다양한 병행 종속성을 포함한 워크플로우 모델링 및 검증

정 희 택* · 이 도 헌*

Modeling and Verification of Workflows with Various Parallel Dependencies

Hee-Taek Ceong* · Do-Heon Lee*

Abstract

A study on workflow system as an automated business processing system is done recently. However, it did not consider various dependencies between parallel tasks. Therefore, this paper proposes modeling and verification of workflows with various parallel dependencies. For this, firstly, we propose four dependencies to specify various parallel dependencies between tasks. They contain sequential starts, parallel starts, sequential commits, and parallel commits. Secondly, we suggest a method to specify various parallel dependencies on workflow graph. Thirdly, we propose a verification method to detect contradictions on workflow specifications.

※ 본 연구는 정보통신부의 정보통신분야 우수학교 지원사업에 의하여 수행한 연구 결과임.
* 전남대학교 전산학과

1. 서 론

워크플로우는 자동화된 업무 흐름을 말한다. 워크플로우 시스템은 컴퓨터와 통신망을 이용하여 체계적으로 워크플로우를 정의, 관리, 그리고 수행하는 시스템이다[Elmagarmid 1997, Rusinkiewicz 1994, Chrysanthis 1994]. 최근 기업의 업무 구조가 복잡해짐에 따라 워크플로우 시스템에 대한 관심이 크게 높아지고 있다[Leymann 1997].

워크플로우 시스템에서 중요한 인자인 모델링 및 검증은 태스크 및 그들간의 종속성 요소를 기반으로 수행되고 있다. 기존 연구로 [Rusinkiewicz 1994, Chrysanthis 1992, Klein 1991, Georgakopoulos 1994, Georgakopoulos 1996, Rusinkiewicz 1990, Tatbul 1997, Attie 1993, Davulcu 1998]에서는 태스크를 시작, 완료 그리고 철회의 3가지 요소로 정의하고 태스크간의 종속성으로 순차 및 단순 병행종속성을 정의하였다. [Rusinkiewicz 1994, Chrysanthis 1992, Klein 1991, Georgakopoulos 1994, Georgakopoulos 1996, Rusinkiewicz 1990, Tatbul 1997]에서는 워크플로우 모델링을 위한 요소만을 제안하였을 뿐 일관성 검증 방안을 제안하지 않았다. 한편, [Attie 1993, Davulcu 1998]에서는 논리를 기반으로 형식화(logic-based formalism)할 수 있는 워크플로우 명세언어를 제안하였고 이를 기반으로 워크플로우를 명세, 검증, 그리고 스케줄링 하였다. 한편, W.M.P. van der Aalst의 연구[1994, 1996a, 1996b, 1998]와 Adam[1998]에서는 패트리넷(petri-net)을 이용한 워크플로우 모델링 및 검증 방안이 연구되었다. W.M.P. van der Aalst[1994, 1996a, 1996b, 1998]의 연구에서는 패트리넷의 플레이스(place) 및 트랜지션(transition)을 이용하여 워크플로우를 모델링 하였다. 즉, 태스크를 트랜지션으로, 워크플로우의 상태변화를 플레이스로 모델링 하였다. 워크플로우의 검증은 기존에 연구된 패트리넷 분석방법을 통해 달성하였다. 즉, 도달성(reachability) 및 지속성(liveness)을 기반으로 이루어졌다. 또한, Adam[1998]에서는 앞선 연구와 달리 태스크의 상태를 플레이스로, 시작, 완료준비, 완료 그리고 철회의 사건을 트랜지션으로 각각 모델링 하였다. 워크플로우 명세를 검증하기 위해 먼저, 패트리넷 상에서 해당 플레이스로의 전이를 보장할 수 없는 홉수관(siphon) 존재여부를 조사하거나 패트리넷이 최종 상태에 도달 가능한지 여부를 이용하여 검증하였다. 또한, 트랜지션의 시간제약에 대해 최소/최대 분석을 통해 트랜지션의 수행가능 여부를 검증하였다. 마지막으로, 워크플로우 관리 연합에서 제안한 참고 모델과 프로세스 모델은 태스크간의 제어 흐름으로 순차 수행(sequential routing) 및 병행 수행(parallel routing)만을 제안하였다[Hollingsworth 1994, Work Group1 1998]. 기존에 이루어진 이러한 모델링 연구들은 본 연구에서 고려하고 있는 다양한 병행 종속성을 고려하고 있지 않으며, 그에 따른 모순 검증을 고려하고 있지 않다.

워크플로우 시스템에서 처리 객체로써 업무 흐름은 요구된 목적을 달성하기 위해 필요한 요소들로 일관성 있게 명세되어야 한다. 업무 흐름의 명세는 업무들 각각을 의미하는 태스크들과 그들간의 종속성으로 기술된다. 워크플로우 시스템에서 고려하는 태스크간 종속성의 예로는 순차, 분기, 반복, 병행 등이 있다. 특히 병행은 동시에 둘 이상의 태스크가 수행되어야 함을 명시하는 것으로 워크플로우 환경에서 필수형태이다. 두 개의 태스크를 병행 수행하는 형태에는 다양한 요구사항이 있을 수 있다. 예를 들면, 보험 워크플로우에서 병행 수행되는 신규 접수 태스크 및 종합검진 태스크간의 요구사항을 고려할 수 있다. 즉, 신규 접수 태스크가 시작된 후 종합검진 태스크가 시작될 수 있으며 종합검진 태스크가 종료되었을 때에만 신규 접수 태스크가 종료되어야 하는 요구사항이 존재한다. 기존에 워크플로우 모델링 연구에서는 제시된 요구사항을 모델링 할 수 없다. 최근에 [정희택 & 이도헌 1998] 연구에서 몇 가지 병행 종속성 형태를 제안하였다. 즉, 병행 수행하는 태스크들간에 중첩 종속성, 간섭된 종속성, 동시시작 종속성,

동시 종료 종속성, 그리고 동시 시작 및 종료 종속성을 제안하였다. 그러나, 제안된 병행 종속성은 태스크들간의 시작 사건과 종료 사건간의 관계를 명시적으로 모델링 함으로써 융통성 있는 태스크간 종속성을 모델링 할 수 없다. 즉, 시작 사건들만의 종속성이나 종료 사건들만의 종속성을 모델링 할 수 없다. 이러한 예로, 여행 예약 워크플로우에서 특정 호텔예약에 대해 특정 회사의 차량 대여를 선호할 때, 병행 수행된 태스크에서 차량 대여 태스크가 완료되었을 때만 호텔 예약 태스크의 완료를 요구하는 완료 사건간의 종속성을 표현할 수 없다. 또한 제안된 연구에서는 수행가능 경로간에 존재하는 모순과 자가 종속 및 다중 종속 모순을 검증할 수 없다.

본 연구에서는 다양한 병행 종속성을 포함한 워크플로우 모델링 및 일관성 검증 방안을 제안한다. Ⅱ장에서는, 다양한 병행 종속성을 제시하고 이를 그래프로 쉽게 표현할 수 있는 워크플로우 모델링 방안을 제안한다. Ⅲ장에서는 다양한 병행 종속성을 포함한 워크플로우 명세에 존재할 수 있는 모순을 제시하고 이를 검증할 수 있는 방안을 제안한다. Ⅳ장에서는 본 연구에 대한 결론을 기술한다.

