Colored Petri-Net을 이용한 상황인식 서비스의 모델링과 검증 방법

(Modeling and Verification Methodology for Context-awareness Service using Colored Petri-Net)

한 승 욱 † 윤 희 용 ††
(Seungwok Han) (Heeyong Youn)

요 약 상황인식 서비스는 유비쿼터스 환경의 핵심으로 동적 환경에서 지능적이고 민감하게 동작하기 위해 상황과 서비스간의 연관관계를 명확하게 분류하기 위한 방법론이 요구된다. 기존의 모델링 방법들은 시나리오 중심의 상황정보 수집, 관리 그리고 표현에 초점이 맞추어 있다. 따라서 설계할 때 다양한 실제 환경에 적용하여 상황인식 서비스 모델의 적합성을 검증하기는 힘들다. 본 논문에서는 colored Petri-Net을 응용한 모델링 방법을 제시한다. 모델은 상황 정보의 변화와 흐름의 표현과 평가에 초점을 두고 시간을 추가하여 수행시간제약에 대한 표현 및 평가가 가능하도록 하였다. 또한 상황 모델링 툴킷과 에이전트 기반의 상황인식 엔진 그리고 상황정보 시뮬레이터를 활용하여 에이전트 기반의 상황인식 서비스를 개발하였다. 본 논문에서 제안한 모델링 방법은 Usilvercare에 적용하여 실용성을 검증하였다.

키워드 : 에이전트 플랫폼, 상황인식, 상황 모델링 툴킷, 모델링, 페트리 넷

Abstract Context-awareness is one of the key features of ubiquitous paradigm. A methodology that is specifying the relationships between the contexts and services needs to be developed to intelligently and sensitively deal with dynamic environment. The existing models on context-aware modeling are difficult to verify the correctness of models with respect to timeliness. In this paper we propose an approach which includes timing constraint in the relations of the context model, and verify its effectiveness using colored Petri-Net. Moreover, a context-modeling toolkit including context-awareness engine and simulator is developed to support agent-based context-aware service. The effectiveness of the proposed methodology is demonstrated using an example of Usilvercare.

Key words : Agent platform, Context-aware, Context-modeling toolkit, Modeling, Petri-Nets

1. 서 론

지금까지의 분산 컴퓨팅 환경은 네트워크를 기반으로 양질의 다양한 정보를 획득하는 것을 목표로 하고 있으나, 유비쿼터스 환경에서는 이와 더불어 다양한 환경에서의 지능적인 서비스 제공을 중요시 하고 있다. 이러한 서비스의 구현을 위해서는 먼저 사용자의 필요와 요구사항을 파악해야 하며 사용자 및 서비스를 둘러싸고 있는 상황을 정확하게 인식해야 한다. 상황인식 기반의 지능적인 서비스는 단순히 사용자의 상황을 인식해서 서비스를 제공해 주는 것이 아니라, 서비스 제공시간 지연이나 부적절한 상황정보 활용으로 인한 사용자가 불편함을 느끼지 않도록 해야 한다. 이러한 요구사항을 만족하기 위해 상황정보의 변화와 흐름을 명확하게 표현하기 위한 모델이 요구된다. 모델은 상황정보와 알고리즘 간의 관계를 명시하고 응용 또는 서비스 개발에 적용

· 이 논문은 본 연구는 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 지식경제부의 유비쿼터스컴퓨팅 및 네트워크원천기반기술개발사업의 08B3-S1-10M 과제, 2009년도 중소기업청의 산학 공동기술개발지원사업의 지원으로 연구되었음.

† 학생회원 : 성균관대학교 정보통신공학부
lovedonny@skku.edu
†† 종신회원 : 성균관대학교 정보통신공학부 교수
youn@ece.skku.ac.kr
(Corresponding author임)

논문접수 : 2008년 5월 29일
심사완료 : 2009년 2월 23일

정보과학회논문지 : 소프트웨어 및 응용 제36권 제4호(2009.4)

가능해야 하며 실제로 적용하기 전에 상황정보를 시뮬레이션과 실시간 검증이 가능해야 한다.

기존의 모델은 특정 상황이나 응용 서비스의 분류에 따라 상황정보를 다루는 것에 초점이 맞추어져 왔으나, 최근 다양한 응용에서 활용하기 위한 일반적인 상황모델에 대한 많은 연구가 진행되고 있다[1]. 상황인식과 상황인식 모델링을 위해 상황정보의 습득과 관리, 그리고 표현을 명확히 하기 위한 다양한 방법들이 제시 되고 있으며 Karen[2]은 포괄적인 상황인식 시스템의 개발과 모델의 평가를 단위테스트와 상황정보를 가지고 테스트가 가능한 모델을 제시하고 있다.

그러나, 현재까지의 연구는 설계할 때 다양한 실제 상황인식 환경에 적용하여 상황인식 서비스의 적합성을 검증하기는 힘들다. 최근 상황인식 모델링을 위해 시스템의 동시적 또는 병행적 프로세스를 표현하기 위한 모델링 방법인 Petri-Net(PN)을 상황인식 모델에 적용한 연구가 최근 활발히 진행되고 있다. Lee[3]는 상황인식을 위해 Timed PN을 사용하여 상황정보의 유효시간을 표현하였으며 Kwon[4]은 Colored Petri-Net(CPN) 모델을 확장하고 소스코드를 생성할 수 있도록 하였다. 또한 Patrick[5]은 Synchronized Petri-Net을 사용하여 상황과 응용의 행동을 표현 하여 가용성 평가에 유용하게 적용할 수 있도록 하였다. 그러나 위와 같은 방법으로는 상황정보 추론을 위한 상황정보와 알고리즘간의 관계를 표현하기 힘들다. 이에 더하여 상황정보를 실시간으로 모델에 적용하여 검증하거나 상황정보와 알고리즘에 시간까지 포함하지 고려하지 않고 있기 때문에 모델에 서비스에 대한 필수 전제조건이 되는 제한 시간에 따른 차등화 서비스를 적용하기 힘들다.

