

도달성 그래프를 이용한 SDL 표현 프로토콜 분석 기법

正會員 김 환 철*

An Analysis Scheme for Protocols Specified in SDL Using Reachability Graph

Hwan Chul Kim* *Regular Member*

요 약

SDL은 구조적 개념, 설계의 최적화, 구현의 유연성 및 설계자 상호간의 이해 증진을 위한 기법을 제공함으로써 통신 시스템의 동작을 기능적으로 명확하게 표현할 수 있으며, 정형적 표현 기법으로 인하여 설계 단계에서의 시스템 분석 방법을 제공한다. 그러나, 일반적으로 통신 시스템의 크고 복잡한 특성 때문에 시스템 동작에 대한 이해 증진과 시스템의 생존성(liveness) 및 도달성(reachability)과 같은 동작 특성을 분석하기에는 많은 어려움이 따른다. 따라서, SDL로 표현한 통신 시스템의 동작 특성을 효과적으로 분석하기 위해서는 이를 실증할 수 있는 분석 기법이 필요하다. 본 논문에서는 통신 프로토콜의 동작 특성을 분석하기 위하여 SDL의 페트리 넷 모델링에서 고려하여야 할 모델링 개념과, SDL의 수치 페트리 넷으로의 변환 법칙을 소개하고, 페트리 넷의 도달성 나무를 개선한 도달성 그래프를 이용하여 SDL로 명세한 시스템의 분석 기법에 대하여 기술한다.

ABSTRACT

SDL has been standardized to specify behavioral aspects of communication systems based on the formal description technique, and it is powerful and user friendly in the sense of supporting human communication and understanding, formal analysis and comparison of behaviors, alternative implementations and design optimization, and its structural decomposition. However, SDL is not sufficient for an efficient handling of entire system descriptions because the communication systems are generally very complex, and composed from the various interactions among sub-systems. Also, it is very difficult to explicitly verify dynamic views such as liveness and reachability. It leads the demands on analysis scheme to verify dynamic behaviors of specified systems. This paper presents modeling concepts of Petri Nets from SDL and transformation rules to Numerical Petri Nets to provide efficient techniques for verification of dynamic behaviors, and proposes the reachability graph that is able to trace all reachable states of a modeled system and reduce an information loss on the reachability tree.

*한국전자통신연구원
論文番號: 96231-0805
接受日字: 1996年 8月 5日

I. 서론

전전자 교환기를 비롯한 다양한 통신 시스템들은 다수의 인터페이스에 의한 병렬 수행과 외부로부터 발생하는 입력에 대한 응답의 형태로 프로세스의 논리적 기능을 순차적으로 수행하는 특성을 가진다. 이러한 통신 시스템의 특성은 일반적으로 복잡하고 과부하 트래픽을 유발하며, 병렬 수행에 대한 실시간 처리가 필요하게 되었지만, 오히려 응용 분야에서는 최소한의 오동작과 동작의 연속성을 요구하고 있다. 통신용 대규모 소프트웨어를 개발하는데 있어서 Stored Program Control(SPC) 기법을 이용함에 따라 소프트웨어의 생산성, 신뢰성, 및 유지 문제에 효과적으로 대처하기 위하여 국제 전신전화 자문 위원회(CCITT; 1993년부터 ITU-T)는 1960년대 후반부터 통신 시스템을 위한 규격/명세 언어(Specification and Description Language; 이하 SDL)를 연구한 바 있으며, 1976년부터 매 4년마다 표준안을 권고하고 있다[4, 9].

SDL은 구조적 개념, 설계의 최적화, 구현의 유연성, 및 설계자 상호간의 이해 증진을 위한 기법을 제공함으로써 통신 시스템의 동작을 기능적으로 명확하게 표현할 수 있으며, 정형적 표현 기법으로 인하여 설계 단계에서의 시스템 분석 방법을 제공하고 있다. 따라서, SDL을 이용한 시스템의 설계 및 분석과, 관련 언어들간의 상호 변환 및 기술 자료의 관리를 위한 도구들의 집합으로 전전자 교환기와 같은 통신 시스템의 개발 환경을 구축하는 연구가 활발히 이루어져 왔다[3, 5, 7]. 확장 유한 상태 기계(Extended Finite State Machine; EFSM) 개념을 기초로 한 SDL은 통신 시스템의 동작을 명확히 나타낼 수 있지만, 통신 시스템의 크고 복잡한 특성 때문에 시스템 동작에 대한 이해 증진과 시스템의 생존성(liveness) 및 도달성(reachability)과 같은 동작 특성을 분석하기에는 많은 어려움이 따른다. 따라서, SDL로 표현한 통신 시스템의 동작 특성을 효과적으로 분석하기 위해서는 이를 실증할 수 있는 분석 기법이 필요하다. 이러한 기법으로는 유사 대비(quasi parallelism), 페트리 넷(Petri Nets), 도달성 그래프(Reachability Graph), 비동기 시스템 상태 그래프(Asynchronous System State Graph), 비동기 통신 나무(Asynchronous Communication Tree), 나무 및 표를 이용한 혼합 표기(Tree

