워크플로우 모델에서의 역할 의존성 분석

*원재강°, *김학성, *김광훈, *정관희
*경기대학교 일반대학원 전자계산학과 그룹웨어연구실
{jkwon, kwang, khchung}@kuic.kyonggi.ac.kr

Role Dependency Analysis in Workflow

*Jae-Kang Won°, *Hak-Seong Kim, *Kwang-Hoon Kim, *Kwan-Hee Chung
*Dept of Computer Science, Kyonggi University

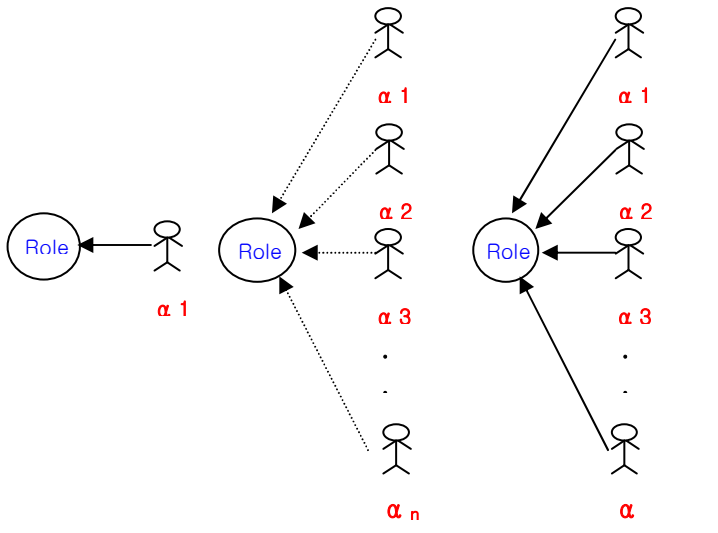
요 약

본 논문에서는 워크플로우 모델링 도구인 ICN(Information Control Net) 모델을 이용하여 워크플로우 역할 의존성 분석 메커니즘을 제안하였다. 즉, ICN 모델로 정의된 워크플로우의 각 액티비티(activity)들 간에 존재하는 역할 의존 관계를 표현하기 위하여 역할 의존 넷(Role Dependency Net)을 정의하였고, ICN 모델로부터 역할 의존 넷을 생성하는 알고리즘을 제안하였다. 본 논문에서 제시된 알고리즘을 이용하여 생성된 역할 기반의 워크플로우 모델은 any-cast 워크플로우와, multi-cast 워크플로우 작업환경을 제공함으로써 현재 대부분 조직에서의 작업 환경인 객체지향 작업 환경 및 분산 작업 환경에서 워크플로우 관리 시스템을 구축할 수 있다.

1. 서론

컴퓨터 기술과 전자통신 기술의 급진적인 발전은 전자적인 작업환경(Electronic Workplace)이라고 하는 새롭고도 매우 효율적인 상호 작용 지원 수단 및 방법을 탄생시켰다. 워크플로우 기술이란 바로 이러한 전자적 작업환경과 단위업무의 변화에 대처하고 보다 효율적인 조직의 운영을 위한 기술이며, 컴퓨터 및 통신 분야뿐만 아니라 사회학 분야나 언어학 분야, 경영학 분야 등의 다각적인 협력 관계를 통해서만 성공적으로 완성될 수 있는 매우 다중적인 분야라고 할 수 있다.[1,3]

워크플로우 기술은 워크플로우 모델 분야와 워크플로우의 각 업무들을 실행하고 그들 간의 업무 흐름을 제어하는 워크플로우 관리 시스템(Workflow Management System) 분야로 나누어질 수 있다. 특히 워크플로우 모델 분야는 조직 내에서 발생하는 워크플로우를 정의하고 분석하여 작성되어진 모든 정보 - 액티비티(activity), 행위자(actor), 역할(role), 데이터(data) –와 같은 객체와 객체들 사이의 관계를 이용하여 워크플로우 관리 시스템을 구현할 수 있다. 즉, 워크플로우 모델 부분이 워크플로우 프로시저를 설계하는 단계와 워크플로우 관리 시스템을 구현하는 단계에 많은 영향을 미치고 있다는 것을 의미한다. 그러므로 워크플로우 모델은 조직내의 특성과 발전된 기술들을 포함하여 워크플로우 관리 시스템이 새로운 기술이나 조직내의 작업환경에 효과적으로 적용될 수 있도록 작성되어야 한다. 그러나 현재와 같이 대규모이며 복잡해진 조직의 업무 프로세서를 정의, 분석하여 워크플로우 관리 시스템을 구현하기 위해서는 새로운 워크플로우 모델링 방법이 요구된다. 즉 이전의 액티비티 기반의 워크플로우 모델링에서는 그림 1 의 (a)에서와 같이 역할(role)에 대하여 단일 행위자(actor)만이 적용되는 uni-cast 한 워크플로우 모델을 적용하여 조직내의 워크플로우 프로시져를 정의하고 워크플로우 관리 시스템을 구현하는데 있어 문제가 없었으나 현재와 같이 사람의 개입을 최소화 하기 위한 워크플로우 시스템(그림 1 (b) : any-cast)이나 다중의 행위자(actor)들이 공동적으로 작업해야 하는 협업 작업과 같은 업무 프로세서를 처리하기 위한 워크플로우 시스템(그림 1 (c) : multi-cast)을 구축하기 위해서는 새로운 모델링 방법이 요구되고 있다.