본 논문에서는 CPN을 응용한 모델링 방법을 제안하고 있다. 모델링은 상황인식 응용의 요구사항을 조사하여 정의하는 디자이너를 위해 상황정보의 변화와 흐름을 표현하며, 상황 정보의 변화와 흐름에 초점을 두고 모델에 시간을 추가함으로써 수행시간에 대한 제약시간을 표현하도록 하였다. 상황인식 모델링 툴은 기 연구개발한 에이전트 플랫폼[6]을 활용하여 에이전트 기반으로 구현하였으며 상황인식 서비스 모델링 할 때 에이전트를 통하여 실시간으로 수집되는 상황정보와 에이전트 시뮬레이터를 사용하여 서비스를 디자인 할 때 모델에 대한 실시간 검증이 가능하도록 하였다. 본 논문에서 제안한 모델링 방법은 USilvercare에 적용하여 실용성을 검증하였다.

본 논문의 구성은 다음과 같다. 2장에서는 상황인식 모델과 에이전트 그리고 PN에 대하여 소개하고 장단점을 살펴보았다. 3장에서는 본 논문에서 제안하는 상황정보 서비스 모델과 제어구조 그리고 실시간 검증에 대하여 서술하고 상황인식 서비스를 예를 들어 모델링을 보여준다. 4장에서는 에이전트 기반의 상황인식 플랫폼과 모델링 툴킷의 실행화면을 보여준다. 그리고 5장에서는 논문의 결론 및 향후 과제에 대해 기술한다.

2. 관련연구

유비쿼터스 환경의 상황인식 서비스에 대한 중요한 요구사항은 상황 모델링과 모델을 설계할 때 실제 상황인식 환경에 적용하여 상황인식 서비스의 적합성을 검증하는 것이다. 이러한 요구사항을 만족하기 위해 모델링 방법론과 자동화된 서비스를 위하여 에이전트 기반의 상황인식 엔진, 모델링 툴킷에 관한 연구가 선행되었다.

2.1 상황인식 모델

상황인식 서비스는 인식, 해석단계에서 사용되는 상황정보의 표현과 교환을 위한 모델이 요구되며 다양한 모델링 방법이 연구되어 왔다. 가장 단순한 데이터 구조를 가지고 있는 Key-value 모델, 속성과 내용을 가지고 있는 마크업 태그로 구성된 계층적 데이터 구조를 가지고 있는 마크업 스키마 모델, UML 다이어그램과 같은 그래픽 요소를 사용하여 모델링을 하는 입장에서 편리하게 사용 가능한 그래픽 모델, 은닉성과 재사용성과 같은 객체지향의 특징을 가지고 동적 상황정보로 인하여 발생하는 문제를 최소화 하기 위한 객체지향 모델, 추론엔진에서 사용하는 룰을 기반으로 하는 로직기반 모델, 상황을 정형화 하고 개념의 형식이나 사용사항의 제약사항을 명시적으로 정의하는 온톨로지기반 모델 등이 연구되어 왔으며 온톨로지 모델은 단순성, 유연성, 확장성, 일반성, 표현력이 뛰어나 다양하게 쓰이고 있다[1].

Karen[2]은 다양한 상황인식 응용개발을 위해 단위테스트와 상황정보를 가지고 테스트가 가능한 그래픽 모델을 기반으로 하는 모델링 방법을 제시하였다. 제시한 모델링 방법은 상황인식 서비스를 설계할 때 직관적으로 문맥상의 분류와 특징에 따라 센싱, 고정, 프로필, 추론, 일시적 그리고 모호한 사실 형식을 정의하고 사람과 통신 채널과 같은 객체를 표현하기 위한 사각형과 요청하는 디바이스와 사용허가와 같이 사실 형식을 다루는 역할을 위해 원을 정의하였다. 그리고 기호와 화살표, 그리고 눈금선을 사용하여 연관관계를 표현하도록 함으로써 직관적으로 모델링을 할 수 있도록 하였다. 그러나 위와 같은 상황인식 방법과 모델은 시나리오 기반의 상황정보의 습득과 관리, 그리고 표현에 초점을 두고 있어 상황정보의 유효시간을 표현하거나 상황모델을 설계할 때 평가하기 힘들다.

2.2 Petri-Net과 모델링

PN은 시스템의 동시적 또는 병행적 프로세스들을 표현하는 모델링 도구로 개발되어 일반적인 시스템의 모

델링과 분석을 포함하여 도식적, 수학적 모델이 가능한 도구로 해석적 분석방법이 이론적으로 잘 정립되어 있으며 다음과 같은 Hack의 기본 정의[7]를 따른다.

정의 1.

$C = \{P,T,F,B\}$, PN의 구조

$P = \{P_1,P_2,\cdots,P_n\}$, $P \neq \emptyset$, place의 유한집합

$T = \{t_1,t_2,\cdots,t_n\}$, $T \neq \emptyset$, $P \cap T = \emptyset$, transition의 유한집합

$F : P \times T \rightarrow N$, (단, N은 음수를 제외한 정수 집합), 정방향 영향함수

$B : P \times T \rightarrow N$, 역방향 영향함수

프로세스는 조건(condition), 사건(event) 및 그들간의 관계를 서술하는 규칙(rule)로 구성된다고 가정할 수 있으며 PN의 그래프에서 조건과 사건을 place(원)와 transition(선분)으로 나타낸다.

