SoC 기반 상황 인식 시스템 구조

An SoC-based Context-Aware System Architecture

이건명*, 손봉기*, 김종태**, 이승욱**, 이지형**, 전재욱**, 조준동**
* 충북대학교 전기전자컴퓨터공학부
** 성균관대학교 정보통신공학부
Keon Myung Lee*, Bong Ki Sohn*, Jong Tae Kim**, Seung Wook Lee**, Ji Hyong Lee**, Jae Wook Jeon**, Jun Dong Cho**
*School of Electric and Computer Engineering, Chungbuk National University, Korea
**School of Information and Communication Engieering, SungKyunKwan University, Korea

요 약

상황 인식(context-awrare)은 인간-컴퓨터 상호작용의 단점을 극복하기 위한 방법으로써 많은 주목을 받고 있다. 이 논문에서는 SoC(System-on-a-Chip)로 구현될 수 있는 상황 인식 시스템 구조를 제안한다. 제안한 구조는 센서 추상화, 컨텍스트 변경에 대한 통지 메커니즘, 모듈식 개발, *if-then* 규칙을 이용한 쉬운 서비스 구성과 유연한 상황 인식 서비스 구현을 지원한다. 이 구조는 통신 모듈, 처리 모듈, 블랙보드를 포함하는 SoC 마이크로프로세서 부분과 규칙 기반 시스템 모듈을 구현한 하드웨어로 구성된다. 규칙 기반 시스템 하드웨어는 모든 규칙의 조건부에 대해 매칭 연산을 병렬로 수행하고, 규칙의 결론부는 마이크로프로세서에 내장된 행위 모듈을 호출함으로써 작업을 수행한다. 제안한 구조의 SoC 시스템은 SystemC SoC 개발 환경에서 설계되고, 성공적으로 테스트되었다. 제안한 SoC 기반의 상황 인식 시스템 구조는 주거 환경에서 컨텍스트를 인식하여 노인을 보조하는 지능형 이동 로봇 등에 적용될 수 있을 것으로 기대된다.

1. 서 론

사용자는 작업 실행 방법에 대해 상세히 입력해야 하는 컴퓨터 사용에 번거로움을 느끼고 있다. 상황 인식 컴퓨팅은 인간-컴퓨터 상호작용의 단점을 극복하기 위한 방법으로 주목을 받고 있다. 컨텍스트(context)는 개체의 상태를 특징짓는데 사용될 수 있는 정보를 말한다[1]. 여기에서 개체는 사용자와 어플리케이션을 포함해서 사람, 장소 또는 사용자와 어플리케이션 간의 상호작용에 관련된 객체가 될 수 있다. 전형적인 컨텍스트 정보에는 위치, 신원, 시간, 행위 등이 있는데, 이러한 정보를 이용하여 컴퓨터는 *who, what, when, where*에 대해 응답할 수 있다. 컨텍스트를 고려하여 사용자에게 서비스나 정보를 제공하는 시스템을 상황 인식 시스템이라 한다. 컨텍스트는 상황 인식 어플리케이션 개발하는데 있어 여러 가지 방법으로 사용될 수 있다. Dey가 제시한 상황 인식 어플리케이션의 특징은 다음과 같이 분류될 수 있다[1]:

- 컨텍스트에 따라 자동으로 사용자에게 정보나 서비스를 제공하는 컨텍스트 기반의 정보 및 서비스 제공
- 컨텍스트에 따라 자동으로 서비스를 실행하거나 수정하는 컨텍스트 기반의 서비스 자동 실행
- 디지털 데이터를 사용자의 컨텍스트와 연관시켜서 차후 검색을 위한 컨텍스트 기반의 정보 증강