[그림 1] 역할과 행위자간 관계에 따른 작업처리 방법

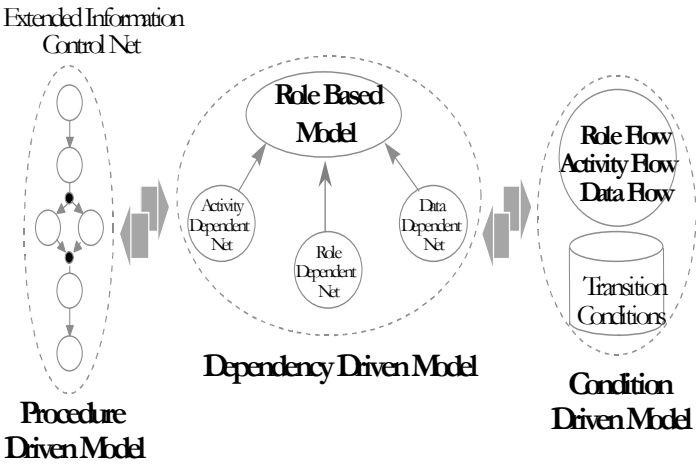
이러한 기능을 제공하기 위하여 본 논문에서는 기존의 액티비티를 기반으로 하는 워크플로우 모델의 개념에서 탈피하여 새로운 역할 기반의 워크플로우 모델을 제시하였으며, ICN 모델을 이용하여 역할 기반 워크플로우 모델을 분석하였다. 즉, 워크플로우 모델에서 액티비티들 간의 실행 순서에 대한 역할 흐름을 분석하여 각 액티비티들 간에 존재하는 역할 의존성 관계를 정의하기 위한 정형적 메커니즘을 제안하였으며, 액티비티들 간의 역할 의존성 관계를 정의하는 역할 의존 넷의 알고리즘을 제안하였다.

2 장에서는 역할 기반 워크플로우 모델에 대한 개념을 설명하고 3 장에서는 ICN 으로 작성된 워크플로우 모델에서 역할 의존성 분석 방법 및 알고리즘을 제안하였다. 끝으로 4 장에서는 본 논문의 결론 및 향후 연구과제에 대하여 기술한다.

2. 역할 기반 워크플로우 모델 개념 및 구조

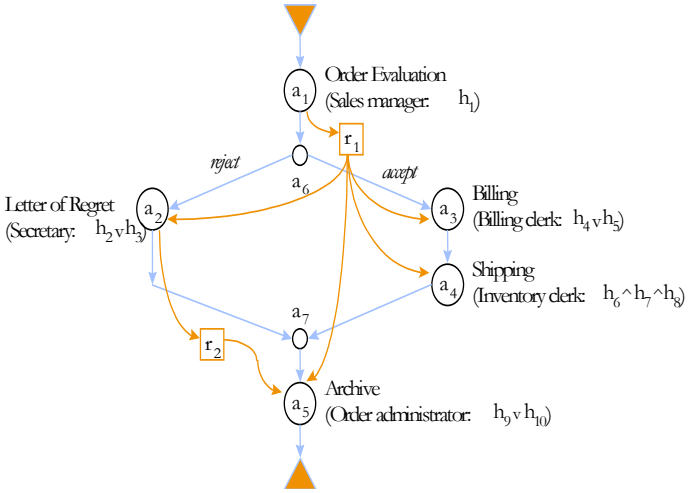
워크플로우 모델은 조직내의 워크플로우를 정의하는 객체들과 그들의 관계로 구성되어 있다. 즉, 워크플로우 모델을 이용하여 사용자는 업무 프로시저를 설계, 정의하고 개발자는 모델링으로부터 생성된 정보를 기본으로 워크플로우 관리 시스템을 개발한다. 이와 같이 워크플로우 관리 시스템의 구현에 많은 영향을 주는 워크플로우 모델은 개발자의 프로그램 양을 최소화 하여야 하고 다양한 응용 프로그램을 표현할 수 있어야 한다. 이러한 기능을 제공하기 위한 “Flow Path workflow system”, ”Action workflow system”, ”View Start workflow system” 과 같은 워크플로우 시스템이 구현되어 있지만, 현재와 같은 작업 환경을 지원하고 조직의 특징을 표현 하기에는 그 자체로는 적당치 않다[5].

역할 기반 워크플로우 모델은 그림 2 에서 보는 것과 같이 3 가지 모델로 구분되어질 수 있다. 이러한 3 가지 모델은 프로시저에 포함 되어있는 데이터와 제어의 흐름을 조절하기 위한 전이 상태를 다루는 워크플로우 관리 시스템으로부터 형성 된다.