조건을 만족하는 것은 place에 토큰을 위치시킴으로써 표현하며 place와 transition간의 연결은 directed arc로 표시한다. transition은 자신에게로 입력되는 모든 place가 토큰을 보유하고 있어야 점화(fire)될 수 있다. transition이 점화되면 자신의 각 입력 place로부터 토큰을 하나씩 제거하고 각 출력 place에 토큰을 하나씩 첨가한다.

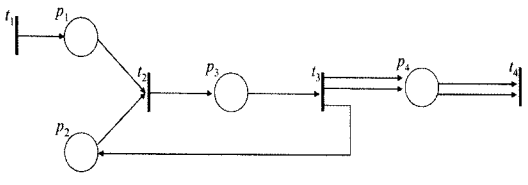
그림 1은 PN의 한 예를 나타낸 것으로 $P = \{p_1, p_2, p_3, p_4\}$, $T = \{t_1, t_2, t_3, t_4\}$인 경우이다.

PN은 다양하게 연구개발 되어 왔으며 CPN은 토큰에 복합정보를 표현할 수 있고, Timed PN은 시간을 고려할 수 있는 등 다양하게 발전되었다[8,9].

PN은 최근 상황인식 모델링에서 수학적 해석적 분석을 위해 다양한 목적으로 적용되어 왔다. Lee[3]는 상황인식을 위해 상황을 이벤트의 조합으로 구성하여 발생하는 이벤트간의 시간적 제한을 기술함으로써 일정시간 동안 어떤 상태를 유지할 때 특정 상황으로 판단할 수

$F(t_1) = 0$ $\quad B(t_1) = \{p_1\}$,
$F(t_2) = \{p_1, p_2\}$, $\quad B(t_2) = \{p_3\}$,
$F(t_3) = \{p_3\}$, $\quad B(t_3) = \{p_2, p_4; p_4\}$,
$F(t_4) = \{p_4, p_4\}$ $\quad B(t_4) = 0$

(a) PN 구조 표현



(b) PN 그래프 표현

그림 1 PN의 구조와 그래프 표현

있도록 하였다. Timed PN을 사용하여 상황정보의 유효시간을 표현하고 복합 이벤트가 발생했을 경우에도 문맥상의 상황을 잘 표현할 수 있도록 하였다. Kwon[4]은 사용자의 생활을 한정된 패턴으로 정의하고 Colored Petri-Net(CPN)[8]모델을 확장하여 상황변화에 대하여 동일시간에 적용되는 독립된 다른 패턴을 transition에 적용함으로써 효과적으로 상황정보를 모델링 할 수 있도록 했다. 이에 더하여 모델을 통하여 소스코드를 생성하여 상황인식 응용에 사용할 수 있도록 하였다. 또한 Patrick[5]은 Synchronized Petri-Net을 사용하여 상황과 응용의 행동을 표현하여 가용성 평가에 유용하게 적용할 수 있도록 하였다. 그러나 위와 같은 모델링 방법들은 상황을 효과적으로 기술하고 상황정보를 정확하게 다룰 수 있으나 모델 자체에 상황인식 환경에 사용되는 복잡한 룰이나 알고리즘을 포함하고 있지 않아 모델과 실제응용에서 사용하는 알고리즘을 평가하기에는 어려운 점이 많다.

2.3 에이전트 플랫폼과 상황인식 에이전트

에이전트란 시스템의 사용자를 대신해서 사용자가 원하는 작업을 자동적으로 수행해 주는 소프트웨어를 지칭한다[10]. 에이전트는 자율성, 지능, 이동성, 사교성 등의 특징을 가지고 있으며 크게 멀티 에이전트와 모바일 에이전트로 구분된다. 멀티 에이전트는 하나의 에이전트로 해결하지 못하는 복잡한 작업을 처리하기 위하여 다양한 작업을 할 수 있도록 개발된 다양한 에이전트간의 협업을 통해 작업을 수행하도록 하는 구조로 사람이 상황인식을 하는 것과 같이 에이전트가 온톨로지의 공유와 상황을 인식하고 이미 알고 있는 것을 토대로 추론함으로써 현재 상황에 적합한 행동을 할 수 있는 상황인식 서비스를 가능하게 할 수 있다[11].

에이전트 기반의 상황인식 시스템은 에이전트가 상황을 수집하고 에이전트간에 지식을 공유함으로써 상황인식을 효율적으로 할 수 있다는 장점을 가지고 있다. CoBra(Context Broker Architecture)[12]는 하나의 시스템을 다중 도메인으로 구성하여 도메인마다 각각의 온톨로지 기반의 모델을 포함하고 상황전달 기능을 하는 브로커를 둠으로써 상황인식 에이전트가 공통된 상황모델을 공유하여 상황인식 서비스를 제공하도록 되어 있다. ACAI(Agent-based Context-Aware Infrastructure for spontaneous applications)[10]는 상황인식 서비스를 위해 상황관리, 조정, 온톨로지, 추론, 시스템 지식기반, 상황전달과 같은 6가지 에이전트로 구분하여 개발하였으며 두단계로 표현된 온톨로지 기반의 모델을 사용함으로써 온톨로지를 공유하고 에이전트간의 협업을 통해 상황인식 서비스를 제공할 수 있도록 하였다. 에이전트 기반의 상황인식은 에이전트간의 협업을 통하여 상

황을 관리하고 추론하여 전달하는 기능을 포함하고 있으나, 실제 모델을 설계할 때 검증하기 어렵다. 따라서 본 논문에서는 CPN을 확장하여 설계할 때 검증을 할 수 있는 상황인식 에이전트와 모델링 툴킷을 제안한다.