and Tabular Combined Notation), ESTELLE_R 및 NESDEL등이 있다[6, 8, 10].

페트리 넷트는 비동기적이고 동시 발생적인 시스템의 묘사와 분석을 가능하게 하는 모델링 도구로써 시스템의 상태 천이를 나무(tree) 형태로 나타내는 도달성 나무(reachability tree)와 행렬 방정식(matrix equation)을 이용한 분석 기법을 제공한다. 그러나, 도달성 나무에 의한 분석 기법은 무한 입력으로 인하여 시스템 상태의 도달성과 생존성 및 상태 천이 순서를 정확하게 분석하지 못하는 단점이 있으며, 행렬 방정식에 의한 분석 기법은 페트리 넷트의 구조 및 상태 천이 순서와 같은 정보를 충분히 표현할 수 없는 단점이 있다[2].

본 논문에서는 통신 프로토콜의 동작 특성 분석을 위하여 페트리 넷트의 도달성 나무를 개선한 도달성 그래프를 이용한 SDL 표현 프로토콜의 분석 기법에 대하여 기술한다. 본 논문의 구성은 다음과 같다. II 장에서는 SPC 기법을 이용한 통신 시스템의 표현을 위하여 ITU-T에서 권고한 SDL을 소개한다. III 장에서는 SDL의 페트리 넷트 모델링에서 고려하여야 할 모델링 개념과, SDL의 특성을 효과적으로 모델링하기 위하여 페트리 넷트를 일반화한 수치 페트리 넷트(NPN; Numerical Petri Nets)를 소개하고, SDL의 페트리 넷트 변환 법칙을 설명한다. IV 장에서는 도달성 나무에 의한 분석 기법을 개선한 도달성 그래프 알고리즘을 제시하고, 도달성 그래프를 이용하여 분석 가능한 동작 특성 및 분석 기법에 대하여 기술한다. 마지막으로 V 장에서는 결론과 향후 연구 과제에 대하여 언급한다.

II. SDL

SPC 기법을 이용한 통신 시스템은 일반적으로 크기가 대규모이고 복잡하며, 높은 신뢰성을 요구하고, 하드웨어에 고장이 발생하거나 오동작 시켰을 때 고도의 오류 복원이 필요하다. 또한, 생명 기간이 긴 특성으로 인하여 비정형적인 표현 방법으로는 시스템의 명세와 묘사가 어려운 문제점을 가지고 있다. 이러한 문제점들을 해결하기 위하여 국제 전신전화 자문 위원회(CCITT; 1993년부터 ITU-T)에서는 1968년 이후 관련 연구를 시작하여 SPC 시스템을 위한 프

로세스들의 기능을 명세하는 데 있어서 표준화된 방법의 필요성을 공표하였다. 1976년에는 통신 시스템의 내부 기능에 대한 정형화된 표현과 분석 기법, 시스템 구현의 유연성, 설계 최적화, 및 구조적 개념의 도입이 가능한 SDL을 권고하였다. 1976년 권고안은 통신 시스템의 순차적 동작 및 동시 수행을 표현하기 위한 기본 개념에 중점을 두었고, 1984년과 1988년 권고안은 시스템의 계층적 구성을 위한 표현 기법을 제시하였으며, 1992년 권고안에서는 소프트웨어 재사용에 중점을 두고 객체 지향적 개념을 위한 기법을 제공하였다[4, 9].