2. 다양한 병행 종속성을 포함한 워크플로우 모델링

태스크들과 그들간의 종속성으로 구성되는 워크플로우는 다양한 병행 수행 요구를 갖는다. 워크플로우의 목적을 달성하기 위해, 병행 수행하는 태스크들간에 다양한 병행 종속성을 요구한다. 본 장에서는 먼저, 태스크간 존재하는 다양한 병행 종속성을 제안한다. 다음으로 워크플로우 그래프 상에서 다양한 병행 종속성의 모델링 방안을 제안한다.

2.1 병행 종속성의 형태

업무흐름을 구성하는 태스크들은 순차 및 단순 병행 종속성 외에도 다양한 병행 종속성을 갖는다. 직접적 순차 순서 관계를 갖는 태스크들은 순차 종속성을 갖고 병행수행을 요하는 태스크들은 단순 병행 종속성을 갖는다. 그러나, 병행 수행하는 태스크간에도 요구사항을 모델링하기 위해 다양한 병행 종속성이 필요하다. 즉, 특정 태스크 수행 후 병행 수행되는 태스크들간에도 시작 및 완료 종속성이 존재한다. 예를 들면, 병행 수행되는 두 태스크에 대해, 한 태스크가 먼저 완료되었을 때만 병행 수행중인 다른 태스크가 완료되어야 하는 요구조건이 존재한다. 이러한 요구조건을 모델링하기 위해 태스크간 병행 종속성이 필요하다.

병행 수행되는 태스크간에 존재할 수 있는 종속성은 시작 및 완료의 수행 순서에 따라 각각 3가지 형태를 갖는다. 태스크들의 시작에 의한 종속성과 완료에 의한 종속성 형태는 <표 1>과 같다. 표에서 < 과 = 는 태스크에서 시작 및 완료 사건의 발생순서 관계를 의미한다. 두 태스크간에 시작 순서는 순차 및 병행 수행될 수 있으며, 순차에서는 서로 다른 수행 순서에 따라 2가지로 구분된다.

〈표 1〉 시작 및 완료순서에 따른 분류

시작 순서에 따른 분류	T_i T_j $St_{Ti} < St_{Tj}$	T_i T_j $St_{Ti} = St_{Tj}$	T_i T_j $St_{Ti} > St_{Tj}$
완료 순서에 따른 분류	T_i T_j $Cm_{Ti} < Cm_{Tj}$	T_i T_j $Cm_{Ti} = Cm_{Tj}$	T_i T_j $Cm_{Ti} > Cm_{Tj}$

<표 1>에서 구분된 시작 및 완료 수행 순서를 기반으로 다양한 병행 수행 종속성을 정의할 수 있다. 한편, 두 태스크간의 이진 관계를 고려할 때, 태스크 식별자의 순서만 다른 동형(isomorphic)의 종속성을 하나로 정의하면 <표 2>와 같다. 즉, St_{Ti} <St_{Tj}와 St_{Ti}> St_{Tj}는 식별자의 순서만 다른 동형의 종속성으로 구분한다. 그러나, 순서관계의 의미는 방향 그래프를 이용한 워크플로우 명세에서 방향을 갖는 간선으로 표현된다. 구분된 종속성들은 병행 수행되는 태스크들간의 다양한 요구조건을 모델링 할 수 있다.

〈표 2〉 다양한 병행 종속성 종류

종속성 기호	시작 순서	완료 순서
SS(Sequential Starts)	$St_{Ti} < St_{Ti}$	-
PS(Parallel Starts)	$St_{Ti} = St_{Ti}$	-
SC(Sequential Commits)	-	$Cm_{Ti} < Cm_{Ti}$
PC(Parallel Commits)	-	$Cm_{Ti} = Cm_{Ti}$
IE(Interleaved Execution)	$St_{Ti} < St_{Tj}$	$Cm_{Ti} < Cm_{Ti}$
SSPC(Sequential Starts and Parallel Commits)	$St_{Ti} < St_{Tj}$	$Cm_{Ti} = Cm_{Ti}$
NE(Nested Execution)	$St_{Ti} < St_{Tj}$	$Cm_{Ti} > Cm_{Ti}$
PSSC(Parallel Starts and Sequential Commits)	$St_{Ti} = St_{Tj}$	$Cm_{Ti} < Cm_{Ti}$
PSPC(Parallel Starts and Paralle Commits)	$St_{Ti} = St_{Tj}$	$Cm_{Ti} = Cm_{Ti}$

<표 2>에서 제시된 종속성에서 IE, SSPC, NE, PSSC, PSPC는 태스크들간의 종속성 요소로써 제안되었고[1][정희택 & 이도헌 1998], 이를 본 연구에서는 태스크들간의 시작 및 완료 사건의 종속성으로 확장한다. 즉, 시작 사건들간의 종속성(즉, SS, PS)과 완료 사건들간의 종속성(즉, SC, PC)을 제안한다. 본 연구에서 제안한 각 종속성의 의미 및 표현 가능한 요구는 다음과 같다.

- SS : 병행 수행되는 두 태스크간의 순서화 된 시작을 의미한다. 즉, 병행 수행되는 태스크중 한 태스크가 시작되었을 때만 다른 태스크가 병행 수행될 수 있음을 모델링 할 수 있다. 예를 들면, 공항 업무 워크플로우에서 승객 하선 태스크 시작 후 승객 수하물 하역 업무가 시작되어야 함을 모델링 할 수 있다.
- PS : 병행 수행되는 두 태스크간의 동시 시작을 의미한다. 즉, 병행 수행되는 두 태스크 모두가 수행가능 할 때만 시작되어야 함을 모델링 할 수 있다.
- SC : 병행 수행되는 두 태스크는 선행 태스크가 완료되었을 때만 후행 태스크가 종료될 수 있음을 모델링 한다. 즉, 선행 태스크의 완료가 후행 태스크의 완료에 필수 요소임을 모델링 할 수 있다. 예를 들면, 여행 예약 워크플로우에서 호텔예약 태스크가 완료되었을 때만 비행기 예약 태스크가 완료될 수 있음을 모델링 할 수 있다.
- PC : 병행 수행되는 두 태스크의 동시 완료를 의미한다. 즉, 두 태스크 모두가 완료될 수 있을 때만 완료될 수 있음을 모델링 할 수 있다. 예를 들면, 은행 워크플로우에서 고객의 신용도 조회 태스크와 대출에 따른 잔고 산출 태스크 모두가 완료되어야 함을 모델링 할 수 있다.

IE, SSPC, NE, PSSC, 그리고 PSPC는 본 연구에서 제안한 SS, PS, SC, 그리고 PC의 요소들로 표현할 수 있다. 이는 다음과 같다.

정리 1. 임의의 태스크 T_i, T_j에 대해 다음을 만족한다.