3. 제안하는 모델링 방법

상황인식 서비스 모델링 방법은 상황인식 컴퓨팅의 가장 중요한 특징으로 상황정보의 종류와 흐름, 변화를 표현하기 위한 방법이다. 상황인식 컴퓨팅은 사람이 상황을 인지하고, 생각해서 판단하는 것과 같은 절차를 거친다. 사람이 상황을 인지하고 판단하는 절차와 같이 사용자에게 상황인식 서비스를 제공하기 위해서는 컴퓨터가 서비스의 종류에 따라 요구되는 상황정보를 구분하고 수집하여 상황에 맞는 형태로 재 가공한 뒤, 응용에 전달해 응용이 적절하게 실행할 수 있도록 해야 한다. 지금까지의 연구는 대부분 상황 데이터의 정의와 표현 방법 및 응용까지 제어를 하여 사용자에게 상황인식 서비스를 제공하는데 초점이 맞추어져 왔다. 또한 미들웨어를 사용하는 경우에도 서비스의 가용성 평가를 위해서 상황인식 서비스를 모두 개발한 후에 진행할 수 있기 때문에 응용과 서비스 설계에서부터 요구되는 상황 데이터와 상황정보의 흐름에 대하여 충분히 검증하기가 힘들었다.

본 논문에서는 CPN을 응용한 상황인식 모델링 툴을 사용하여 상황인식 서비스 모델링 할 때 실시간으로 수집되는 상황정보와 시뮬레이터를 사용하여 디자인 시간에 모델에 대한 실시간 검증이 가능하도록 하였으며, 에이전트 플랫폼을 기반으로 하여 상황인식 엔진과 상황인식 모델링 툴을 개발 하였다.

3.1 상황 정의

상황은 '상황의 요소를 특성을 나타내기 위해 사용될 수 있는 모든 정보'로 정의할 수 있다[13]. 이러한 상황정보를 이용한 서비스는 대상이 되는 서비스 주체에 따라 그 정보의 가치가 변화한다. 에이전트 기반의 상황인식 서비스는 에이전트의 자율성을 만족시키기 위하여 상황정보에 따라 특정 응용의 행동(실행)을 지정해 주는 것이 아니라 상황 정보를 전달함으로써 에이전트가 지능적 판단을 통하여 상황에 최적화된 서비스를 제공할 수 있도록 하였다.

제안된 방법은 사람이 상황을 인지하고 경험 및 지식 기반으로 추론하고 행동하는 것과 같이 상황을 이벤트로 표현하고, 이벤트의 내용에 따라 에이전트의 자율적, 지능적 행동을 최대화하기 위해 상황에 따른 응용의 제어가 아닌 상황정보 표현에 초점을 두었다. 상황정보 서비스 모델에 사용되는 상황정보는 다음과 같은 요소를 포함하고 있어야 한다.

$C = \{c_1, c_2, \cdots, c_n\}$	상황정보 집합
$OM = \{om_1, om_2, \cdots, om_n\}$	온톨로지 정의 유한집합
$CN = \{cn_1, cn_2, \cdots, cn_n\}$	상황정보 이름
$CV = Val(CN)$	상황정보 값
$CT = Time(CN)$	상황정보 초기 발생시간

하나의 상황정보는 상황정보의 이름과 각각의 이름에 대한 상황정보 값을 가지며, 이름과 상황정보의 값은 온톨로지에 정의되어 있어야 하며 다음과 같이 표현할 수 있다.

$$c = \{(cn, cv, ct) \mid \forall cn, \forall cv \in om\} \quad (1)$$

3.2 상황 모델

사용자에게 특정 서비스를 제공하기 위해서 사용되는 주변 환경의 상황정보는 서비스의 특징에 따라 한정 지을 수 있으며 다양하게 표현할 수 있다. 상황정보 서비스 모델은 상황정보를 이용하는 응용 서비스를 설계할 때 사용하는 모델로 서비스 주체의 특징에 따라 요구되는 상황정보의 종류와 상황정보의 흐름, 그리고 상황정보의 변화를 표현하기 위한 방법으로 그래픽이나 모델링 언어로 표현할 수 있다. 지금까지 워크플로우, UML, PN과 같은 다양한 모델링 방법들이 상황인식에 적용되어 왔으나, 상황정보의 흐름과 변화에 초점을 둔 것이 아니고 서비스를 실행하는 순서나 방법에 대해 초점을 두고 있기 때문에 상황정보의 변화와 흐름을 표현하기에는 제약이 많다.

본 논문에서는 CPN을 응용하여 상황정보의 변화와 흐름을 표현하며, 직접적인 응용에 대한 구체적인 행동은 표현하지 않고 상황 정보의 변화와 흐름에 초점을 두었다. 이는 응용에 따라 독립적으로 다양하게 구현하며 행동할 수 있는 이점을 제공한다. 서비스 제공자는 사용자 조사 및 도메인 연구를 통하여 상황인식 서비스를 모델링 할 때 상황정보와 제어흐름에 대하여 구체적으로 정의하고 표현한다. 이 때에 사용되는 모델은 서비스에서 요구되는 사건, 행동, 일어난 일을 표현할 수 있어야 하며 센서기기로부터 수집하는 하위정보로부터 상위정보를 추출하는 과정에서 사용하는 온톨로지와 인공지능 엔진에 사용하기 위한 룰과 계산식을 표현할 수 있어야 한다.