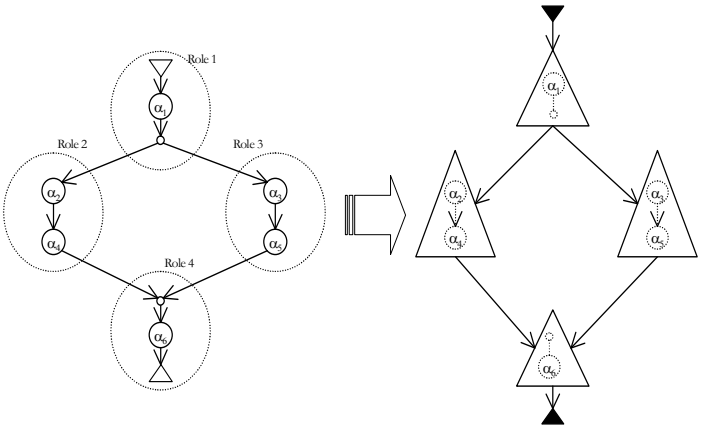
[그림 2] 역할을 기반으로 하는 모델

프로시저 운용 모델(Procedure Driven Model: PDM)은 워크플로우 관리 시스템의 모델링 부분에 의하여 조절 되어지며 ICN 모델에 의하여 표현된다. 의존 운용 모델(Dependency Driven Model: DDM)은 추상화 되어진 워크플로우 모델을 나타낸다. DDM 은 PDM 으로부터 상태 운용 모델(Condition Driven Model: CDM)을 생성하기 위한 중간 매개체의 역할을 한다. DDM 이라고 불리어지는 역할 기반 워크플로우 모델의 추상화 단계는 PDM 으로부터 자동적으로 발생되어질 수 있다. 결국 역할 기반 워크플로우 모델은 데이터베이스 상의 선행 데이터의 집합으로 표현되어진다. 워크플로우 프로시저를 구성하는 액터 의존 넷, 역할 의존 넷, 액티비티 의존 넷, 그리고 데이터 의존 넷은 각각의 액터, 역할, 액티비티, 그리고 데이터들 사이의 의존성 분석에 의하여 ICN 모델로부터 생성되어진다. 역할 선행 정보는 활성 역할 구성 요소들 사이의 제어의 흐름을 의미한다. 액티비티 선행 정보는 워크플로우 프로시저에서 액티비티들 사이의 실행 흐름을 나타낸다. 데이터 입/출력 정보는 액티비티들의 작업들을 실행하고 생성하기 위해 필요로 되어지는 입/출력 저장소들을 의미한다. 선행 정보는 DDM 으로부터 자동적으로 발생되어질 것이며, 상태 운용 모델(Condition Driven Model: CDM)로 불리어지는 워크플로우 구조의 활성 구성 요소들은 전이 조절과 같은 중요한 역할을 수행한다. 전이 제어 운용 모델(Transition Control Driven Model)은 프로시저에서의 역할 흐름, 데이터 흐름, 액티비티 흐름을 얻어내기 위한 형태를 제공한다.



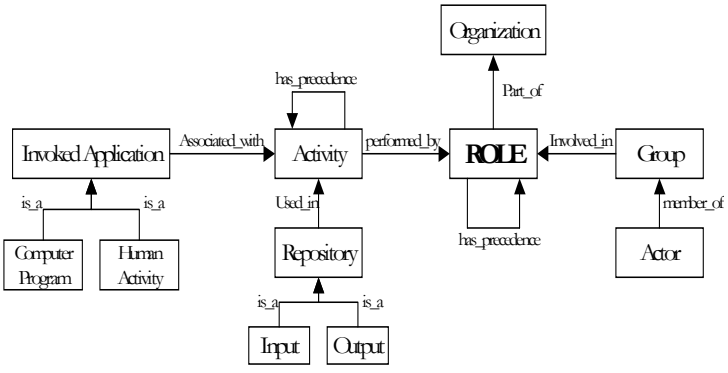
[그림 3] 전형적인 워크플로우 모델의 예

그림 3 은 주문처리에 대한 업무절차를 나타내는 워크플로우 모델이며 하나의 엔드 노드(end-node), 다섯개의 액티비티(a1 ~ a5), 두개의 저장소(r1 과 r2), 하나의 OR 분기와 OR 결합, 열개의 액터(actor), 그리고 다섯개의 역할로 구분된다. 또한 직선 형태로 된 실선은 제어의 흐름을 나타내고 곡선 형태의 실선은 정보의 흐름을 나타낸다. 그림 3 에서 a2, a3, 그리고 a5 는 소수의 액터에 의하여 수행 되어지며 a4 는 동시에 3 개의 액터(h6, h7,h8)가 관여한다[5]. 과거의 전통적인 워크플로우 시스템은 이러한 다수의 액터가 관여하는 경우 기술적 측면이나 구조적 측면에서 효과적인 지원을 하지 못했다. 이러한 경향은 거대하고 복잡하게 변화 되어가는 최근의 작업 환경에서 서로 동시에 작업을 하는 것이 가능하게 하고, 액터의 개입을 최소화 하여 유연성 있어야 하는 새로운 형태의 워크플로우 관리 시스템을 요구하게 되었다. 이와 같은 요구를 충족시키기 위하여 역할 기반 워크플로우 모델을 제안하게 되었다. 즉 역할 기반 워크플로우 모델은 다수의 액터를 역할의 관점에서 재정의 한 것이다.



[그림 4] 역할 기반 모델의 기본적 구상

[그림 4]에서의 왼쪽 그림은 4 개의 역할과 6 개의 액티비티들로 구성되어진 업무절차를 ICN 모델을 이용하여 나타내고 있다. 오른쪽 그림은 ICN 모델에 의하여 작성된 역할 기반 모델의 모양이다. 양쪽의 그림의 가장 큰 차이점은 워크플로우가 실행되기 위한 필요한 정보를 어떻게 기술하는 부분이다. 즉, 왼쪽의 그림은 각각의 액티비티에 역할이 할당되어 있으나, 오른쪽 그림에서는 역할에 액티비티의 집합이 정의되어 있다. 일반적으로 워크플로우 모델은 사용자와 개발자를 위한 인터페이스를 정의한다. 그림 4 의 왼쪽 그림은 사용자에게 충분한 정보를 제공하고 있지만 개발자들에게는 적당치 않다. 예를들어 그림 4 의 내용을 객체지향 기법을 사용하여 워크플로우 시스템을 구축하는 경우 그림 4 의 왼쪽 그림에서 액티비티는 워크플로우 엔진의 컴퍼넌트로 표현될 수 있으나 역할에 대한 정보는 수동적 정보로서 필요시에만 데이터베이스에 저장되어 있는 정보를 엔진에서 이용한다. 이러한 구조는 현재와 같은 작업환경 - 협업 시스템, 협동 그룹웨어 등 –에는 적당치 못하다. 그러나 우측의 그림은 액티비티와 역할이 합쳐져 하나의 요소로서 엔진의 컴퍼넌트로 구성될 수 있고 워크플로우 엔진에서는 커플링 메커니즘을 통하여 협업 시스템 등과 같은 현재와 같은 작업환경과 사용자의 요구에 제공될 수 있다. 워크플로우 모델링 관점에서 살펴볼 때, 위와 같은 시스템을 구축하기 위해서는 다음과 같은 두가지 방법이 있을 수 있다. 첫째, 기본의 ICN 모델로부터 역할 기반 모델을 생성하는 것과 둘째, 워크플로우 프로시져를 새로운 방법으로 모델링 하는 것인데 본 논문에서는 전자의 방법을 택하여 모델링 하는 기법을 제안한다.