다양한 상용 완제품 센서들이 나오고, 강력하고 네트워크화된 컴퓨터가 보편화되고, 이동 컴퓨팅 디바이스가 대중화됨에 따라, 상황 인식 어플리케이션에 대한 수요가 증가되고 있다. 상황 인식 어플리케이션에서는 컨텍스트 정보를 수집하는 센서가 중요한 역할을 한다. 컨텍스트 정보는 카메라, 압력 센서, 능동 배지 등과 같은 전형적이지 않은 디바이스로부터 추출되기 때문에, 많은 상황 인식 어플리케이션은 센서와 밀접하게 결합되어 개발되어 왔다. 이러한 구현 방법은 기존의 컴포넌트를 재사용하거나 대체하기 어렵기 때문에 소프트웨어 공학적 측면에서 좋은 방법이 아니다. 경우에 따라서는 컨텍스트를 다른 형태의 컨텍스트로 변경하기 위한 컨텍스트의 해석이 필요할 때도 있고, 상황 인식 서비스를 위해 컨텍스트가 변경될 때 이를 통지해야 하는 경우도 있다. 앞으로 이동 디바이스나

본 논문은 21C 프론티어 "인간기능 생활지원 지능로봇 기술개발" 과제에 의하여 지원되었으며 이에 감사드립니다.

이동 로봇에 내장된 상황 인식 어플리케이션이 많이 등장할 것이다. 따라서 상황 인식 어플리케이션을 용이하게 구축할 수 있는 환경이 필요하다. 이러한 관점에서 이동 디바이스나 이동 로봇을 위한 SoC에 구현될 수 있는 상황 인식 시스템 구조를 제안한다.

이 논문의 구성은 다음과 같다. 2장에서는 상황 인식 시스템 구조에 대한 관련 연구를 살펴보고, 3장에서는 새로운 상황 인식 시스템 구조를 제안한다. 4장에서는 제안한 구조의 SoC 하드웨어 구현에 대해 설명하고, 5장에서는 결론을 맺는다.

2. 관련 연구

다양한 상황 인식 어플리케이션에 적용할 수 있는 많은 상황 인식 어플리케이션 지원 구조가 개발되었지만, 대부분은 상황 인식 시스템의 일부 특징만을 구현하고 있다.

Stick-e Notes 시스템[2]은 상황 인식 어플리케이션을 지원하기 위한 일반적인 프레임워크로, 사용자는 *if-then* 규칙을 사용하여 상황 인식 서비스를 쉽게 구축할 수 있다. 이 시스템은 일반 사용자를 위한 시스템이기 때문에, 서비스 기술 규칙을 작성할 때의 의미(semantics)는 상당히 제한적이다.

CoolTown[3]은 사람, 장소, 디바이스 등의 실세계 객체를 Web presence라고 하는 웹 페이지에 사상(mapping)할 수 있는 상황 인식 어플리케이션 지원 기반구조(infrastructure)이다. 이 시스템은 획득한 컨텍스트 정보에 따라 필요한 경우에는 Web presence 갱신한다. 그러나 컨텍스트에 기반한 서비스의 자동 실행이나 컨텍스트 기반의 정보 증강과 같은 상황 인식 특징은 지원하지 않는다.

Schilit의 System Architecture[4]는 사용자와 디바이스의 컨텍스트에 초점을 맞춘 상황 인식 이동 컴퓨팅을 지원하는 구조이다. 이 시스템 구조는 디바이스 상태와 기능을 관리하는 디바이스 에이전트(device agent), 사용자 선호도를 관리하는 사용자 에이전트(user agent)와 사용자와 디바이스의 위치 정보를 관리하는 능동 맵(active map)으로 구성된다. 그러나 위치 컨텍스트만을 사용하는 사용자 에이전트와 디바이스 에이전트가 다른 형태의 컨텍스트를 사용하기 위해서는 재작성되어야 하기 때문에, 새로운 컨텍스트나 센서를 추가하거나 교체하여 시스템을 확장하기 어렵다.

TEA 프로젝트[6]는 개인용 이동 디바이스에서 컨텍스트를 인식할 수 있는 구조를 제안했다. 이 구조에서의 컨텍스트 인식은 블랙보드 모델에 의해 이루어진다. 먼저, 센서로부터 획득한 컨텍스트를 블랙보드에 추가한다. 컨텍스트 해석기는 이를 요약하거나 해석하여 다시 블랙보드에 저장하고, 어플리케이션은 관련된 컨텍스트를 참조한다.