상황인식 정보의 개체, 속성과 동적인 행동을 표현하기 위해 상황인식 서비스 모델은 다음과 같이 표현이 가능하다.

- Color: 상황정보의 이름 및 값
- Place: 상황정보 제공자와 소비자(서비스, 응용)
- Transition: 상위 상황정보를 얻어내기 위한 추론엔진 룰 또는 알고리즘

- Arc: if then룰을 사용하여 place에서 transition으로 점화(fire)하기 위한 조건 표현

정의 2. 상황정보 모델

$SM = <\Sigma, P, T, A, N, C, E, \tau>$

Σ: 공백이 아닌 상황정보의 유한집합, (color set)

P: 상황정보 제공자 또는 소비자의 유한집합, (place)

T: 상황에 특성화된 룰 또는 알고리즘의 유한집합, (transition, $P \cap T = \varnothing$)

A: 룰과 상황정보와의 연관관계 (arc, $P \cap T = P \cap A = T \cap A = \varnothing$)

N: 노드 함수, $N: A \rightarrow \{P \times T\} \cup \{T \times P\}$

C: 상황정보(color), $C: P \rightarrow \Sigma$

E: 상황정보의 범위와 조건의 유한집합, (arc 표현)

τ: 프로세스 수행에 걸린 시간 (transition, τ는 T에 대응)

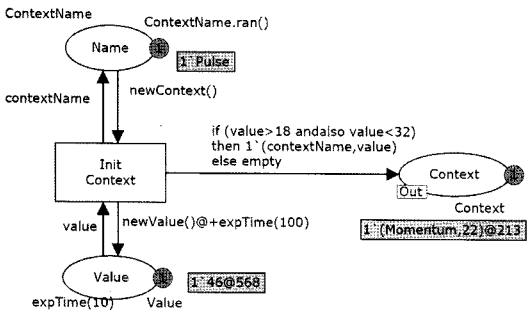


그림 2 상황정보 생성을 위한 모델과 시뮬레이션

정의 2를 사용해 신체온도, 운동량, 맥박중 하나의 상황이름을 가지고 있는 상황정보의 생성은 그림 2와 같이 표현할 수 있다. type place는 상황 이름을 신체온도, 운동량, 맥박 중 하나의 형식으로 임의로 선택하며 그 상황에 대한 값은 0부터 60사이의 값으로 value place에서 랜덤하게 생성된다. 이때, 상황정보의 발생은 exp-Time(100) 함수를 사용하여 지수 시간으로 발생시키며 이는 평균값을 100으로 하는 지수분포다. arc는 상황값(value)이 18에서 32 사이일 때만 context place로 상황이 전달된다.

그림 2에서 상황정보는 운동량(Momentum)의 상황형식을 가지며, 22의 값을 가지고 있고 213만큼의 시간이 지난 뒤에 발생하였음을 표현한다.

다음 그림 3은 상황정보 생성을 위한 모델에 사용된 Standard ML(SML)[14]을 나타낸다. 상황정보는 (1)에서 정의한 것과 같이 상황정보를 시간을 포함하는 상황이름과 값으로 정의하며 그림 3에서 colset Context로 정의한다.

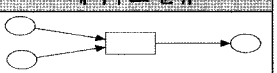
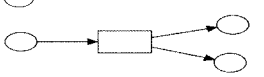
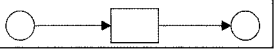
```
colset ContextType = with A | B | C;
colset VALUE = int with 0..60 timed;
colset Context = product ContextType*VALUE timed;
var contextType:ContextType;
var value:VALUE;
var context:Context;
fun newContext() = ContextType.ran();
fun newValue() = VALUE.ran();
fun expTime(mean:int) =
  let
    val realMean = Real.fromInt mean
    val rv = exponential((1.0/realMean))
  in
    floor(rv+0.5)
  end;
```

그림 3 상황정보 생성을 위한 예제

3.3 서비스 모델의 제어구조

PN으로 표현된 서비스 모델링 데이터를 분할 정복(divide and conquer)방법에 따라 최소화 하여 상황정보의 흐름을 표현하게 되면 다음과 표 1과 같은 제어 구조로 구분 할 수 있다. 제어구조 분류는 상황에 대한 종속성과 모델에 대한 분석을 쉽게 할 수 있도록 만든다.

표 1 상황인식 제어구조 분류

구분	제어구조 분류
(A) 복합상황	
(B) 복합전달	
(C) 단순상황	

상황정보에 따른 제어구조는 다음과 같은 특징을 가지고 있다.

(A) 복합상황: 여러 가지 상황의 조합을 통하여 하나의 상황을 생성할 때

(B) 복합전달: 상황이 여러 가지 상황정보에 사용될 때

(C) 단순상황: 하나의 상황이 다른 상황으로 연결될 때

이와 같은 제어구조는 하나하나의 상황정보에 초점을 두어 표현할 수 있으며, 모든 모델을 완성하기 전에도 transition에 사용된 추론엔진이나 알고리즘을 테스트해 볼 수 있다. Usilvercare에서 (A)는 복합상황인식 데이터를 통하여 위험상태인지, 아닌지를 구분할 때 사용되며, (B)는 위치와 같이 상황 정보가 다양한 상황정보 생성에 사용하거나 다양한 응용에 전달 하는 것을 의미한다. (C)는 심장마비와 같은 긴급상황에 적용된다. 위 3.1의 정의에서 place는 상황정보 제공자와 소비자를 의미하기 때문에 (C)에 있는 전방의 place는 (B)로 대치 가능하며, 후방의 place는 (A)로 대치 가능하다.