[그림 5] 역할 기반 모델의 상호 연관 표현도

그림 5 에서는 역할 기반 모델에서의 워크플로우 요소들 사이의 관계들을 나타내고 있다. 즉, 이전의 워크플로우 모델 정의와는 다르게 역할을 중심으로 하여 기술하였다. 기존의 워크플로우 모델은 역할들이 액티비티들에 의하여 정해진 반면에 역할 기반 모델에서는 액티비티들이 역할에 의해 결정 되어진다. 역할들은 그들의 선행 정보를 가지고 있어야 하며, 동시에 액티비티들도 그들의 선행 정보를 가지고 있다. 활성 역할 구성 요소와 활성 액티비티 구성 요소들로 이루어진 워크플로우 엔진은 이러한 선행 정보들을 바탕으로 워크플로우 프로시저의 실행을 제어할 수 있다. 특히 역할 활성 구성 요소들은 실제 역할과 교신하는 것을 필요로 하는 사람들을 위하여 협력 작용 환경을 제공해야만 한다 [5].

3. 역할 의존성 분석 메커니즘

3.1 ICN(Information Control Net) 모델

기본 ICN 은 4 개의 구성 요소인 프로시저, 액티비티, 선후관계와 자료 저장소들로 구성되어있다. ICN 은 큰 원으로 표현되는 일련의 액티비티와 그에 따른 역할을 포함하고 있으며, 작고 빈 원으로 표현되는 OR 노드, 작고 채워진 원으로 표현되는 AND 노드, 정사각형 모양으로 표현되는 저장소 노드, 그리고 이러한 노드들을 연결하는 선으로 구성되어있다. 선은 실선과 점선으로 표현되는데 이들은 각각 노드들 간의 선후관계 및 자료 저장소와의 입·출력을 표현한다. 또한 ICN 은 OR 노드를 통하여 기존의 모델에서 제공할 수 없었던 선택분기를 지원하며 노드 프로시저를 통해서 알맞은 노드로 분기할 수 있도록 하여 액티비티의 비결정성을 해결할 수 있도록 하였다 [1].

ICN 에서 A 를 일련의 액티비티들의 집합이라 하고, R 을 일련의 자료 저장소의 집합이라 할 때 수식적인 표현과 정의는 다음과 같다.

$\Gamma = (\delta, \gamma, \boldsymbol{I}, \boldsymbol{O})$

A: 액티비티들의 집합

R: 저장소들의 집합

δ: 액티비티 간의 선행관계

γ: 액티비티에서 필요한 입출력 자료 저장소

$\boldsymbol{I}$: 초기 입력 자료 저장소의 집합

$\boldsymbol{O}$: 마지막 출력 자료 저장소의 집합

[표 1] ICN 모델의 정형적 정의

• I 는 초기에 입력되는 자료 저장소의 유한집합이며 ICN 의 실행 전에 어떠한 외부의 프로세스에 의하여 로드(load) 되어져야 한다고 가정한다.

• O 는 마지막으로 출력되는 자료 저장소들의 유한 집합이며 ICN 의 실행 후에 어떠한 외부의 프로세스에 의해서 이용되는 정보를 포함하고 있다고 가정한다.

• $\delta = \delta_i \cup \delta_o$

$\delta_i : \boldsymbol{A} \rightarrow \wp(\boldsymbol{A})$는 하나의 액티비티를 선행하는 액티비티들의 집합에 연결하는 관계를 나타내며, $\delta_o : \boldsymbol{A} \rightarrow \wp(\boldsymbol{A})$는 하나의 액티비티를 후행하는 액티비티들의 집합에 연결하

는 관계를 나타낸다.

• $\gamma = \gamma_i \cup \gamma_o$

γ_i : $A \rightarrow \wp(R)$은 하나의 액티비티를 선행하는 액티비티 집합들을 입력 자료 저장소들의 집합과 연결하는 관계를 나타내며, γ_o : $A \rightarrow \wp(R)$은 하나의 액티비티를 후속하는 액티비티 집합들을 출력 자료 저장소들의 집합과 연결하는 것을 나타낸다.