$IE(T_i,T_j) \Leftrightarrow SS(T_i,T_j) \wedge SC(T_i,T_j)$
$SSPC(T_i,T_j) \Leftrightarrow SS(T_i,T_j) \wedge PC(T_i,T_j)$
$NE(T_i,T_j) \Leftrightarrow SS(T_i,T_j) \wedge SC(T_j,T_i)$
$PSSC(T_i,T_j) \Leftrightarrow PS(T_i,T_j) \wedge SC(T_i,T_j)$
$PSPC(T_i,T_j) \Leftrightarrow PS(T_i,T_j) \wedge PC(T_i,T_j)$.

증명. <표 2>에 의해 자명함으로 생략한다. □

2.2 다양한 병행 종속성을 포함한 워크플로우 그래프

워크플로우 모델링 요소를 이용한 워크플로우 명세는 그래프를 기반으로 기술한다. 워크플로우를 구성하는 태스크는 노드로 표현되며 그들간의 종속성은 간선으로 표현된다. 그래프에는 초기 시작 태스크를 의미하는 노드와 마지막으로 수행될 종료 태스크가 하나씩 존재한다. 다양한 병행 종

1) [정희택 & 이도헌 1998] 연구에서는 SSPC를 ST(Simultaneous Terminations)으로, PSSC를 SS(Simultaneous Starts)로, PSPC를 PE(parallel Execution)으로 표현하였을 뿐 의미는 동일하다.

속성을 포함한 워크플로우 그래프는 아래와 같이 정의한다.

정의 1. 워크플로우 그래프

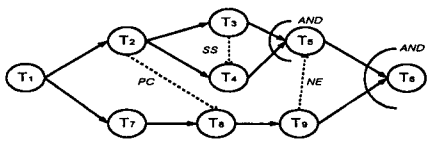
워크플로우 그래프는 $WFG=(V,E)$는 다음과 같이 정의되는 방향 그래프이다. V는 태스크의 집합이고 E는 제어 연결자(control connector)의 집합이다.

각 태스크 $t \in V$는 태스크 식별자, AND 연결 형태를 갖는 제어 연결자의 식별자 집합, OR 연결 형태를 갖는 제어 연결자의 식별자 집합, 대안태스크 집합과 같은 네 개의 속성으로 구성된다. 여기에서, 태스크 식별자는 태스크 t를 유일하게 구별하게 한다. AND 연결 형태를 갖는 제어 연결자의 식별자 집합은 선행 태스크 모두가 완료될 때 태스크 t로의 전이를 가능하게 하는 제어 연결자들의 집합이다. OR 연결 형태를 갖는 제어 연결자의 식별자 집합은 선행 태스크들 중 *하나*의 완료로부터 태스크 t로의 전이를 가능하게 하는 제어 연결자들의 집합이다. 대안 태스크 집합은 태스크 t가 철회되었을 때 수행될 수 있는 대안 태스크의 집합이다.

각 제어 연결자 $e \in E$는 식별자, 시작 태스크 식별자, 종료 태스크 식별자, 가능 조건, 종속성 형태로 구성된다. 식별자는 제어연결자 e를 유일하게 구별하게 한다. 시작 태스크 식별자와 종료 태스크 식별자는 제어 연결자 e가 연결하는 태스크의 식별자이다. 가능 조건(enable condition)은 제어 연결자에 부가된 조건식으로 워크플로우 데이터, 시간 데이터, 태스크 수행상태를 인수로 갖는 논리식이다. 종속성 형태는 순차 종속성이나 병행 종속성(즉, SS, PS, SC, PC, IE, SSPC, NE, PSSC, 그리고 PSPC) 중 하나이다. □

워크플로우 그래프에서 AND 및 OR 연결 형태는 선행 태스크와 후행 태스크간의 제어흐름을 의미한다. AND 연결 형태는 선행 태스크 수행 후 둘 이상의 태스크 수행을 요구하는 제어 흐름을 표현한다. AND 연결 형태와 달리 조건에 의해 여러 후행 태스크 중 한 태스크만의 수행을 요하는 제어흐름은 OR 연결 형태로 표현된다.

본 연구에서 제안한 다양한 병행 종속성은 워크플로우 그래프에서 노드들간에 레이블을 갖는 제어 연결자로 간단히 표현할 수 있다. 즉, 해당 종속성 형태를 제어 연결자의 레이블로 기술함으로써 워크플로우 그래프를 쉽게 확장할 수 있다. 또한, 종속성 표현에 있어 $St_{Ti} < St_{Tj}$(또는 $Cm_{Ti} < Cm_{Tj}$)와 $St_{Ti} > St_{Tj}$($Cm_{Ti} > Cm_{Tj}$)의 순서관계를 구분하기 위해 전자의 종속성은 T_i에서 T_j로 방향을 갖는 제어 연결자로 표현하며, 후자는 반대 방향의 제어 연결자로 표현한다. 정의된 워크플로우 그래프를 기반으로 간단한 워크플로우 명세는 (그림 1)과 같다. 순차 종속성은 간단한 표현을 위해 레이블이 없는 제어 연결자로 표현하였고, AND 및 OR 연결형태를 갖는 제어 연결자의 식별자 집합은 그래프의 판독을 쉽게 하기 위해 반호와 AND 및 OR 레이블로 표현하였다. 또한, 다양한 병행 종속성을 갖는 제어식별자는 구별하기 쉽게 점선으로 표현하였다. (그림 1)은 세 가지 병행 종속성 요구를 모델링 하였다. 첫째, 태스크 T_2와 T_8은 동시에 완료되어야함을 모델링 하였다. 둘째, 태스크 T_3와 T_4는 순차 시작하여 병행 수행되어야 함을 모델링 하였다. 셋째, 태스크 T_5는 T_9 수행 동안에 수행되어야 함을 모델링 하였다. 한편, T_5로 들어오는 제어 연결자 상에 AND는 T_3와 T_4가 모두 완료되었을 때 T_5가 수행될 수 있음을 간략히 표현하는 연결 형태이다.



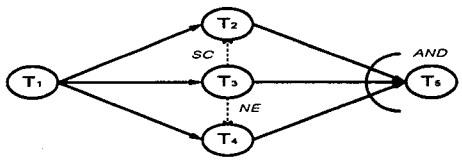
(그림 1) 다양한 병행 종속성을 포함한 워크플로우 그래프

[예제 1] 다양한 병행 종속성을 포함한 워크플로우 모델링을 보이기 위해 "여행 예약" 워크플로우를 가정한다. 도시에서 도시로의 여행에 필요한

교통편 및 숙박 시설의 예약을 수행하는 태스크들로 구성된다고 가정한다. 워크플로우가 여행 접수 태스크(T_1), 비행기표 예약 태스크(T_2), 호텔 예약 태스크(T_3), 자동차 대여 태스크(T_4), 그리고 예약 결과 알림 태스크(T_5)로 구성된다고 가정한다. 이 때 태스크들간에 다음과 같은 요구조건을 포함한 워크플로우 그래프는 (그림 2)와 같다.

<요구 조건>

"자동차 대여는 호텔 예약이 처리되는 동안에 성공적으로 처리되어야 하며 호텔 예약이 가능할 때만 비행기표를 예약해야 한다."