SDL의 구조적 개념은 시스템을 다수의 모듈 단위로 나누어 시스템의 묘사와 이해에 도움을 주기 위한 것이다. SDL의 구조적 개념은 시스템의 각 모듈을 분리하여 다룰 수 있고, 시스템의 실질적인 구조를 나타낼 수 있으며, 단단계 계층으로 시스템을 추론할 수 있어 시스템 구조를 효과적으로 작성할 수 있는 기반을 제공한다. SDL에서 시스템 구조는 블럭과 채널 및 프로세스들로 구성된다. 설계 초기 단계에서의 시스템은 다수의 블럭과, 블럭들의 사이 혹은 블럭과 시스템의 경계를 연결하는 다수의 채널들로 구성되어 있다. 블럭은 시스템의 계층적 구조를 지원하며, 채널은 블럭과 블럭 사이 혹은 블럭과 시스템 경계 사이에서 시그널 전송을 위한 통로로 사용된다. 프로세스는 일련의 정보를 요구하는 논리 기능의 기본 단위가 된다. 복잡한 대규모 시스템을 효과적으로 다루기 위한 SDL에서의 구조적 개념은 다음과 같으며, 그림 1과 2는 SDL의 구조적 개념에 따라 명세한 교

환 시스템의 예를 보인 것이다.

- 시스템은 다수의 블럭과 이들 블럭을 연결하는 채널로 구성된다.
- 각 블럭 혹은 채널은 하위 블럭과 하위 채널을 포함하는 하부 구조를 가질 수 있다.
- 최하위 블럭은 논리 기능을 정의하는 다수의 프로세스를 가진다.
- 시스템의 실질적인 기능은 프로세스들의 논리 기능에 의하여 이루어진다.

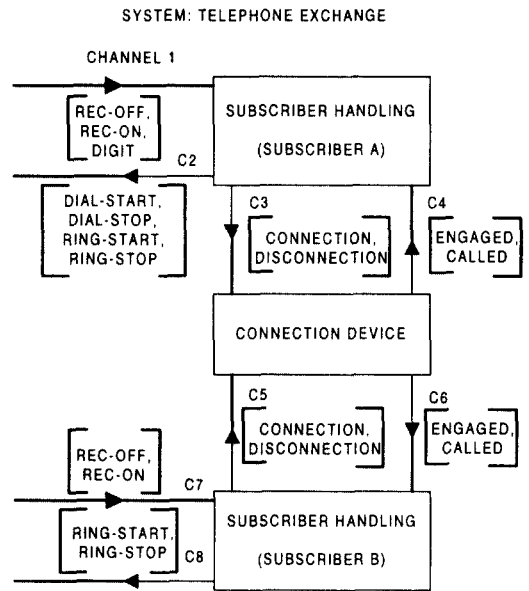


그림 1. Telephone Exchange를 SDL의 구조적 개념으로 명세한 예

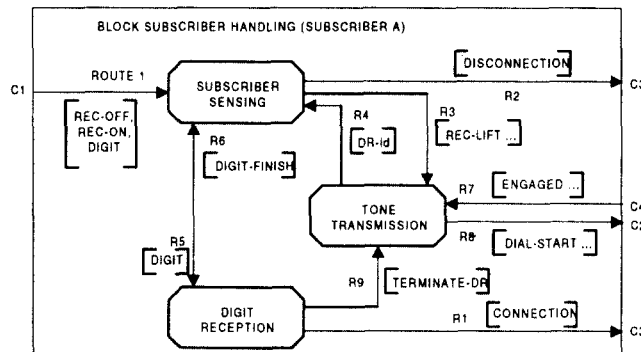


그림 2. Subscriber Handling 블럭을 SDL의 구조적 개념으로 명세한 예

- 각 프로세스는 시그널로 불리는 메시지에 의하여 다른 프로세스와 비동기적이고 동시 수행으로 동작한다.
- 채널은 시그널의 유효 경로로써 시그널의 전송을 위한 통로이다.