Dey의 Context Toolkit[1]은 상황 인식 어플리케이션의 구축과 실행을 지원하는 프레임워크이다. 이 프레임워크는 센서로부터 컨텍스트를 얻는 컨텍스트 위젯(context widget), 컨텍스트를 해석하기 위한 컨텍스트 해석기, 관련 컨텍스트를 통합하기 위한 컨텍스트 통합기(aggregator), 어플리케이션에서 재사용 가능한 상황 인식 행위나 서비스를 제공하기 위한 컨텍스트 서비스, 자원 검색을 위한 검색기(discoverer)등의 구성 요소로 이루어진다. 상황 인식 어플리케이션 개발자는 라이브러리 함수로 구현되어 있는 구성 요소를 사용하거나 수정할 수 있다.

이 논문에서 제안하는 상황 인식 시스템 구조는 상황 인식 서비스에서 센서 데이터 처리를 분리하는 센서 추상화, 컨텍스트 변경 통지, 유연한 센서와 컨텍스트 추가 및 교체를 지원하는 시스템 진화(system evolution), 모듈식 구현 방법에 의한 쉬운 어플리케이션 구축, 이동 디바이스나 이동 로봇의 내부 프로세서(SoC)의 구현을 지원한다.

3. SoC 기반의 상황 인식 시스템 구조

제안하는 상황 인식 시스템 구조는 그림 1과 같이 통신 모듈, 처리 모듈, 규칙 기반 시스템과 블랙보드로 구성된다. 통신 모듈은 외부 및 내부 메시지를 수신하여 해당되는 처리 모듈로 전송하고, 시스템에 의해 생성된 결과를 외부 시스템에 송신한다. 처리 모듈은 컨텍스트 획득 모듈, 질의 처리 모듈, 행위 모듈, 작업 처리 모듈을 포함한다. 규칙 기반 시스템은 *if-then* 규칙을 관리하고, 다음 실행할 규칙을 결정한다. 블랙보드는 다른 모듈들이 데이터와 정보를 공유할 수 있는 공유 데이터 저장소이다. 블랙보드의 일부는 규칙 기반 시스템의 작업 메모리(working memory)의 역할을 한다.

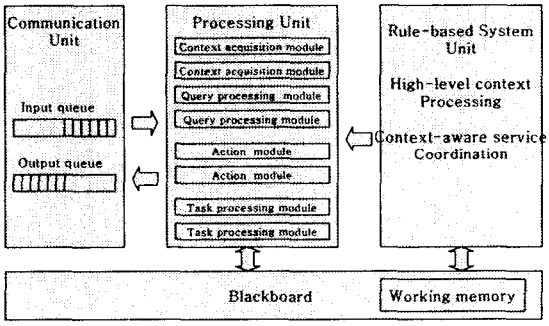

그림 1. SoC 기반의 상황 인식 시스템 구조

제안한 구조는 상황 인식 서비스와 센서 데이터 처리를 독립적인 처리 모듈에서 수행하는 전략을 사용한다. 이러한 전략은 어플리케이션으로부터 센서의 세부사항을 분리하여, 어플리케이션이 관심 있는 컨텍스트만을 처리하도록 한다. 또한, 상황 인식 어플리케이션 개발자가 어떤 상황에서 어떤 행위를 실행할 것인가를 나타내는 *if-then* 규칙을 작성할 수 있기 때문에, 상황 인식 서비스를 쉽게 구축할 수 있다. 규칙 기반 시스템은 *if-then* 규칙을 관리하고, 규칙의 조건부에 대한 매칭 연산을 수행하지만, 규칙의 결론부는 처리 모듈 내의 행위 모듈에 의해 실행된다.

통신 모듈(Communication Unit)

통신 모듈은 입력 메시지 큐와 출력 메시지 큐를 포

함하고, 사건 지향적(event-driven)으로 동작한다. 각 메시지 큐는 다음과 같은 과정을 통해 메시지를 전송한다:

do
fetch a message from the queue
find the processing module corresponding to the message
activate the processing module with the message
while (true)