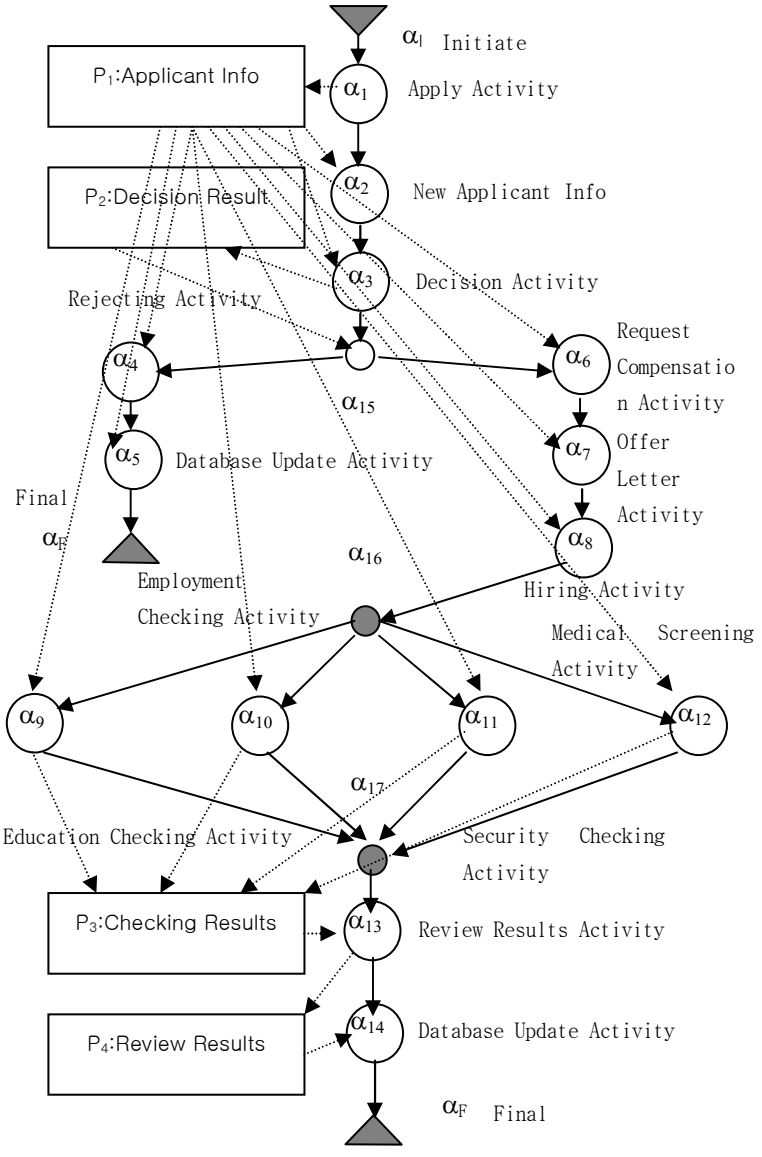
위의 ICN 모델 정의를 바탕으로 [그림 3]의 공식화된 표현의 예를 들어 보면 [표 2]와 같다.

$A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$
$R = \{r_1, r_2\}$
$I = \{\}$
$O = \{\}$

$\delta_i(a_1)=((\lambda)), \delta_o(a_1)=((a_6)); \gamma_i(a_1)=(\lambda), \gamma_o(a_1)=(r_1)$
$\delta_i(a_2)=(a_6), \delta_o(a_2)=((a_7)); \gamma_i(a_2)=(r_1), \gamma_o(a_2)=(r_2)$
$\delta_i(a_3)=(a_6), \delta_o(a_3)=((a_4)); \gamma_i(a_3)=(r_1), \gamma_o(a_3)=(\lambda)$
$\delta_i(a_4)=(a_3), \delta_o(a_4)=((a_7)); \gamma_i(a_4)=(r_1), \gamma_o(a_4)=(\lambda)$
$\delta_i(a_5)=(a_7), \delta_o(a_4)=(\lambda); \gamma_i(a_5)=(r_1, r_2), \gamma_o(a_5)=(\lambda)$
$\delta_i(a_6)=(a_1), \delta_o(a_6)=((a_2), (a_3)); \gamma_i(a_6)=(\lambda), \gamma_o(a_6)=(\lambda)$
$\delta_i(a_7)=(\varnothing, a_4), \delta_o(a_7)=((a_5)); \gamma_i(a_7)=(r_2), \gamma_o(a_7)=(\lambda)$

[표 2] 주문처리 워크플로우 모델의 공식화된 표현

3.2 역할 의존성 분석



[그림 6] 워크플로우 모델에서의 확장되어진 ICN

그림 6 에서는 확장되어진 ICN 모델을 이용한 워크플로우 모델을 나타내고 있는 그림이며, ICN 모델의 공식화되어진 표현은 아래와 같다.

$\Gamma = (\delta, \gamma, \varepsilon, \pi, \kappa, I, O)$ over A, R, P, C, T //Extended ICN

$A = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_9, \alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}, \alpha_I, \alpha_F\}$ //Activities

$R = \{p_1, p_2, p_3, p_4\}$ // Repositories

$P = \{\eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7\}$ // Roles

$C = \{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_8, o_9\}$; where $\{o_7, o_8\}$ = group-actor // Actors

$T = \{\text{default}, \text{or}_1(\text{hire='Yes'}), \text{or}_1(\text{hire='No'}), \text{and}_1(\text{default})\}$ // Control-transition conditions