- 시작/종료는 프로세스 동작의 시작과 종료를 나타내며, 시작 심볼 다음에는 천이 심볼이 나타난다.
- 상태 심볼은 원하는 입력 시그널이 전송될 때까지 프로세스의 동작이 중지됨을 나타내며, 이는 프로세스의 제어 흐름이 지정된 상태에 있음을 뜻한다.
- 입/출력 심볼은 상태 천이를 위한 자극/반응 시그널을 명시하며 시그널은 로컬 변수를 가질 수 있다.
- 저장 심볼은 프로세스가 임의의 상태에서 천이를 유발하는 신호가 아닌 다른 신호가 도착했을 때, 다른 상태 천이를 위하여 해당 시그널을 저장한다.
- 타스크는 천이 과정에서 데이터 값의 배정등을 위하여 사용하며, 시간에 관련된 동작 표현에 사용한다.
- 결정은 천이 과정에서 데이터 값의 질의에 따른 천이 경로를 선택한다.
- 프로시듀어 호출은 천이 과정의 동작으로 오로지 하나의 입/출구 경로를 가진다.
- 프로세스 생성 호출은 주어진 초기 파라미터 값에 따라 프로세스 인스턴스를 생성한다.

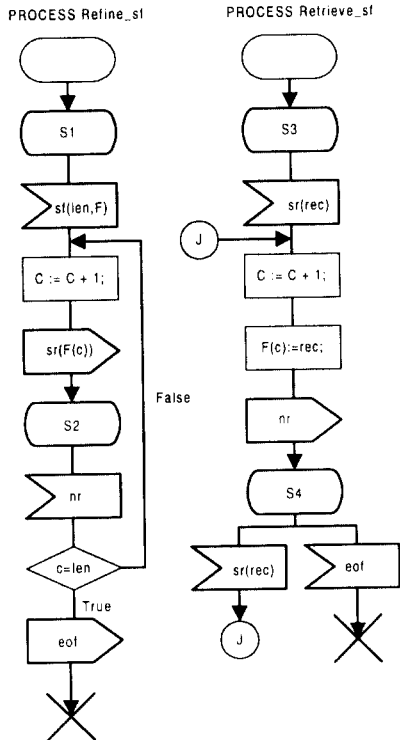


그림 3. 시그널 변환의 논리 기능을 SDL로 명세한 예

SDL에서의 프로세스는 자극/반응의 형태로써 프로세스의 논리적 기능을 구분하여 기술하며, 이러한 접근 방법은 EFSM에 기초를 두고 있다. 프로세스의 논리 기능은 시작(start), 종료(stop), 입/출력 시그널(input/output), 저장(save), 상태(state), 결정(decision), 타스크(task), 프로시듀어 호출(procedure call), 프로세스 생성 요구(create request), 및 천이(transition) 심볼을 이용하여 명세하며, 각 상태는 일련의 천이를 거쳐 다른 상태에 도달하게 된다. 그림 3은 시그널 변환의 실질적인 논리 기능을 아래의 SDL 개념을 이용하여 명세한 예이다.

III. SDL의 페트리 넷 모델링

1. SDL의 페트리 넷 모델링 개념

컴퓨터/통신 시스템이 발달함에 따라 시스템의 동시 발생적 요소들 사이의 정보 전달에 대한 복잡성 때문에 시스템의 이해와 설계 및 구현 단계에서의 오류 해결이 어려워지게 되었다. 이러한 문제점을 해결하기 위하여 비동기적이고 동시 발생적인 시스템의 정보 흐름을 분석하고 표현할 수 있는 모델 개발이 추진되었고, 이러한 모델중의 하나가 페트리 넷이다[2]. 페트리 넷은 간결한 그래프식 표현 방법과, 시스템의 동작 특성을 결정하고 입증할 수 있는 분석 도구의 제공 및 하향(top-down) 혹은 상향(bottom-up) 접근 방식을 이용한 설계 기법을 제공한다. 이러한 장점으로 인하여 자원의 분배와 운영체제, 큐잉(queueing) 네트워크, 트래픽 제어, 분산 체계 시스템, 수학적 증명, 두뇌 모델링 등 많은 연구 분야에서 페트리 넷이 사용되고 있다. 그러나, 오직 하나의 토큰

형태만을 제공하고, 비어있는 입력 장소(empty place)를 점화 가능 조건으로 사용할 수 없으며, 점화의 우선순위 및 시그널로부터 얻은 정보에 대한 점화의 분류를 표현할 수 없는 제약 사항을 가진다. 이러한 제약 사항으로 인하여 통신 프로토콜과 같은 스위칭 시스템의 동작 특성을 해석하고 분석하기에는 어려움이 있다.

SDL로 명세한 통신 프로토콜을 페트리 넷을 이용하여 동작 특성을 분석하기 위해서는 SDL로부터 페트리 넷로의 변환이 이루어져야 하고, SDL의 페트리 넷 변환은 SDL 개념과 일치하는 다음과 같은 일관적인 모델링 개념이 요구된다.