(그림 2) 여행 예약 워크플로우 그래프

그림에서 T_4와 T_3간의 종속성을 중첩 종속성(NE)으로 모델링 함으로써 호텔 예약 수행 중에 자동차 대여 수행을 보장할 수 있다. 또한 T_3과 T_2가 순차 완료 종속성(SC)으로 모델링 됨으로써 호텔 예약이 완료될 때만 비행기표 예약의 완료를 보장할 수 있다. □

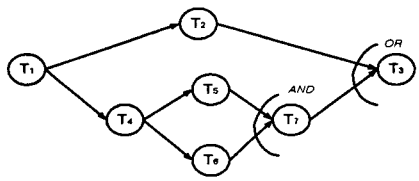
3. 워크플로우 명세의 일관성 검증

워크플로우 그래프로 기술된 워크플로우 명세는, 요구된 제약 조건을 태스크간의 종속성으로 정확하게 표현할 수 있다. 그러나, 다양한 종속성을 포함한 전체 워크플로우 명세가 논리적 모순이 없고 수행 가능함을 가정할 수 없다. 즉, 태스크간에 수행 불가능한 종속성 요구를 포함할 수는 있다. 본 장에서는 다양한 병행 종속성을 포함한 워크플로우 명세에 존재할 수 있는 모순을 검증한다. 이를 위해, 첫째 다양한 병행 종속성을 제외한 워크플로우 그래프에서 수행 가능 경로를 식별하기 위한 방안을 제안한다. 결정된 수행 가능경로를 기반으로 둘째, 워크플로우 명세에 존재할 수 있는 모순을 제시하고 이를 검증할 수 있는 일관성 검증 방안을 제안한다.

3.1 수행 가능 경로 결정

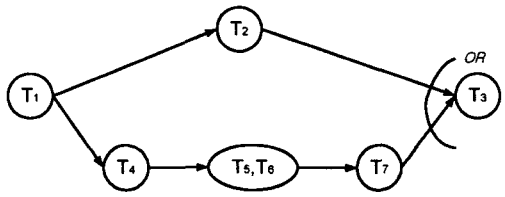
워크플로우 그래프에서 수행 가능 경로를 결정하기 위해, 먼저 다양한 병행 종속성을 제외한 워크플로우 그래프를 고려한다. 이는 다양한 병행 종속성은 동시에 수행되는 태스크간의 요구 조건을 명세할 뿐 수행 가능 경로를 결정함에 있어 의미가 없기 때문이다. 다양한 병행 종속성을 제외한 워크플로우 그래프의 특성은 N개의 태스크로 구성될 때 한 개의 시작 태스크와 한 개의 종료 태스크를 갖는 비 순환 그래프이다. N개의 태스크에 대해 수행될 수 있는 경로는 AND 및 OR의 연결형태에 의해 많은 경우의 수[2]가 존재한다.

AND와 OR의 연결 형태를 포함한 워크플로우 그래프에서 수행 가능 경로를 구하기 위해, AND 연결 형태에 연관된 선행 노드들을 단일 노드로 변환한다. 이는 AND 연결 형태를 갖는 태스크의 선행 태스크들은 병행 수행되는 태스크 집합으로써 동일한 수행 경로에 포함되기 때문이다. (그림 3)과 같이 T_7의 AND 연결 형태는 T_5와 T_6가 병행 수행되어야함을 의미한다. 즉, T_5와 T_6은 동일한 수행 경로에 포함되기 때문에 단일 노드로 변환하면 (그림 4)와 같다. (그림 4)에서의 수행 가능한 경로는 $<T_1,T_2,T_3>$ 또는 $<T_1,T_4,T_5,T_6,T_7,T_3>$가 된다.



(그림 3) AND와 OR 연결 형태를 갖는 워크플로우 그래프

2) 모든 태스크들이 OR 연결 형태로 구성되고 한 개씩의 시작 및 종료 노드를 갖는 비 순환 그래프에서 최대 경로 수는 $\sum_{i=2}^{N-1}(N-i)!$ 개이다.



(그림 4) AND 연결 형태를 제거한 워크플로우 그래프

워크플로우 그래프는 하나 이상의 AND 및 OR 연결 형태를 포함할 수 있다. 또한 각각의 연결 형태는 AND나 OR 연결형태를 내포할 수 있다. 내포된 구조를 구분하기 위해 오직 하나의 연결형태만으로 구성된 부분 그래프를 원자의(atomic) AND(또는 OR) 연결형태라 하고 AND(OR) 연결형태에서 AND(OR) 제어 흐름을 요구하는 노드를 AND(OR) 시작 노드라 하고 대응하는 노드를 AND(OR) 종결 노드라 한다. (그림 3)에는 T_4를 AND 시작 노드로, T_7은 AND 종결 노드로 갖는 원자의 AND 연결형태가 존재한다. 원자의 AND(OR) 연결형태를 기반으로 워크플로우 그래프는 여섯 가지 구조로 구분된다. 즉, 원자의 AND 연결형태 구조, 원자의 OR 연결형태 구조, AND 연결형태에 하나이상의 원자의 AND 연결형태를 내포한 구조, AND 연결 형태에 하나 이상의 원자의 OR 연결형태를 내포한 구조, OR 연결형태에 하나 이상의 원자의 AND 연결형태를 내포한 구조, 그리고 OR 연결 형태에 하나 이상의 원자의 OR 연결형태를 내포한 구조로 구분된다. 각 구조에서 AND 연결 형태의 제거는 다음 규칙에 의해 달성된다. 원자의 OR 연결형태 구조와 OR 연결형태에 하나 이상의 원자의 OR 연결형태를 내포한 구조는 AND 연결형태를 포함하지 않기 때문에 고려하지 않는다.

규칙 1. 원자의 AND 연결형태 구조
: AND 시작 노드와 AND 종결 노드사이에 있는 모든 노드를 하나의 노드로 합병한다.

규칙 2. AND 연결형태에 하나이상의 원자의 AND 연결형태가 내포된 구조
: 규칙 1을 원자의 AND 연결형태에 반복 적용함으로써 AND 연결 형태를 제거한다.

규칙 3. AND 연결형태에 하나 이상의 원자의 OR 연결형태가 내포된 구조

단계 1. AND 시작 노드와 AND 종결 노드사이에 있고 원자의 OR 연결형태에 포함되지 않는 노드들을 하나의 노드로 합병한다. 합병된 노드를 기본(default) 노드라 한다. 기본 노드를 시작 노드로 하고 AND 종결노드를 종료노드로 하는 제어 연결자를 구성한다.

단계 2. 원자의 OR 연결형태 각각에 대해, OR 시작 노드와 OR 종말 노드사이에 있는 노드들을 집합으로 변환한다. 이때 집합은 노드들의 태스크 식별자를 원소로 갖는다. 모든 집합들을 카테션 곱(cartesian product)하여 모든 조합을 결정한다. 결정된 조합 각각에 대해 노드를 생성한다.