$I = \{\varnothing\}$

$O = \{\varnothing\}$

$\delta_i(\alpha_1)=\{(\alpha_1)\}, \delta_o(\alpha_1)=\{(\alpha_2)\};$	$\gamma_i(\alpha_1)=\{\varnothing\}, \gamma_o(\alpha_1)=\{p_1\};$	$\varepsilon_p(\alpha_1)=\{\eta_1\};$
$\delta_i(\alpha_2)=\{(\alpha_1)\}, \delta_o(\alpha_2)=\{(\alpha_3)\};$	$\gamma\gamma_i(\alpha_4)=\{p_1\}, \gamma_o(\alpha_4)=\{\varnothing\};$	$\varepsilon_p(\alpha_2)=\{\eta_2\};$
$\delta_i(\alpha_3)=\{(\alpha_2)\}, \delta_o(\alpha_3)=\{(\alpha_{15})\};$	$\gamma_i(\alpha_5)=\{p_1\}, \gamma_o(\alpha_5)=\{\varnothing\};$	$\varepsilon_p(\alpha_3)=\{\eta_3\};$
$\delta_i(\alpha_4)=\{(\alpha_{15})\}, \delta_o(\alpha_4)=\{(\alpha_5)\};$	$\gamma_i(\alpha_6)=\{p_1\}, \gamma_o(\alpha_6)=\{p_1\};$	$\varepsilon_p(\alpha_4)=\{\eta_2\};$
$\delta_i(\alpha_5)=\{(\alpha_4)\}, \delta_o(\alpha_5)=\{(\alpha_F)\};$	$\gamma_i(\alpha-)=\{p_1\}, \gamma_o(\alpha_7)=\{p_1\};$	$\varepsilon_p(\alpha_5)=\{\eta_9\};$
$\delta_i(\alpha_6)=\{(\alpha_{15})\}, \delta_o(\alpha_6)=\{(\alpha_7)\};$	$\gamma_i(\alpha_8)=\{p_1\}, \gamma_o(\alpha_8)=\{p_1\};$	$\varepsilon_p(\alpha_6)=\{\eta_2\};$
$\delta_i(\alpha_7)=\{(\alpha_6)\}, \delta_o(\alpha_7)=\{(\alpha_8)\};$	$\gamma_i(\alpha_9)=\{p_1\}, \gamma_o(\alpha_9)=\{p_3\};$	$\varepsilon_p(\alpha_7)=\{\eta_2\};$
$\delta_i(\alpha_8)=\{(\alpha_7)\}, \delta_o(\alpha_8)=\{(\alpha_{16})\};$	$\gamma_i(\alpha_{10})=\{p_1\}, \gamma_o(\alpha_{10})=\{p_3\};$	$\varepsilon_p(\alpha_8)=\{\eta_2\};$
$\delta_i(\alpha_9)=\{(\alpha_{16})\}, \delta_o(\alpha_9)=\{(\alpha_{17})\};$	$\gamma_i(\alpha_{11})=\{p_1\}, \gamma_o(\alpha_{11})=\{p_3\};$	$\varepsilon_p(\alpha_9)=\{\eta_4\};$
$\delta_i(\alpha_{10})=\{(\alpha_{16})\}, \delta_o(\alpha_{10})=\{(\alpha_{17})\};$	$\gamma_i(\alpha_{12})=\{p_1\}, \gamma_o(\alpha_{12})=\{p_3\};$	$\varepsilon_p(\alpha_{10})=\{\eta_4\};$
$\delta_i(\alpha_{11})=\{(\alpha_{16})\}, \delta_o(\alpha_{11})=\{(\alpha_{17})\};$	$\gamma_i(\alpha_{13})=\{p_3\}, \gamma_o(\alpha_{13})=\{p_4\};$	$\varepsilon_p(\alpha_{11})=\{\eta_5\};$
$\delta_i(\alpha_{12})=\{(\alpha_{16})\}, \delta_o(\alpha_{12})=\{(\alpha_{17})\};$	$\gamma_i(\alpha_{14})=\{p_4\}, \gamma_o(\alpha_{14})=\{\varnothing\};$	$\varepsilon_p(\alpha_{12})=\{\eta_6\};$
$\delta_i(\alpha_{13})=\{(\alpha_{17})\}, \delta_o(\alpha_{13})=\{(\alpha_{14})\};$	$\gamma_i(\alpha_{15})=\{p_2\}, \gamma_o(\alpha_{15})=\{\varnothing\};$	$\varepsilon_p(\alpha_{13})=\{\eta_2\};$
$\delta_i(\alpha_{14})=\{(\alpha_{13})\}, \delta_o(\alpha_{14})=\{(\alpha_F)\};$	$\gamma_i(\alpha_{16})=\{\varnothing\}, \gamma_o(\alpha_{16})=\{\varnothing\};$	$\varepsilon_p(\alpha_{14})=\{\eta_7\};$
$\delta_i(\alpha_{15})=\{(\alpha_3)\}, \delta_o(\alpha_{15})=\{(\alpha_4),(\alpha_6)\};$	$\gamma_i(\alpha-)=\{\varnothing\}, \gamma_o(\alpha_{17})=\{\varnothing\};$	$\varepsilon_p(\alpha_{15})=\{\eta_7\};$
$\delta_i(\alpha_{16})=\{(\alpha_8)\}, \delta_o(\alpha_{16})=\{(\alpha_9,\alpha_{10},\alpha_{11},\alpha_{12})\};$	$_i(\alpha_2)=\{p_1\}, \gamma_o(\alpha_2)=\{\varnothing\};$	$\varepsilon_p(\alpha_{16})=\{\eta_7\};$
$\delta_i(\alpha_{17})=\{(\alpha_9,\alpha_{10},\alpha_{11},\alpha_{12})\}, \delta_o(\alpha_{17})=\{(\alpha_{13})\};$	$\gamma_i(\alpha_3)=\{p_1\}, \gamma_o(\alpha_3)=\{p_2\};$	$\varepsilon_p(\alpha_{17})=\{\eta_7\};$