- EFSM의 개념을 표현할 수 있어야 한다.
- 제어 정보 및 자극/반응 메시지를 표현하기 위하여 하나 이상의 토큰 형태를 제공할 수 있어야 한다.
- 향후 상태 천이를 위하여 사용되는 저장 시그널을 표현할 수 있어야 한다.
- 시그널로부터 얻은 정보 혹은 기억장치의 데이터를 참조하여 상태 천이 경로를 결정할 수 있어야 한다.
- 큐(queue) 및 시그널의 비영속성(non-persistence)을 표현할 수 있어야 한다.
- 시스템의 계층적 구조를 표현할 수 있어야 한다.

2. 수치 페트리 넷(Numerical Petri-Nets)

수치 페트리 넷은 페트리 넷의 기본 규칙과 심볼 및 동작 규칙을 유지함은 물론, 이해 증진을 위한 서술 능력을 개선한 일반화된 페트리 넷으로써 통신 프로토콜과 Queueing 시스템 및 병렬 처리 시스템과 같은 분야에서 널리 사용되고 있다. 수치 페트리 넷은 시스템의 동작을 좀더 상세하게 표현하기 위하여 아래와 같은 특성을 지원한다[1].

- 토큰(token)은 고유의 값과 특성을 가진다.
- 넷에 연관된 기억장치에 읽고 쓰는 것이 가능하다.
- 각각의 호(arc)에 대한 점화 가능 및 점화 조건을 표현할 수 있다.
- 상태 천이를 위한 점화 가능 조건은 입력 장소(input place)내의 토큰과 그에 연관된 기억장치에 존재하는 자료에 의하여 결정된다.

- 상태 천이의 점화 동작은 입/출력 장소(input/output place)에서 토큰의 점화와 그에 연관된 기억장치에 존재하는 자료에 대한 조작을 포함한다.

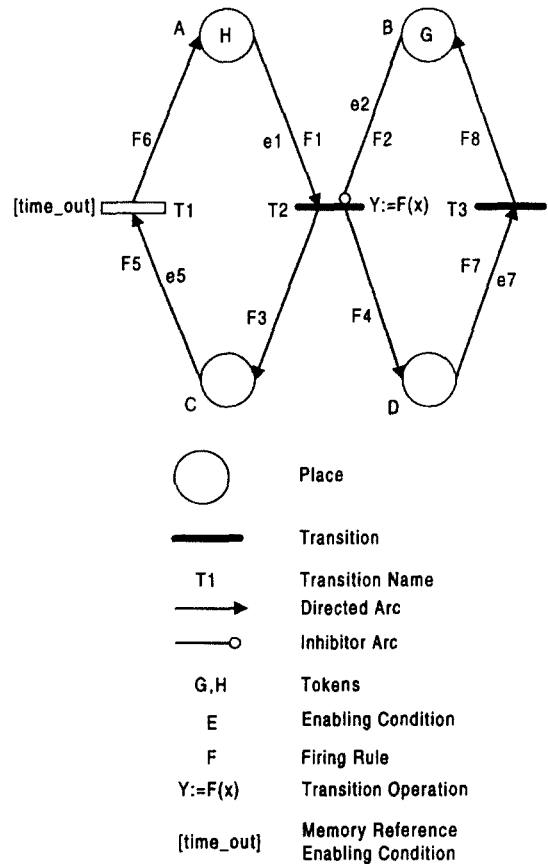


그림 4. NPN 그래프의 예

수치 페트리 넷의 수행은 수치 페트리 넷 그래프의 고유 특성에 따른 토큰 분포와 각 토큰 수에 의하여 제어되며, 상태 천이는 오직 점화 가능할 경우에 발생한다. 점화 가능이란 입력 장소에 존재하는 토큰의 종류와 수가 상태 천이를 위한 점화 가능 조건을 만족하고, 상태 천이에 연관된 기억장치의 조건을 동시에 만족할 때 발생한다. 다음은 상태 천이의 동작 순서를 보인 것이다.

- 입력 상태 천이 함수에 따라 입력 장소로부터 토큰을 제거한다.

- 네트 기억장치의 자료에 대하여 상태 천이 동작을 수행한다.
- 출력 상태 천이 함수에 따라 출력 장소에 토큰들을 추가한다.

