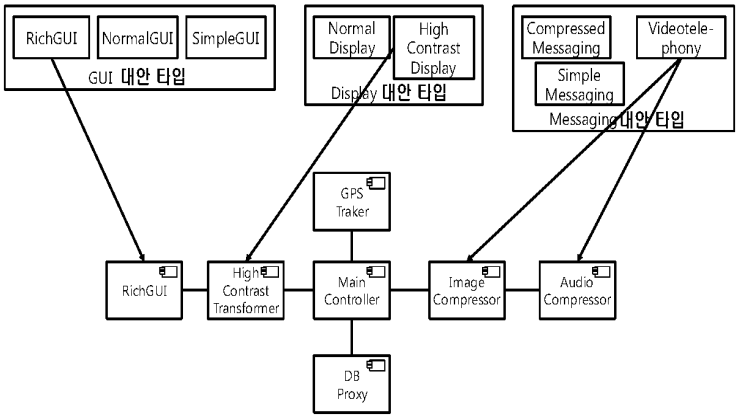
도면9


AND 관계
8
5
3
0
5
0
0
+5
-3
-8
-5
-3
-5
0
-5
0
-5
+3

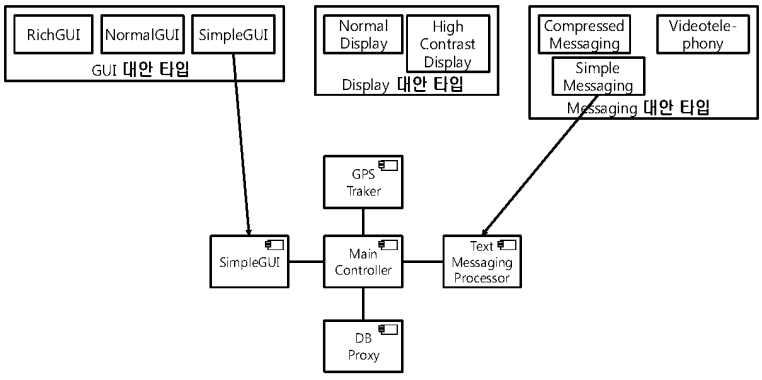
도면10



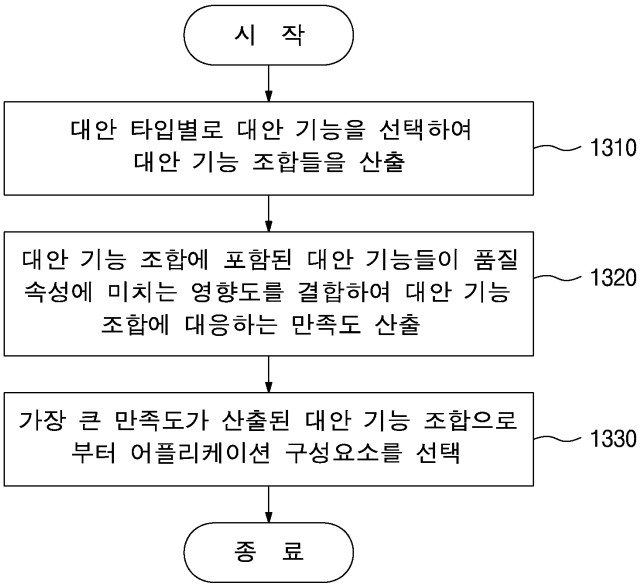
도면11



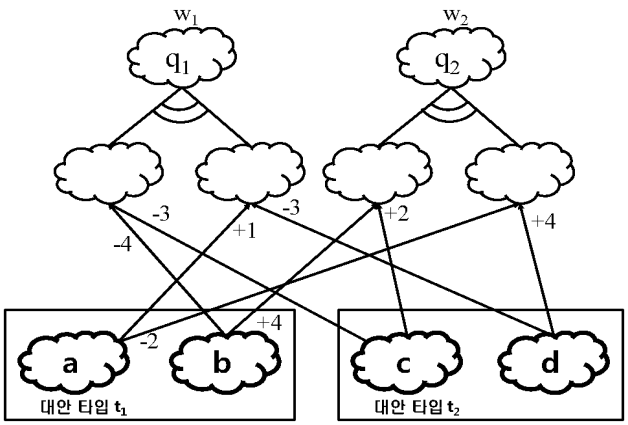
도면12



도면13



도면14



도면15

Algorithm 1 A genetic algorithm

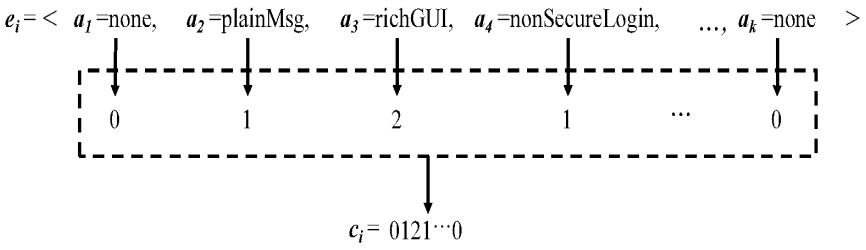
```
1: procedure GA(genes)
2:     Choose an initial population of chromosomes
3:     while termination condition is not satisfied do
4:         repeat
5:             perform crossover with crossover probability P_c
6:             perform mutation with mutation probability P_m
7:             evaluate fitness of offspring
8:         until sufficient offspring created
9:         select population of the next generation
10:    end while
11: end procedure
```

도면16

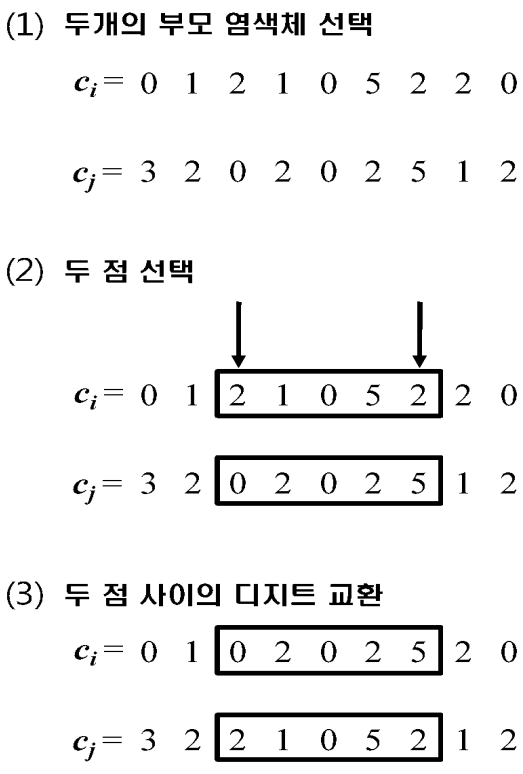
a_i = i번째 아키텍처 결정 변수

e_i = i번째 아키텍처 인스턴스

c_i = i번째 염색체

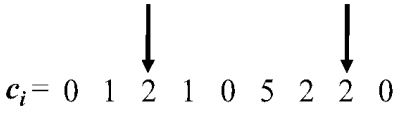


도면17



도면18

(1) 돌연변이 확률을 가진 디지트 선택



(2) 선택된 디지트들을 임의의 값으로 변경