제안된 분류법을 사용하여 하나의 서비스에 대한 모델에서 각각의 상황정보에 대한 흐름을 표현하고 테스트해 볼 수 있지만, 이와 같은 방법으로는 Usilvercare와 같이 응급 상황에 대한 시간적 요구사항까지 만족하기는 힘들다. 예를 들면, 심장마비와 같은 상황이 발생하였을 경우 4분 이내에 응급처치를 하지 않으면 생존율이 급격하게 떨어지게 되는데 복잡한 상황인식 절차와 알고리즘을 수행하거나, 상황인식 방법에 따라 심장마비의 상황을 인식하는데 시간을 많이 소비하게 되면 상황인식 서비스가 필요 없어지게 된다. 따라서 실시간 검증에서는 전체 transition에 사용된 엔진이나 알고리즘이 실제 프로세스에 수행하는데 걸리는 시간을 측정하고 적용함으로써 유용성을 판단 할 수 있다.

제안하는 모델에서 룰과 알고리즘은 transition으로 표현할 수 있다. 사용된 룰과 알고리즘의 가용성은 기존의 서비스를 수정하지 않고, 기 설치되어 있는 하드웨어와 소프트웨어 인프라, 그리고 응용 서비스 도메인에 적용해 봄으로써 손쉽게 평가가 가능하다. transition t_1을 통하여 c_0 상황에서 c_1 상황으로 변환될 때에 $(c_0 \xrightarrow{t_1} c_1)$과 같이 표현할 수 있다. t_1에 대하여 실제 프로세스를 수행하는데 소요되는 제한 시간은 τ_i로 표현할 수 있다.

수행시간에 대한 오차 범위를 포함하는 프로세스를 수행 시간은 룰과 알고리즘을 적용했을 때 수행시간(lt_i)과 수행시간에 대한 최대값과 최소값의 차이(st_i)를 더하여 구함으로써 실제 시스템에 적용했을 때 발생하는 오차를 줄일 수 있으며 다음과 같이 표현할 수 있다.

$$\tau_i = lt_i + st_i \tag{2}$$

따라서 하나의 상황인식 서비스 모델이 포함하고 있는 모든 transition에 대한 수행시간을 더한 결과는 상황이 발생했을 때, 사용자에게 서비스 제공을 위해 요구되는 최소 시간보다 작아야 한다. 하나의 서비스 모델에 대한 transition의 수를 tn으로 하고, 서비스에서 요구되는 최대 제약 시간을 TL이라 했을 때 다음과 같은 공식이 성립해야 서비스 모델이 성공적이라는 것을 보여줄 수 있다.

$$TL \geq \sum_{i=1}^{tn} \tau_i \tag{3}$$

위에서 제시한 상황인식 서비스 모델의 제어구조와 공식(3)에 따라 실시간으로 수집되는 상황정보에 따라 상황인식 서비스 모델의 사용성을 평가할 수 있다. 최소 요구시간에 따라 상황을 인식하기 위한 방법과 대안을 실시간으로 변경 및 적용함으로써 상황인식 엔진에 사용된 룰 엔진과 알고리즘, 그리고 서비스 모델에 대하여 실시간 평가 및 시뮬레이션이 가능하다. 예를 들어 위치, 체온, 심전도등의 수치변화 수집 주기와 상황인식에 사용되는 알고리즘의 수행시간 오차범위 조율에 따라 상황인식 방법을 결정하는데 도움을 줄 수 있다. 또한, 심전도를 기반으로 하는 심근경색 및 심장마비에 대하여 해당 센서가 동작을 할 수 없는 경우, 운동량 센서와 위치센서를 조합하여 사용자의 상황을 판단하는데 사용할 수 있다. 이와 같은 접근 방법은 의료계에서 연구한 실버케어와 같은 헬스케어 서비스를 시간 제약조건 하에 서비스 설계와 평가에 적용 가능하며, 센서나 그 밖의 디바이스를 활용하여 수집된 상황정보의 오류 때문이 아닌 서비스 모델의 (여러 가지 상황정보를 가져와서 판단할 때 생기는 오류) 오류를 최소화 할 수 있다.

정의 2를 사용하여 상황인식 서비스 모델을 설계하면 다음과 같다. 사용자 정보 UPR은 흡연여부, 과거질병(고혈압), 과거질병(저혈압), 현재질병(고혈압), 과거 질병(저혈압), 체중, 나이와 같은 정보를 포함하고 있으며 다음과 같이 표현된다. UPR={SMO, PD_HBP, PD_LBP, CD_HBP, CD_LBP, WE, AGE} 상황정보는 UPR을 포함하여 체온(BT), 운동량(MOM), 응급지수(ER), 맥박(PU)와 사용자 정보와 운동량으로 판단한 부가정보($INFO$), 변화율 판단 알고리즘을 적용한 맥박 변화율(PV)가 존재한다. T는 부가정보판단 알고리즘(T_1)과, 변화율 판단 알고리즘(T_2)을 나타낸다.