제안한 구조에서는 센서 메시지, 질의 메시지, 작업 메시지, 내부 모듈 호출 메시지를 처리한다. 센서 메시지(sensor message)는 외부 센서로부터 수신한 센서 데이터를 캡슐화한 것으로, 해당되는 컨텍스트 획득 모듈(context acquisition module)로 전송되어 컨텍스트 정보가 추출된다. 질의 메시지(query message)는 다른 시스템의 정보를 얻기 위해 상황 인식 어플리케이션이 생성하는 메시지이다. 시스템에 질의 메시지가 수신되면, 관련된 질의 처리 모듈(query processing module)로 넘겨진다. 외부 시스템이 수신 시스템에게 어떤 작업을 수행해줄 것을 요청하는 작업 메시지(task message)는 적절한 작업 처리 모듈(task processing module)로 전달되어 처리된다. 내부 모듈 호출 메시지(internal module call message)는 다른 모듈이 처리 모듈을 호출하는 요청 메시지이다. 처리 모듈 간의 밀결합(tight coupling)을 피하기 위해, 내부 모듈 호출 메시지를 사용해 간접적으로 처리 모듈을 호출한다. 제안한 구조는 처리 모듈을 사용해 주변 장치를 제어하고, 통신 모듈을 통해 외부 시스템이나 센서에 명령이나 메시지를 전송한다. 전송 메시지는 출력 메시지 큐에 추가되고, 출력 메시지 큐 관리자는 각각의 메시지를 차례대로 처리하기 위해 적절한 처리기(handler)를 호출한다. 통신 모듈은 어떤 모듈이 어떤 종류의 메시지를 처리하는 지에 대한 정보를 옐로우 페이지(yellow page)에 유지하고 관리한다. 즉, 새로운 메시지가 추가될 때마다 메시지와 이를 처리하는 처리 모듈의 정보가 옐로우 페이지에 추가된다.

처리 모듈(Processing Unit)

처리 모듈은 컨텍스트 획득 모듈, 질의 처리 모듈, 행위 모듈, 작업 처리 모듈로 구성된다. 컨텍스트 획득 모듈(Context acquisition module)은 센서 데이터와 블랙보드의 사용 가능한 정보에서 컨텍스트 정보를 추출한다. 제안한 구조는 컨텍스트 변경이 처리 모듈이나 *if-then* 규칙에 통지될 수 있게 한다. 컨텍스트 획득 모듈은 통지될 처리 모듈 리스트와 규칙에 대한 플래그 리스트를 이용해 통지 메커니즘을 구현한다. 컨텍스트 변경을 *if-then* 규칙에 통지하기 위해 컨텍스트 변경을 규칙에 알리는 역할을 하는 부가적인 플래그를 규칙의 조건부에 추가한다. 관련된 컨텍스트가 변경되면, 컨텍스트 획득 모듈은 컨텍스트와 관련된 플래그 값을 셋팅한다. 만약 어떤 규칙이 선택되어 실행되면 행위 모듈(action module)은 그 규칙의 모든 통지 플래그 값을 리셋한다. 컨텍스트 획득 모듈은 센서 데이터 처리 루틴과 처리 모듈에 대한 내부 모듈 호출 메시지를 생성하고 블랙보드의 규칙 플래그를 수정하는 갱신 루틴(update routine)을 포함한다. 컨텍스트 획득 모듈은 컨텍스트 변경을 통지하기 위해 이전의 센서 데이터와 이에 대응하는 컨텍스트 및 컨텍스트 갱신 시간 정보를 유지한다. 질의 처리 모듈은 블랙보드에서 요청된 데이터를 검색하고, 검색 결과를 포함하는 출력 메시지를 생성하여 출력 메시지 큐에 추가한다. 행위 모듈은 규칙의 결론부를 실행하는데, 규칙 기반 시스템은 규칙의 결론부가 하나 이상의 행위 모듈 호출을 허용한다. 따라서, 제안한 구조는 개발자가 *if-then* 규칙을 사용해서 복잡하고 까다로운 어플리케이션을 쉽게 구축할 수 있게 한다. 한편으로, 행위 모듈은 기본 컨텍스트(primitive context)에 대한 컨텍스트를 해석한다. 작업 처리 모듈(task processing module)은 외부 시스템의 요청 작업을 처리하고 결과 메시지를 전송한다.

규칙 기반 시스템 모듈(Rule-based System Unit)

규칙 기반 시스템 모듈은 고수준의 컨텍스트 처리를 수행하고, 상황 인식 서비스를 기술하는 *if-then* 규칙을 포함한다. 고수준 컨텍스트 추출은 규칙 기반 시스템에 인코딩될 수 있는 *if-then* 규칙으로 표현될 수 있다. 다음은 현재 개발하고 있는 노인 간병인 데모 시스템 규칙의 예이다.