단계 3. 모든 원자의 OR 연결형태에서 OR 시작(종말) 노드들을 각각 합병한다. 합병된 OR 시작노드를 종료 노드로, AND 시작 노드를 시작 노드로 하는 제어 연결자를 구성한다. 한편, 합병된 OR 종말노드는 시작노드로, 기본 노드는 종료노드로 하는 제어 연결자를 구성한다.

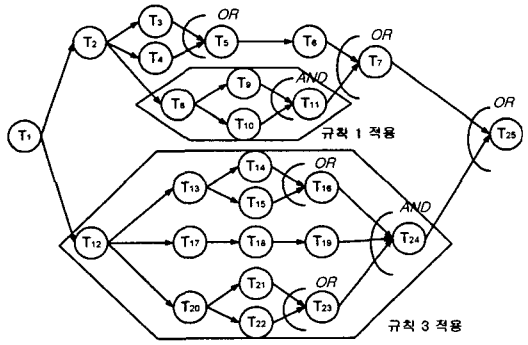
단계 4. 단계 2에서 생성된 노드 각각은 합병된 OR 시작 노드를 시작 노드로 하는 제어 연결자와 합병된 OR 종료노드를 종료노드로 하는 제어 연결자를 생성한다.

규칙 4. OR 연결형태에 하나 이상의 원자의 AND 연결형태가 내포된 구조
: 규칙 1의 적용에 의해 원자의 AND 연결형태는 존재하지 않는다.

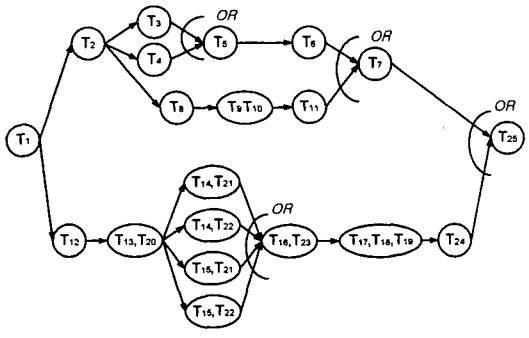
제시된 규칙들에 의해 변환된 워크플로우 그래

프에서 수행가능 경로를 결정한다. 수행 가능경로의 결정은 먼저, 워크플로우 시작 노드의 태스크 식별자를 인접한 노드들로 전달한다. 다음으로 인접한 노드는 전달된 태스크 식별자에 자신의 태스크 식별자를 추가하여 인접한 노드에 전달한다. 두 번째 과정을 워크플로우 종료 노드까지 반복함으로써 모든 수행 가능한 경로를 결정한다. 수행가능 경로 중 오직 하나의 경로에 속하는 태스크들만이 워크플로우 관리시스템에 의해 수행된다. 한편, 경로를 구성하는 태스크 식별자의 순서는 AND와 OR의 연결형태의 수행적 특성 때문에 고려하지 않는다.

[예제 2] 수행가능 경로의 결정 과정을 보이기 위해 (그림 5)의 워크플로우 그래프를 가정한다. 앞서 제시한 규칙들을 적용함으로써 (그림 6)의 워크플로우 그래프를 생성할 수 있다. 특히 규칙 3의 적용에 있어 단계 3의 적용은 다음과 같다. 먼저, OR



(그림 5) 워크플로우 그래프에서 AND 연결형태 제거 규칙 적용



(그림 6) AND 연결형태를 제거한 그래프

시작 및 종말 노드들을 각각 합병하여 {T_{13},T_{20}}과 {T_{16},T_{23}} 노드를 생성한다. 다음으로 {T_{13},T_{20}}을 T_{12}에 대해 후행 노드로, {T_{16},T_{23}}은 단계 1에 의해 결정된 노드 {T_{17},T_{18},T_{19}}의 선행 노드로 제어 연결자를 설정한다. 변환된 워크플로우 그래프에서의 인접한 노드들간에 태스크 식별자를 전달함으로써 결정된 수행 가능 경로는 다음과 같다.

<수행가능 경로>

$\langle T_1, T_2, T_3, T_5, T_6, T_7, T_{25}\rangle$
$\langle T_1, T_2, T_4, T_5, T_6, T_7, T_{25}\rangle$
$\langle T_1, T_2, T_8, T_9, T_{10}, T_{11}, T_7, T_{25}\rangle$
$\langle T_1, T_{12}, T_{13}, T_{20}, T_{14}, T_{21}, T_{16}, T_{23}, T_{17}, T_{18}, T_{19}, T_{24}, T_{25}\rangle$
$\langle T_1, T_{12}, T_{13}, T_{20}, T_{14}, T_{22}, T_{16}, T_{23}, T_{17}, T_{18}, T_{19}, T_{24}, T_{25}\rangle$
$\langle T_1, T_{12}, T_{13}, T_{20}, T_{15}, T_{21}, T_{16}, T_{23}, T_{17}, T_{18}, T_{19}, T_{24}, T_{25}\rangle$
$\langle T_1, T_{12}, T_{13}, T_{20}, T_{15}, T_{22}, T_{16}, T_{23}, T_{17}, T_{18}, T_{19}, T_{21}, T_{25}\rangle$

□

3.2 일관성 검증

앞서 수행가능 경로를 결정하기 위해 다양한 병행 종속성을 제외하였다. 이는 워크플로우 명세에 존재할 수 있는 모순을 검증하기 위한 과정이며, 본래 워크플로우 명세는 본 연구에서 제안하고 있는 다양한 병행 종속성을 포함한다. 즉, 워크플로우 명세는 (그림 1, 2)와 같이 다양한 병행 종속성을 포함한다. 다양한 병행 종속성 요소들로 구성된 전체 워크플로우는 일관성을 위배할 수 있다. 즉, 다양한 병행 종속성을 포함한 워크플로우 명세에서, 서로 다른 수행 경로에 포함된 태스크간에 종속성 관계를 내포하거나, 수행 경로를 구성하는 태스크간에 수행 불가능한 종속성 모순을 포함할 수 있다. 전자는, 동시에 수행될 수 없는 태스크들간에 종속성을 갖음으로써, 그들간에 강요할 수 없는 종속성을 갖는 논리적 모순이 존재한다. 후자는, 같은 수행 경로에 포함된 태스크들간에 서로 상이한 두 개 이상의 종속성을 갖거나, 자신에서 자신으로 종속성을 강요하는 논리적 모순이 존재한다. 본 연구에서는 먼저, 전자의 모순을

정의하고 이를 발견할 수 있는 방안을 제안한다. 다음으로, IE, SSPC, NE, PSSC, 그리고 PSPC 이외에 본 연구에서 제안한 SS, PS, SC, 그리고 PC를 고려한 모순 검증 방안을 제안한다.

3.2.1 경로간 종속 모순

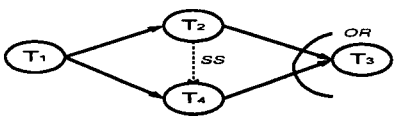
서로 다른 경로에 포함된 태스크에 대해, 그들 사이에 존재하는 병행 종속성은 강요할 수 없는 종속성이다. 이는, 워크플로우 그래프에서 발견될 수 있는 모든 경로들 중 워크플로우 관리 시스템에 의해 수행되는 수행 경로는 오직 하나이기 때문이다. 즉, 수행 경로 상에 있는 태스크는 수행될 수 없는 태스크의 시작 및 완료를 기다리는 모순이 발생한다. 이러한 모순을 다음과 같이 경로간 종속 모순이라 정의한다.