그림 4는 그래프 기반의 맥박 변화율 측정을 위한 서비스 모델을 보여준다. 여기에서 TL는 실제 알고리즘을 수행하는데 사용되는 시간으로 τ의 합계가 1800이 되

$\Sigma = \{UPR, MOMT, ERR, INFOR, PUR, PVR\}$
$P = \{BT, MOM, ER, INFO, PU, PV\}$
$T = \{T_1, T_2\}$
$A = \{BTtoT_1, MOMtoT_1, T_1toINFO, ERtoT_2, INFOtoT_2, PUtoT_2, T_2toPV\}$
$N(a) = \{SOURCE, DEST\}$ a가 SOURCE to DEST 형태
$C(p) = \{P\}$ 만약 $p \in \{BT, MOM, ER, INFO, PU, PV\}$
$\tau(t) = \{30, 25\}$

$$E(a) = \begin{cases} \text{if } bt = true \text{ then 1`}bt \text{ else empty} & \text{if } a = BTtoT_1 \\ \text{if } mom > 0 \text{ andalso } mom < 15 \text{ then 1`}mom \text{ else empty} & \text{if } a = MOMtoT_1 \\ \text{if } info > 0 \text{ andalso } info < 1 \text{ then 1`}info \text{ else empty} & \text{if } a \in \{T_1toINFO, INFOtoT_2\} \\ \text{if } er > 0 \text{ andalso } er < 100 \text{ then 1`}er \text{ else empty} & \text{if } a = ERtoT_2 \\ \text{if } pu > 3 \text{ andalso } pu < 20 \text{ then 1`}pu \text{ else empty} & \text{if } a = PUtoT_2 \\ \text{if } pv > 0 \text{ andalso } pv < 1 \text{ then 1`}pv \text{ else empty} & \text{if } a = T_2toPV \end{cases}$$

그림 4 맥박 변화율 측정을 위한 서비스 모델

는데 이는 모델에 따라 수행했을 경우 맥박변화율 측정에 1.8초가 사용됨을 나타낸다. 이 LT는 실제 수행시간이므로 추론엔진을 사용할 경우에는 엔진에 포함되어 있는 룰의 수나 학습엔진에 사용되는 요소의 개수에 따라 변화하므로 수행시간(lt_i)과 수행시간에 대한 최대값과 최소값의 차이(st_i)에 주의하여야 한다.

4. 상황인식 모델링 툴킷 구현 및 평가

본 논문에 사용한 에이전트 플랫폼은 FIPA(Foundation for Intelligent Physical Agents)[15] 표준에 따라 C++ 기반으로 저자가 기 연구개발한 멀티에이전트 플랫폼[6]을 사용하였으며, 상황인식 에이전트는 기존의 추론엔진을 사용하기 위해 Java로 개발 하였다. 상황인식 에이전트는 도메인 별로 다르게 구현하여 도메인별 온톨로지를 포함하도록 하였다. 에이전트간의 상황정보 교환과 표현을 위해 XML 기반의 마크업 스키마 모델을 사용하였으며, 상황인식 엔진은 온톨로지 기반으로 상황정보간의 관계와 추론 룰을 저장하여 상황 정보 요청을 받아들여 상황정보 모델을 기반으로 상위레벨 상황정보를 도출하는 상황인식 서비스를 위한 핵심 기능을 담당한다.

그림 7은 에이전트 기반으로 구현되어 있는 모델링 툴킷으로 상황인식 서비스 모델에 CPN을 응용한 모델링을 표현하였다. 타원형태의 place에는 상황정보를 표현하였으며, 직사각형으로 표현된 transition에는 상황인식을 위한 룰, 그리고 화살표 arc는 룰과 상황정보와의 연관관계와 조건을 표현 하였다.

상황인식 모델링 툴킷은 에이전트로 구현이 되어 있어 모델링 툴킷을 사용하여 본 논문에서 제시한 방법으로 서비스에 대한 모델을 할 때에 실시간으로 상황인식에 사용되는 룰이나 알고리즘을 실제 상황인식 에이전트에 등록하고, 실행함으로써 표 1과 같이 각각의 상황정보 모델에 대하여 최종적으로 발생하는 상황정보에 대한 룰과 알고리즘의 수행시간을 측정함으로써 상황인식 서비스를 위한 제한시간에 대한 검증이 가능하다.

USilvercare에 적용한 모델은 9가지 모델로 룰을 사용하여 알고리즘을 표현하였으며, 비트 컴퓨터의 손목센서, 센서네트워크 미들웨어[16]와 에이전트 플랫폼을 연동하여 개발하였으며 소요시간에 따라 센싱주기와 모델을 수정하거나 룰을 수정하였다. 사용자의 상황에 따라 발생할 수 있는 다양한 조건은 손목센서 시뮬레이션 에이전트를 사용하였다. 단일 사용자 환경에서 각각의 모

표 1 USilvercare 상황정보 모델별 수행시간

상황 모델	룰 개수	상황 개수	소요 시간(초)
위치 이탈	2	7	1.1
응급 호출	1	1	0.5
대기온도 범위	2	12	2.5
심장마비	1	1	0.8
고열/저체온	16	3	2.3
실외 오래거주	10	3	2.1
거동 마비	3	1	0.8
맥박변화율	15	4	1.8
체온변화율	12	3	1.7