IF the master's present location is not of one of rest spots and has been staying at the location during more than the specified time, THEN mater is in the state of 'long-resting-at-improper-location'

규칙 기반 시스템은 상황 인식 제공, 상황 인식에 기반한 자동 실행과 상황 인식 기반의 정보 증강과 같은 상황 인식 서비스를 구축하는데 사용될 수 있다. 상황 인식 제공(context-aware presentation)은 규칙의 조건부에 고려하는 컨텍스트를 기술하고, 결론부에 호출될 행위 모듈의 이름을 기술하는 *if-then* 규칙을 생성함으로 가능하다. 행위 모듈은 현재의 상황에 기반하여 블랙보드에서 관련 있는 정보를 수집하여 사용자에게 제공한다. 개발자는 행위 모듈을 처리 모듈로 적재하고, *if-then* 규칙을 규칙 기반 시스템에 등록한다. 그런 다음 행위 모듈과 내부 모듈 호출 메시지 사이의 관련 정보를 통신 모듈의 옐로우 페이지에 등록하여 새로운 상황 인식 제공 서비스를 초기화한다. 상황 인식 서비스의 결론부는 작업을 자동으로 실행하는 행위 모듈을 호출한다. 상황 인식에 기반한 정보 증강(context-aware augmentation) 서비스에서, 행위 모듈은 관심있는 블랙보드 컨텍스트 객체에 관련 정보를 추가하고, 필요하다면 증강된 데이터를 나타내는 새로운 객체를 생성한다. 규칙 기반 시스템 모듈이 생성하는 내부 모듈 호출 메시지에는 호출되는 행위 모듈 이름과 조건부에서 참조하는 관련 컨텍스트 정보가 포함된다.

블랙보드(Blackboard)

컨텍스트 상태 변수, 규칙 통지 플래그, 중간 결과를 저장하는 임시 변수, 관련된 데이터의 데이터베이스를 저장하는 블랙보드는 통신 모듈, 처리 모듈, 규칙 기반 시스템 모듈에 의해 접근되고 사용된다. 블랙보드의 일부는 규칙 기반 시스템 모듈의 작업 메모리로 사용된다. SoC 기반의 하드웨어를 구현할 때, 작업 메모리는 SoC 하드웨어 구성요소와 고정 배선(hardwired)된다. 데이터베이스를 포함한 블랙보드의 모든 요소는 유일한 식별자를 가지며, 처리 모듈에 의해 객체로 간주된다. 또한, 데이터베이스에 레코드를 삽입,삭제,갱신,질의를 처리하는 처리 모듈이 있다.

4. SoC기반 구현

그림 2는 제안한 SoC 기반의 상황 인식 시스템 구조로서, 통신 모듈, 처리 모듈, 블랙보드를 포함하는 SoC 마이크로프로세서 부분과 규칙 기반 시스템 모듈을 구현한 하드웨어로 구성된다. 규칙 기반 시스템 하드웨어는 병렬로 모든 규칙의 조건부에 대한 매칭 연산을 수행한다. Rule Base CAM(content- addressible memory)은 모든 규칙의 조건부를 인코딩한 것으로 하나의 CAM 배열이 하나의 규칙에 할당되고, 동치 테스트(equivalence test)에 기반한 매칭 연산은 모든 규칙에 대해 동시에 이루어진다. 상충 해소 모듈(Conflict Resolution module)은 만족되는 규칙들 중에서 실행될 규칙를 선택한다. 현재 우선순위와 LRU(least recently used)에 기반한 상충 해소 전략을 지원한다. 상충 해소 모듈은 선택된 규칙의 결론부에 해당하는 행위 모듈 호출 메시지를 생성하거나 작업 메모리에 대해 갱신 연산을 수행한다. 빠른 병렬 매칭을 하기 위해, Rule Base CAM에서 수행되는 조건들은 모두 동치 테스트이다. 일반적인 규칙은 동치 테스트 뿐만 아니라 $>,\geq,\neq,<,\leq$와 같은 연산자를 포함한 비교 테스트도 할 수도 있다. 제안하는 구조는 비교 테스트를 전처리하기 위해 크로스바 네트워크(Crossbar Network module)를 사용한다. 이 네트워크는 비교 테스트를 수행할 수 있는 여러 개의 PPM(preprocessing modules)을 포함한다. Switch Input module은 PPM이 규칙의 비교 연산을 분산하여 수행하고 결과를 SoC 작업 메모리에 저장할 수 있도록 크로스바 네트워크를 제어한다. 모든 비동치 조건 테스트가 전처리되면, Rule Base CAM 모듈이 활성화된다. SoC 개발 설계 및 시뮬레이션 도구인 SystemC 소프트웨어를 사용하여 규칙 기반의 SoC 구조를 설계하고 검증하였다. SoC 마이크로프로세서는 통신 모듈, 처리 모듈과 블랙보드를 구현한다.