$\pi_c(\eta_1)=\{o_1\};$	$\kappa_i(\alpha_1)=\{d\}, \kappa_o(\alpha_1)=\{d\};$
$\pi_c(\eta_1)=\{o_2, o_3, o_4\};$	$\kappa_i(\alpha_2)=\{d\}, \kappa_o(\alpha_2)=\{d\};$
$\pi_c(\eta_3)=\{o_5\};$	$\kappa_i(\alpha_3)=\{d\}, \kappa_o(\alpha_3)=\{d\};$
$\pi_c(\eta_4)=\{o_6\};$	$\kappa_i(\alpha_4)=\{\text{or}_1(\text{hire='No'})\}, \kappa_o(\alpha_4)=\{d\};$
$\pi_c(\eta_5)=\{o_7\};$	$\kappa_i(\alpha_5)=\{d\}, \kappa_o(\alpha_5)=\{d\};$
$\pi_c(\eta_6)=\{o_8\};$	$\kappa_i(\alpha_6)=\{\text{or}_1(\text{hire='No'})\}, \kappa_o(\alpha_6)=\{d\};$
$\pi_c(\eta_7)=\{o_9\};$	$\kappa_i(\alpha_7)=\{d\}, \kappa_o(\alpha_7)=\{d\};$
	$\kappa_i(\alpha_8)=\{d\}, \kappa_o(\alpha_8)=\{d\};$
	$\kappa_i(\alpha_9)=\{\pi_1\}, \kappa_o(\alpha_9)=\{\text{and}_1(d)\};$
	$\kappa_i(\alpha_{10})=\{\pi_2\}, \kappa_o(\alpha_{10})=\{\text{and}_1(d)\};$
	$\kappa_i(\alpha_{11})=\{\pi_3\}, \kappa_o(\alpha_{11})=\{\text{and}_1(d)\};$
	$\kappa_i(\alpha_{12})=\{\pi_4\}, \kappa_o(\alpha_{12})=\{\text{and}_1(d)\};$
	$\kappa_i(\alpha_{13})=\{d\}, \kappa_o(\alpha_{13})=\{d\};$

[표 3] 고용 워크플로우 모델의 공식화된 표현

역할 의존 넷(Role Dependent Net)은 프로시저에 존재하는 역할들 사이의 변화 순서들을 나타내며, 프로시저 드라이븐 모델(Procedure Driven Model)에서 제어 흐름 부분인 δ와 역할 지정 부분인 ε의 집합들로 구성되어 있다. 역할 의존 넷은 결국 워크플로우 프로시저의 역할 변화 제어들을 생성한다.

역할 의존 넷은 일반적으로 $\Lambda=(\sigma, \psi, S, E)$로서 정의되며, 역할들의 집합인 P 와 액티비티들의 집합인 A 를 바탕으로 구성된다. P 는 η로 표현되어진 역할들을 가지고

있으며 A 는 α로 표현되어진 액티비티들을 가지고 있다.

- $\sigma = \sigma_I \cup \sigma_O$

  σ_o: $P \rightarrow \wp(P)$는 자신을 포함한 후행 집합에서 역할의 단일 값을 갖는 함수이며, 액티비티들을 수행하고 있는 후속 작업을 의미한다. σ_I: $P \rightarrow \wp(P)$는 선행 집합에서 역할의 단일 값을 갖는 함수이며, 이전의 역할 또는 현재의 역할이 액티비티를 수행하는 것을 의미한다.
- $\psi = \psi_i \cup \psi_o$

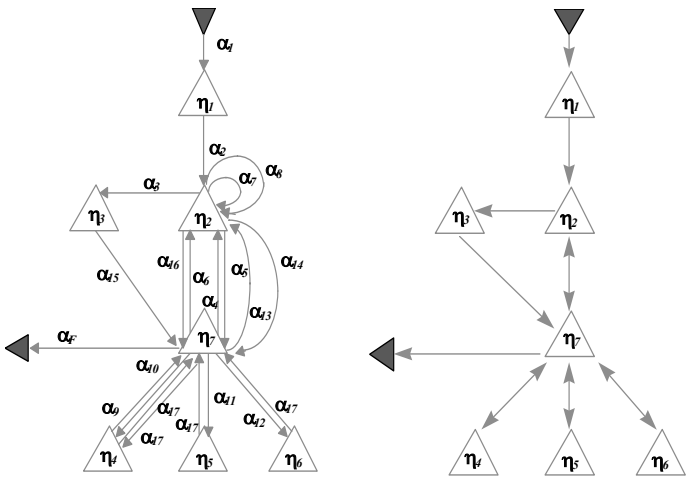
  ψ_i 는 역할 의존 넷 구조상에서의 선행 역할과 관련되어지는 전체 액티비티에 포함되는 α의 집합을 의미한다. ψ_o 는 역할 의존 넷 구조상에서의 후행 역할과 관련 되어지는 전체 액티비티에 포함되는 α의 집합을 의미한다.
- S 는 어떤 외부의 프로시저에 의하여 로드(load) 되어질 것이라고 가정한 최초 액티비티들의 집합을 나타낸다.
- E 는 어떤 외부의 프로시저에 포함되어진 액티비티들 중 마지막 액티비티들의 집합을 나타낸다.

이러한 역할 의존 넷의 공식화된 표현은 아래와 같다.