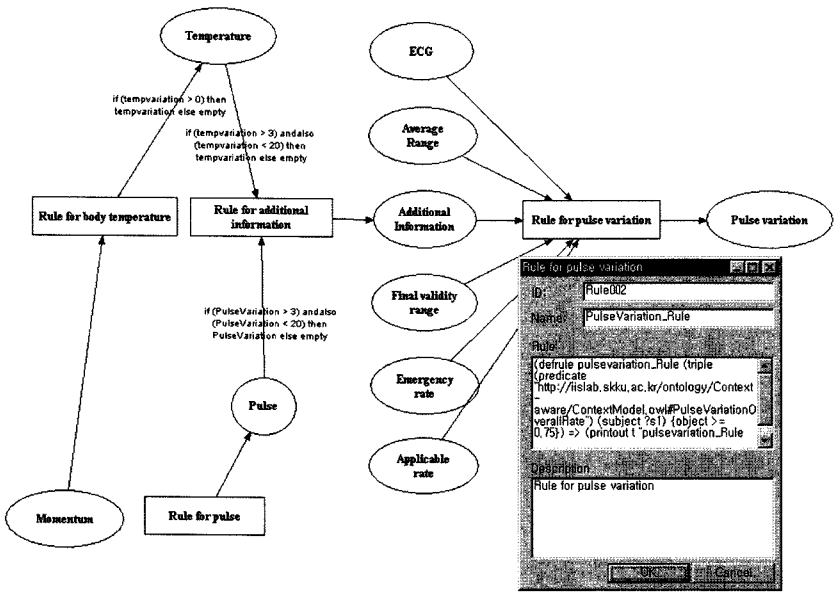


그림 7 상황인식 모델링 툴킷

델에 따라 소요시간은 1초에서 5초까지 차이가 있었으며, 상황정보 센싱주기와 상황정보의 수에 따라서 현격한 차이를 보였다.

5. 결 론

본 논문은 지능형 상황인식 응용을 개발하기 위한 상황인식 모델링 방법과 툴킷을 제안하였다. 모델링 방법은 구조적 모델링을 위해 CPN 모델을 확장해 상황정보와 상황인식을 위한 룰, 알고리즘을 표현할 수 있도록 하여 설계할 때 모델의 분석 및 평가에 활용할 수 있다. 상황 모델은 하드웨어와 소프트웨어의 제약사항과 필수 전제조건을 포함하는 시간 제약 서비스에서의 성능분석을 위해 시간을 고려한 상황인식 서비스에서 상황정보와 알고리즘간의 관계를 표현 하였으며, 실제 환경에서의 검증을 가능하게 함으로써 다양한 상황인식 응용 개발을 용이하게 하였다. 또한 멀티에이전트 기반으로 구현하여 기존의 동작하고 있는 환경에 영향을 주지 않고 상황인식 에이전트와 상황정보 제공, 소비자를 통해 센서와 RF-ID와 같은 상황정보를 수집하기 위한 장비가 없이 독립적으로 상황인식 엔진과 상황인식 서비스 개발을 가능하게 하였다. 향후 CPN모델을 응용하여 서비스의 공정성, 교착상태, 생존성등을 평가하고, 최적화된 상황인식 서비스의 설계와 에이전트의 속성을 최대한 활용할 수 있도록 지원하기 위한 지속적인 연구가 필요하다.

참 고 문 헌

[1] Anind K. Dey, "Understanding and Using Context," Personal and Ubiquitous Computing, Vol.5, No.1, pp. 4-7, 2001.

[2] Patrick, R., O. B., Dominique, V., et al., "Context-aware environments: from specification to implementation," Expert Systems, Vol.24, No.5, pp. 305-320, 2007.

[3] Strang, T. and Linnhoff-Popien, C., "A context modeling survey," In First International Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp, pp. 33-40, 2004.

[4] Henricksen, K. and J. Indulska, "Developing context-aware pervasive computing applications: Models and approach," Pervasive and Mobile Computing, Vol.2, No.1, pp. 37-64, 2006.

[5] Hack, M., "Decidability Questions for Petri Nets," Ph. H. dissertation, MIT, Massachusetts, Dec, 1975.

[6] Jensen, K., "Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use," EATGCS Monographs on Theoretical Computer Science. Springer. 1992.

[7] Ramchandani, C., "Analysis of asynchronous concurrent system by timed Petri nets," Cambridge, MA. MIT, Project MAC, TR.120, 1974.

[8] Brown, P. J., Davies, N., et al., "Towards a better understanding of context and context-awareness," Handheld and ubiquitous computing, Springer, No. 170, pp. 304-307. 1999.

[9] FIPA-Foundation for Intelligent Physical Agents, http://www.fipa.org, Park, A. H., Park, S. H., Youn, H. Y., "A Flexible and Scalable Agent Platform for Multi-agent Systems," Transactions on engineering, computing and technology, Vol.19. pp. 1-6, 2007.

[10] Byung Kwon, O., "Modeling and generating context-aware agent-based applications with amended colored Petri nets," Expert Systems with Applications, Vol.27, No.4, pp. 609-621, 2004.

[11] Lee, K. M., "Colored Timed Petri Nets-based Context Inference," Journal of the Research Institute for Computer and Information Communication, Vol.14, No.2, pp. 41-48, 2006.

[12] Paulson, L. C., "ML for the working programmer," Cambridge University Press, 2nd edition, 1996.

[13] Chen, H., "An intelligent broker architecture for context-aware systems," PhD. Dissertation, University of Maryland, 2003.

[14] Khedr, M. and A. Karmouch., "ACAI: agent-based context-aware infrastructure for spontaneous applications," Journal of Network and Computer Applications, Vol.28, No.1, pp. 19-44, 2005.

[15] Kim, Y. B., Kim, C., Lee, J. W., "A middleware platform based on multi-agents for u-Healthcare services with sensor networks," Agent and Multi-Agent Systems: Technologies and Applications, No.4953, pp. 683-692, 2008.

한 승 욱

2003년 강남대학교 전자계산학과(학사) 2005년 성균관대학교 정보통신공학부(석사). 2005년~현재 성균관대학교 정보통신공학부 박사과정. 관심분야는 시스템 소프트웨어, 분산처리, 미들웨어

윤 희 용

정보과학회논문지 : 소프트웨어 및 응용 제 36 권 제 3 호 참조