5. 결 론

SoC 기반 상황 인식 시스템 구조는 이동 디바이스나 지능형 이동 로봇에 내장된 상황 인식 어플리케이션을 구현하기 위해 설계되었다. 주거 환경에서 노인을 보조하기 위해 가공된 약 50여개의 규칙으로 이루어진 규칙 베이스에 대한 규칙 기반 시스템 하드웨어는 SystemC 시뮬레이션 개발 환경에서 성공적으로

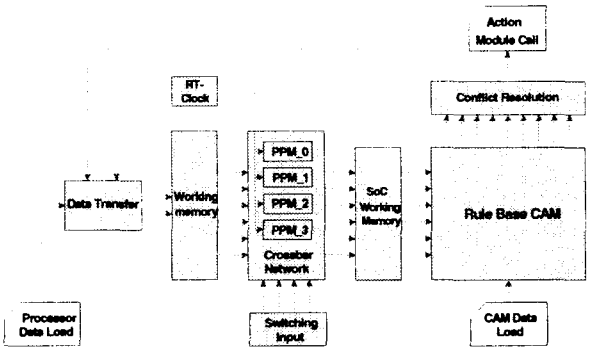

그림 2. 규칙 기반 시스템 모듈을 위한 SoC 구조

테스트되었다. 제안한 구조는 개발자가 센서나 컨텍스트를 쉽게 추가하거나 대체할 수 있고, 새로운 상황 인식 서비스를 구현할 수 있는 센서 추상화를 지원한다. 또한 컨텍스트 변경에 대한 통지 메커니즘은 컨텍스트 값에 대한 지속적인 감시를 피할 수 있게 하여 시스템 구현을 간단하게 하고, 메시지 지향(message-driven)적인 처리 모듈 호출은 모듈식 시스템 개발을 지원한다. 제안한 구조는 *if-then* 규칙을 이용해 행위와 컨텍스트를 연관시킬 수 있기 때문에 개발자가 쉽게 상황 인식 서비스를 구축할 수 있고, SoC와 같은 프로세서에 적합하게 개발되었기 때문에 이동 디바이스나 이동 로봇에 적용될 수 있다.

참고 문헌

[1] A. K. Dey, Providing Architectural Support for Building Context-Aware Applications, Ph.D. dissertation, Georgia Institute of Technology, 2000.

[2] P. J. Brown, The stick-e document: a framework for creating context-aware applications, Electric Publishing, Vol. 9, No. 1, pp. 1-14, 1996.

[3] D. Caswell, D. P. Debaty, Creating Web representations for places, Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing, pp. 114-126, 2000.

[4] B. N. Schilit, System architecture for context-aware mobile computing, Ph. D. dissertation, Columbia University, New York, 1995.

[5] J. Pascoe, Adding generic contextual capabilities to wearable computers, Proceedings of the 2nd IEEE International Symposium on Wearable Computers, pp. 92-99, 1998.

[6] A. Schmidt, K. A. Aidoo, et al, Advanced Interaction in Context, Proceeding of HUC'99, pp. 89-101, 1999.

[7] A. Dey, J. Mankoff, G. Abowd, S. Carter, Distributed mediation of ambiguous context in aware environments, Proceeding of the 15th annual ACM symposium on User interface software and technology, 2002.