정의 2. 경로간 종속 모순

수행가능 경로 집합 $PS=\{ps_1, ps_2, \cdots, ps_n\}$과 수행가능한 경로 ps_i에 대해, 다음을 만족하는 태스크 t_i에서 $t_j(t_i \neq t_j)$로의 병행 종속성이 존재하면 이를 경로간 종속 모순이라 한다.

- $t_i \in ps_i$이고 $t_j \notin ps_i$ □

경로간 종속 모순이 존재하는 간단한 예로는 그림 7을 고려할 수 있다. 이때 수행가능 경로 결정 규칙에 의해 산출된 경로는 $PS=\{\{T_1, T_2, T_3\}, \{T_1, T_4, T_3\}\}$이다. 이때 $T_2 \in \{T_1, T_2, T_3\}$이고 $T_4 \notin \{T_1, T_2, T_3\}$인 T_2와 T_4간에 병행 종속성 SS가 존재한다. 즉, 경로간 종속 모순이 존재한다.



〈그림 7〉 경로간 종속 모순의 예

제시한 경로간 종속 모순을 워크플로우 그래프에서 보이면 정리 1과 같고, 이를 발견하기 위한 알고리즘은 다음과 같다.

정리 2. 워크플로우 그래프에서 경로간 종속 모순

워크플로우 그래프 $WFG=(V, E)$에서, V에 속하는 임의의 노드 t_i와 $t_j(t_i \neq t_j)$가 병행 종속성을 갖고 t_i와 t_j가 동일 수행가능 경로에 포함되지 않으면 WFG는 모순이 존재한다.

증명. 증명을 위해, V에 속하는 임의의 노드 t_i와 t_j가 병행 종속성을 갖고 t_i와 t_j가 동일 수행가능 경로에 포함되지 않으면 WFG는 모순이 존재하지 않는다고 가정한다. 먼저, 병행 종속성을 제외한 워크플로우 그래프에 대한 수행가능 경로 결정 규칙을 적용하여 경로를 생성한다. 일반성에 위배됨이 없이 경로 ps_m과 ps_n이 생성되고 $t_i \in ps_m$과 $t_j \in ps_n$를 만족한다고 가정하자. 경로는 AND 연결 형태를 갖지 않는 OR 연결형태를 기반으로 생성되었기 때문에 하나의 경로만 수행된다. 이때 $t_i(t_j)$에서 $t_j(t_i)$로의 종속성 요구는 그들이 병행수행 될 때 강요될 수 있는 것으로 $t_i(t_j)$는 수행할 수 없는 $t_j(t_i)$의 수행을 무한히 기다리는 교착상태(deadlock)가 발생한다. 즉, 가정에 거짓이므로 WFG에 t_i와 $t_j(t_i \neq t_j)$가 병행 종속성을 갖고 t_i와 t_j가 동일 수행가능 경로에 포함되지 않으면 WFG는 모순이 존재한다. □

<경로간 종속 모순 발견 알고리즘>

[단계 1] 병행 종속성 제거

워크플로우 그래프에서 병행 종속성을 제외한다. 제외된 종속성 정보를 병행 종속성 정보로 유지한다.

[단계 2] AND 연결 형태의 제거

워크플로우 그래프에서 AND 연결 형태를 제거하기 위해 규칙 1~4를 적용한다.

[단계 3] 수행 가능 경로를 결정

먼저, 워크플로우 시작 노드의 태스크 식별자를 인접한 노드들로 전달한다. 다음으로 인접한 노드는 전달된 태스크 식별자에 자신의 태스크 식별자를 추가하여 인접한 노드에 전달한다. 두 번째 과정을

워크플로우 종료 노드까지 반복함으로써 모든 수행 가능한 경로를 결정한다.

[단계 4] 경로간 종속 모순 발견

다음을 병행 종속성 각각에 대해 수행한다.

If 병행 종속성을 갖는 두 태스크가 서로 다른 경로에 포함

Then 경로간 종속 모순

3.2.2 경로 내에서 자가 종속 모순 및 다중 종속 모순

수행가능 경로 내의 태스크들간에 존재할 수 있는 모순을 검증하기 위해 먼저, SS, PS, SC, 그리고 PC의 이행적 특성을 분석한다. 이는, 하나의 태스크를 매개로 이행적 조합에 의해 새로운 태스크간 종속성을 산출할 수 있기 때문이다. 네 가지 종속성에 대해, 각각의 조합을 표현하면 표 3과 같다. T_i와 T_j간에 동시 시작 종속성을 $PS(T_i,T_j)$로 표현한다고 가정하면, 표에서 $PS(T_i,T_j) \wedge SS(T_j,T_k) \rightarrow SS(T_i,T_k)$와 같은 이행적 특성을 표현하고 있다.

〈표 3〉 태스크간 종속성 요소의 조합

∧		(T_j,T_k)			
		SS	PS	SC	PC
(T_i,T_j)	SS	SS	SS	*	*
	PS	SS	PS	*	*
	SC	*	*	SC	SC
	PC	*	*	SC	PC

* 는 이행적 특성이 존재하지 않는다.

분석된 특성을 기반으로 IE, SSPC, NE, PSSC, 그리고 PSPC 종속성을 포함한 워크플로우 명세에 대한 자가 종속 모순 및 다중 종속 모순을 발견할 수 있다. 먼저 자가 종속 모순의 예를 보이기 위해, 수행가능 경로 내에 존재하는 태스크들이 다음과 같은 병행 종속성을 갖는다고 가정한다.

$SS(T_i, T_j) \wedge SSPC(\underline{T_j, T_k}) \wedge SS(T_k, T_i)$ (1)

$SS(T_i, T_j) \wedge \underline{SS(T_j, T_k) \wedge PC(T_j, T_k)} \wedge SS(T_k, T_i)$ (2)

$\underline{SS(T_i, T_j) \wedge SS(T_j, T_k) \wedge SS(T_k, T_i)} \wedge PC(T_j, T_k)$ (3)

$SS(T_i, T_k) \wedge SS(T_k, T_i) \wedge PC(T_j, T_k)$ (4)

$\boldsymbol{SS(T_i, T_i)} \wedge PC(T_j, T_k)$ (5)

(1)과 같은 워크플로우 명세는 정리 1에 의해 (2)와 같이 변환되고, (3)과 (4)에 대해 표 3의 이행성을 적용함으로써 (5)와 같은 모순된 종속성 명세를 발견할 수 있다. 즉, T_i 시작 후 T_i를 시작해야 하는 자가 종속 모순이 발견된다.

다음으로, 다중 종속 모순의 예로써, 다음과 같은 워크플로우 기술을 가정하자.