$\Lambda = (\sigma, \psi, S, E)$ over A, P // Role Dependent Net	
$A = \{ \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_9, \alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}, \alpha_I, \alpha_F \}$ // Activities	
$P = \{ \eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7, \eta_I, \eta_F \}$ // Roles	
$S = \{ \varnothing \}$; $E = \{ \varnothing \}$	
$\sigma_i(\eta_I)=\{\varnothing\}, \sigma_o(\eta_I)=\{\eta_1\}$;	$\psi_i(\eta_I)=\{ \varnothing \}, \psi_o(\eta_I)=\{ \alpha_1\}$;
$\sigma_i(\eta_1)=\{\eta_1\}, \sigma_o(\eta_1)=\{\eta_2\}$;	$\psi_i(\eta_1)=\{ \alpha_1\}, \psi_o(\eta_1)=\{ \alpha_2\}$;
$\sigma_i(\eta_2)=\{\eta_1\}$, $\sigma_o(\eta_2)=\{ \eta_2, \eta_3, \eta_7 \}$;	$\psi_i(\eta_2)=\{ \alpha_2, \alpha_4, \alpha_6, \alpha_7, \alpha_8, \alpha_{13} \}$, $\psi_o(\eta_2)=\{ \alpha_3, \alpha_5, \alpha_7, \alpha_8, \alpha_{14}, \alpha_{16} \}$;
$\sigma_i(\eta_3)=\{\eta_2\}, \sigma_o(\eta_3)=\{\eta_7\}$;	$\psi_i(\eta_3)=\{ \alpha_3\}, \psi_o(\eta_3)=\{ \alpha_{15}\}$;
$\sigma_i(\eta_4)=\{\eta_7\}, \sigma_o(\eta_4)=\{\eta_7\}$;	$\psi_i(\eta_4)=\{ \alpha_9, \alpha_{10}\}, \psi_o(\eta_4)=\{ \alpha_{17}\}$;
$\sigma_i(\eta_5)=\{\eta_7\}, \sigma_o(\eta_5)=\{\eta_7\}$;	$\psi_i(\eta_5)=\{ \alpha_{11}\}, \psi_o(\eta_5)=\{ \alpha_{17}\}$;
$\sigma_i(\eta_6)=\{\eta_7\}, \sigma_o(\eta_6)=\{\eta_7\}$;	$\psi_i(\eta_6)=\{ \alpha_{12}\}, \psi_o(\eta_6)=\{ \alpha_{17}\}$;
$\sigma_i(\eta_7)=\{\eta_2, \eta_3, \eta_4, \eta_5, \eta_6\}$, $\sigma_o(\eta_7)=\{\eta_2, \eta_4, \eta_5, \eta_6, \eta_F\}$;	$\psi_i(\eta_7)=\{ \alpha_5, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}\}$, $\psi_o(\eta_7)=\{ \alpha_4, \alpha_6, \alpha_9, \alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_F \}$;
$\sigma_i(\eta_F)=\{\eta_7\}, \sigma_o(\eta_F)=\{ \varnothing \}$;	$\psi_i(\eta_F)=\{ \alpha_F\}, \psi_o(\eta_F)=\{ \varnothing \}$;

[표 4] 역할 의존 넷의 공식

위의 확장되어진 ICN 모델의 공식에 의하여 역할 의존넷을 구성할 수 있다.



[그림 7] 역할 의존 넷(Role Dependency Net)

[그림 7] 은 역할 의존 넷의 공식화되어진 표현에 의해 구성된 역할 의존 넷을 도식화한 것이다. 왼쪽 그림은 역할 의존 넷의 도식화된 표현이며, 역할들 사이의 제어 변화들 뿐만 아니라 역할을 수행하는 액티비티들의 실행 흐름도 보여주고 있다. 오른쪽 그림은 단지 역할 사이의 의존 관계를 표현하기 위한 역할 의존 넷을 도식화 하였다. 단일 화살표는 직접적인 의존 관계를 나타내며, 이중 화살표는 두 역할들 사이의 다양한 의존 관계를 나타내고 있다.

이러한 방식에 의해 구성되어진 역할 의존 넷을 이용한 알고리즘은 다음과 같다.

INPUT : A procedure driven model;
OUTPUT : A role-dependent net;
BEGIN
FOR all $x \in A$ in an extended ICN
  // Get a role flow subnet.
ADD $\varepsilon_p(x)$ TO $\sigma_i(\varepsilon_p(\delta_o(x)))$;
ADD $\varepsilon_p(\delta_o(x))$ TO $\sigma_o(\varepsilon_p(x))$;
  // Get an activity on the arc between two roles.
ADD x TO $\psi_i(\varepsilon_p(\delta_o(x)))$;
ADD x TO $\psi_o(\varepsilon_p(x))$;
ROF;
END.

4. 결론 및 향후 발전 방향

본 논문에서는 협력 시스템들을 위한 역할 기반의 모델 개념에 관하여 소개하였다. ICN 모델링 기법을 이용하여 표현된 워크플로우 모델로부터 역할 의존성을 분석하여 역할 의존 넷을 작성하였으며, 그 알고리즘을 제안하였다. 이와 같이 역할 의존성을 분석하여 작성된 워크플로우 모델은 오늘날과 같은 협력 시스템 환경하의 워크플로우 관리 시스템에 적용될 수 있으며, 워크플로우 관리 시스템과 관련된 응용프로그램의 작성을 최소화 할 수 있다. 향후 연구 과제로는 본 논문에서 제시된 역할 의존성 분석을 통하여 모델링 및 시뮬레이션 기능을 제공하기 위한 역할 기반 모델링을 수행하는 모델링 툴을 개발하여 다양한 작업환경에서 사용될 수 있는 역할 기반 워크플로우 시스템을 구현하는 것이다.

참고 문헌

[1] Clarence A. Ellis and Gary J. Nutt, “Office Information Systems and Computer Science”, Computing Surveys, Vol. 12, No. 1, March 1980.

[2] Kwang-Hoon Kim and Su-Ki Paik, “Actor-Oriented Workflow Model”, The Second Cooperative Database Systems for Advanced Applications, Wollongong Australia, March 1999.

[3] Clarence A. Ellis, “Formal and Informal Models of Office Activity”, Proceedings of the 1983 Would Computer Congress, Paris, France, April 1983.

[4] Andy Podgurski and Lori A. Clarke, “A Formal Model of Program Dependencies and Its Implications for Software Testing, Debugging, and Maintenance”, IEEE Trans. On SE, Vol. 16, No. 9, Sep. 1990.

[5] Clarence A. Ellis, Gary J. Nutt, “The Modeling and Analysis of Coordination Systems”, University of Colorado/Dept. of Computer Science Technical Report, CU-CS-639-93, Jan. 1993

[6] Clarence A. Ellis, “Formal and Informal Models of Office Activity”, Proceedings of the 1983 Would Computer Congress, Paris, France, April 1983

[7] “The Business Imperative for Workflow & Business Process Reengineering”, A Special Advertising Section, Fortune, Feb. 1996.