$IE(T_i, T_j) \wedge PSSC(T_j, T_k) \wedge PS(T_i, T_k)$ (1)

$SS(T_i, T_j) \wedge SC(T_i, T_j) \wedge PS(T_j, T_k) \wedge SC(T_j, T_k) \wedge PS(T_i, T_k)$ (2)

$SS(T_i, T_j) \wedge PS(T_j, T_k) \wedge SC(T_i, T_j) \wedge SC(T_j, T_k) \wedge PS(T_i, T_k)$ (3)

$\boldsymbol{SS(T_i, T_k)} \wedge SC(T_i, T_k) \wedge \boldsymbol{PS(T_i, T_k)}$ (4)

(1)과 같은 워크플로우 명세는 정리 1에 의해 (2)와 같이 변환되고, (3)에 대해 <표 3>의 이행성을 적용하면 (4)와 같은 모순된 종속성 명세를 발견할 수 있다. 즉, T_i와 T_k가 순차시작(SS)과 동시시작(PS)의 서로 다른 종속성을 갖음으로써 다중 종속 모순이 존재한다.

다양한 병행 종속성 형태를 포함한 수행 가능 경로에 대한 자가 종속 및 다중 종속 모순은 다음과 같다.

정리 3. 수행가능 경로에서 자가 종속 모순

워크플로우 그래프 *WFG*에서 수행가능 경로 집합 $PS = \{ps_1, ps_2, \cdots, ps_n\}$과 수행 가능한 경로 ps_i에서 $t_j \in ps_i$인 노드 t_j에 대해, t_j에서 t_j로의 종속성이 존재하면 워크플로우 그래프에는 모순이 존재한다.

증명. 증명을 위해, 임의의 노드 t_j에서 t_j로 가는 순환이 존재하면, 워크플로우 그래프에는 모순이

존재하지 않는다고 가정한다. 일반성에 위배됨이 없이 t_j에서 t_j로 경로 $p=t_j, t_{j+1}, t_{j+2}, \cdots, t_k, t_j$이 존재하고 경로에 존재하는 모든 노드가 수행가능 경로 ps_i에 속한다고 가정하자. 이때 [정희택 & 이도헌 1998]에서 제안된 종속성을 정리 1에 의해 SS, PS, SC, PC로 변환하고 표3의 조합에 의해 종속성 S를 구할 수 있다. 이는 한 태스크의 시작과 종료가 자신의 시작과 종료에 각각 종속됨을 의미하는 것으로 수행될 수 없는 워크플로우의 정의이다. 즉, 가정이 거짓이므로 *WFG*에 순환이 존재하면 *WFG*에는 모순이 존재한다. □

정리 4. 수행가능 경로에서 다중 종속 모순

워크플로우 그래프 *WFG*에서 수행가능 경로 집합 $PS=\{ps_1, ps_2, \cdots, ps_n\}$과 수행 가능한 경로 ps_i에서 $t_j \in ps_i$이고 $t_k \in ps_i$인 노드 t_j, t_k에 대해, t_j에서 t_k로의 둘 이상의 종속성 존재하면 워크플로우 그래프에는 모순이 존재한다.

증명. 증명을 위해, 임의의 노드 t_j에서 t_k로의 둘 이상의 종속성이 존재하면 WG는 모순이 존재하지 않는다고 가정한다. 일반성에 위배됨이 없이 $t_{j+1} \neq t'_{j+1}, t_{j+2} \neq t'_{j+2}, \cdots, t_{k-1} \neq t'_{k-1}$에 대해 노드 t_j에서 t_k로의 경로 $P=t_j, t_{j+1}, t_{j+2}, \cdots, t_{k-1}, t_k$ 및 $P'=t_j, t'_{j+1}, t'_{j+2}, \cdots, t'_{k-1}, t_k$이 존재하고 P, P'에 속하는 모든 노드가 ps_i에 속한다고 가정하자[3]. 이때 [정희택 & 이도헌 1998]에서 제안된 종속성을 정리 1에 의해 SS, PS, SC, PC로 변환하고 표3의 조합에 의해 종속성 S(경로 P에 대한)와 S'(경로 P'에 대한)을 구할 수 있다. 이는 태스크 t_j와 t_k가 종속성 S 및 S'를 갖음을 의미한다. S ≠ S'이면 상이한 종속성 S와 S'를 t_j와 t_k간에 강요해야 한다. 즉, S는 태스크 t_j의 시작(또는 종료) ***후에*** 태스크 t_k의 시작(또는 종료)을 강요한다면, S'는 태스크 v_i의 시작(또는 종료) ***전에*** 태스크 v_j의 시작(또는 종료)을 강요할 수 있다. 이러한 이유로 기술된 워크플로우는 수행될 수 없다. 가정이 거짓이므로, 병렬 경로가 존재하고 경로의 종속성 조합 결과가 서로 상이하면 WG에는 모순이 존재한다. □

3) 증명을 간략히 하기 위해 두 개의 경로만을 가정한다. 셋 이상의 경로도 유사하게 증명될 수 있다.

<자가 종속 모순 발견 알고리즘>

[단계 1] 종속성 변환

각 수행 가능 경로에서 [정희택 & 이도헌 1998]에서 제안된 종속성 형태를 정리 1에 의해 본 연구에서 제안된 종속성 형태(즉, SS, PS, SC, PC)로 변환한다.

[단계 2] 자가 종속 모순 발견

선행 태스크와 후행 태스크간의 병행 종속성에 대해 다음을 적용한다.

- 후행 태스크와 종속성 관계를 갖는 새로운 후행 태스크를 반복 결정한다.
- 선행 태스크 및 후행 태스크들의 순서화된 집합을 이행적 태스크 집합으로 정의한다.
- If 각 이행적 태스크 집합에 동일한 태스크 식별자가 둘 이상 존재
  Then 자가 종속 모순

< 다중 종속 모순 발견 알고리즘>

[단계 1] 종속성 변환

각 수행 가능 경로에서 [정희택 & 이도헌 1998]에서 제안된 종속성 형태를 정리 1에 의해 본 연구에서 제안된 종속성 형태(즉, SS, PS, SC, PC)로 변환한다.

[단계 2] 이행적 태스크 집합 결정

선행 태스크와 후행 태스크간의 병행 종속성에 대해 다음을 적용한다.

- 후행 태스크와 종속성 관계를 갖는 새로운 후행 태스크를 반복 결정한다.
- 선행 태스크 및 후행 태스크들의 순서화된 집합을 이행적 태스크 집합으로 정의한다.

각 이행적 태스크 집합에 대해 <표 3>을 적용함으로써 시작 태스크와 종료 태스크간의 종속성을 결정한다.

[단계 3] 다중 종속 모순 발견

동일한 시작 및 종료 태스크를 갖는 이행적 태스크 집합에 대해 다음을 적용한다.

If 시작 및 종료 태스크간의 종속성이 서로 상이

Then 다중 종속 모순

4. 결 론

본 연구는 다양한 병행 종속성을 포함한 워크플로우 모델링 및 검증방안을 제안하였다. 먼저, 병행 수행되는 태스크들간의 종속성으로 네 가지 종속성을 제안하였다. 제안된 종속성에는 순차 시작, 병행 시작, 순차 완료, 그리고 병행 완료 종속성으로 구분하였다. 둘째, 워크플로우 그래프에서 다양한 병행 종속성을 기술할 수 있는 방안을 제안하였다. 셋째, 다양한 병행 종속성이 포함된 워크플로우 그래프에 존재할 수 있는 모순을 정의하고 이를 검증할 수 있는 방안을 제안하였다. 이를 위해 먼저, 워크플로우 그래프에서 수행가능 경로를 결정할 수 있는 방안을 제안하였다. 다음으로 수행가능 경로간에 종속성이 존재함으로써 발생하는 경로간 종속 모순을 정의하였고 이를 검증하기 위한 방안을 제안하였다. 넷째, 기존에 제안된 종속성(즉, IE, SSPC, NE, PSSC, 그리고 PSPC)과 본 연구에서 제안된 종속성으로 구성된 워크플로우 명세에서 자가 종속 및 다중 종속 모순을 제시하고 이를 검증할 수 있는 방안을 기술하였다.

본 연구는 병행 수행하는 태스크간의 다양한 종속성 요구를 모델링 할 수 있다. 이는, 병행 수행되는 태스크간이라도 그들간에 필요한 다양한 제어흐름을 모델링 할 수 있음을 의미한다. 또한, 제안된 종속성 요소를 레이블을 갖는 제어 연결자로 모델링 함으로써, 워크플로우 그래프에 쉽게 모델링 할 수 있다. 다음으로 AND와 OR의 제어흐름을 포함하고 있는 워크플로우 그래프에서 수행가능 경로를 파악함으로써 검증 수단이외에 워크플로우 수행환경의 변화에 적절히 반응할 수 있다. 즉, 파악된 경로를 기반으로, 앞으로 수행될 태스크가 수행 불가능할 때, 태스크들의 불필요한 수행 없이 현 워크플로우 수행을 종료할 수 있다. 마지막으로, 다양한 병행 종속성을 포함한 워크플로우 명세에 대한 일관성 검증 방안을 제안하였다. 즉, 제안된 일관성 검증 알고리즘에 의해 워크플로우에 존재할 수 있는 모순을 발견할 수 있다.

향후 연구에서 병행 종속성을 포함한 워크플로우의 스케줄링 기법을 개발하고자 한다.

참 고 문 헌

[1] 정희택, 이도헌, "시간제약을 포함한 워크플로우 모델링," 「한국정보과학회 춘계학술 대회」, 25권, 1호, pp.101-103, April 1998.

[2] Elmagarmid A. and W. Du, 'Workflow Management : State of the Art vs. State of the Market,' *Advances in Wrokflow Management Systems and Interoperability*, A. Doğaç(Ed.), NATO, pp.1-17, 1997.

[3] Georgakopoulos D. and M. F. Hornick, "A Framework for Enforceable Specification of Extended Transactional Models and Transactional Workflow," *International Journal of Intelligent and Cooperative Information Systems*, Vol.3, No.3, pp.225-253, 1994.

[4] Georgakopoulos, D., M. F. Hornick and F. Manola, "Customizing Transaction Models and Mechanisms in a Programmable Environment Supporting Reliable Workflow Automation," *IEEE transaction on knowledge and data engineering*, Vol.8, No.4, pp.630-649, 1996.

[5] Hollingsworth, D. "Workflow Management Coalition-The Workflow Reference Model," *TC00-1003 issue 1.1*, 1994

[6] Leymann F. and D. Roller, "Workflow-based applications," *IBM system journal*, Vol.46, No.1, pp102-123, 1997.

[7] Davulcu, H. Michael Kifer, C.R. Ramakrishnan and I.V. Ramakrishnan, "Logic Based Modeling and Analysis of Workflows," *Proc. of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp.25-33, June 1998.

[8] Klein, K. "Advanced Rule Driven Transaction Management," *Proc. of the IEEE COMPCON*, pp.562-567, 1991.

[9] Rusinkiewicz, M. E. A. K. Elmagarmid, Y. Leu and W.Litwin, "Extending the Transaction Model to Capture more Meaning," *SIGMOD RECORD*, Vol.19, No.1, pp.3-7, 1990.

[10] Rusinkiewicz M. and A. Sheth, "Specification and Execution of Transactional Workflow," *Modern Database Systems : The Object, Interoperability and Beyond*, W. Kim(Ed.), Addison- Wesley, 1994.

[11] Adam, N.R. Vijayalakshmi Atluri and Wei-Kuang Huang, "Modeling and Analysis of workflows Using Petri Nets," *Journal of Intelligent Information Systems*, Vol.10, No. 2, pp.1-29, March 1998.

[12] Tatbul, N. S. Arpinar, P. Karagoz, I. Cingil, E. Gokkoca, M. Altinel, P. Koksal, and A. Dogac," A Workflow Specification Language and its Scheduler," *Proc. of 11th Symposium on Computer and Information Systems*, 1997.

[13] Attie, P. C. M. P. Singh, A. Sheth and M. Rusinkiewicz, "Specifying and Enforcing Intertask Dependency," *VLDB conference*, pp.134-145, 1993.

[14] Chrysanthis P. K. and K. Ramamritham, "ACTA: The SAGA Continues," *Chapter 10 in Database Transaction Model for Advanced Applications*, A. K. Elmagarmid(Ed.), 1992.

[15] Chrysanthis P. K. and K. Ramamritham, "Synthesis of Extended Transaction Models Using ACTA," *ACM transaction on Database Systems*, Vol.19, No.3, pp.450-491, 1994.

[16] Work Groupl, "Interface 1 : Process Definition Interchange Process Model," *Workflow Management Coalition Specification, TC-1016*, 1998.

[17] Van der Aalst," W.M.P. Modelling and Analysis of production Systems Using a Petri Net Based Approach," *Proc. of Conf. on Computer Integrated Manufacturing in the Process Industries*, pp.179-193, USA, 1994.

[18] Van der Aalst," W.M.P. Three Good Reasons for Using a Petri-net-based Work flow Management System," *Proc. of th Int. Working Conf. on Information and Process Integration in Enterprises*, pp.179-201, Nov 1996.

[19] Van der Aalst, W.M.P. "Petri-net-based Workflow Management Software," *Proc. of the NFS workshop on Workflow and Process Automation In Information System*, pp. 114-118, May 1996.

[20] Van der Aalst, W.M.P. "The Application of Petri Nets to Workflow Management," *The Journal of Circuits, Systems and Computers*, 1998.

■ 저자소개

정 희 택

전남대학교 전산통계학과를 졸업하고, 전남대학교 전산학과에서 이학석사학위를 취득하였다. 현재 전남대학교 전산학과 박사과정중에 있으며, 주 관심분야는 워크플로우 시스템, 데이터 웨어하우징, 데이터 마이닝 분야이다.

이 도 헌

한국과학기술원 전산학과를 졸업하고, 이학석사, 그리고 한국과학기술원 전산학과에서 이학박사학위를 취득하였다. 현재, 전남대학교 전산학과 조교수로 재직하고 있으며, 주 관심분야는 데이터 마이닝, 워크플로우 시스템, 데이터 웨어하우징, 퍼지 데이터베이스 분야이다.