호는 상태 천이가 발생할 때, 출력 장소에 추가할 토큰을 정의하는 점화 규칙을 가진다. 그림 4에서 $F1, F2, F5, F7$ 은 입력 호에 대한 점화 규칙을, $F3, F4, F6, F8$ 은 출력 호에 대한 점화 규칙을 나타낸 것이다.

3. 변환 법칙

본 절에서는 SDL로 명세한 통신 프로토콜의 동작 특성을 분석하기 위하여 SDL로부터 수치 페트리 네트의 변환 법칙을 기술한다. 일반적으로 통신 시스템의 동작은 실시간 처리 프로세스들의 상호 통신이며, SDL에서는 논리 기능을 수행하는 실체를 프로세스로 정의한다. 따라서, SDL로 명세한 통신 프로토콜의 동작 특성 분석을 위하여 프로세스 상태와 논리 동작을 표현한 흐름도를 수치 페트리 네트로 변환하여야 한다. 변환 법칙은 SDL 개념을 유지함은 물론 SDL로 표현할 수 있는 SPC 시스템의 특성을 명확하게 표현할 수 있도록 일관된 방법에 의하여 이루어져야 한다. 아래의 변환 법칙은 SDL로부터 수치 페트리 네트의 변환이 1절에서 기술한 SDL의 페트리 네트 모델링 개념을 유지하고 있으며, 변환이 일대일(1:1)로 일관되게 이루어 짐을 보여준다. 그림 6은 향후 상태 천이를 위한 저장 시그널과, 시그널의 비

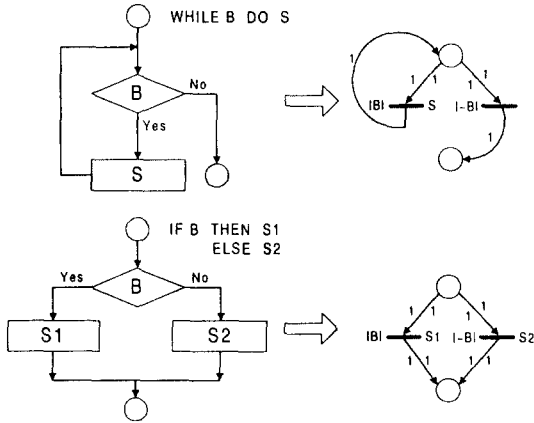
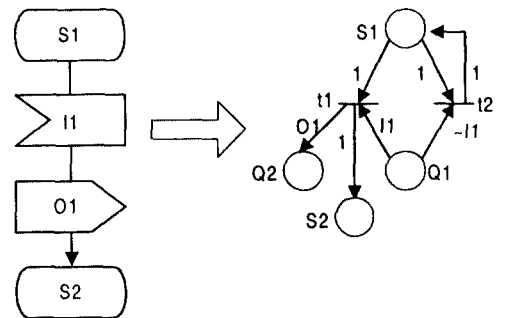


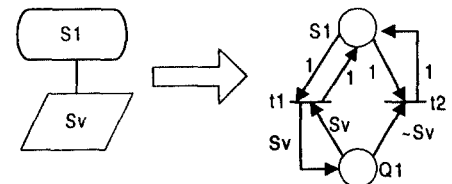
그림 5. NPN 그래프로 표현한 구조적 프로그래밍의 예

그림 4는 수치 페트리 네트의 예를 보인 것이며, 그림 5는 구조적 프로그래밍을 수치 페트리 네트 그래프로 표현한 예를 보인 것이다. 수치 페트리 네트의 입/출력 장소에 존재하는 토큰들은 고유 특성과 값을 가질 수 있다. 모든 입력 호는 입력 호에 연결된 입력 장소에 존재하는 토큰에 대한 점화 가능 조건 e 를 가진다. 모든 천이 막대는 네트 기억장치로부터 자료를 읽기 위한 기억장치 점화 가능 조건 MR 을 가질 수 있다. 상태 천이는 천이 막대에 연결된 모든 입력 호에 대한 점화 가능 조건과 기억장치 점화 가능 조건을 동시에 만족할 때 발생한다.

그림 4에서 상자 모양의 천이 막대 $T1$ 은 시스템의 계층적 구조를 추론하여 모델링하기 위한 것이며, 천이 막대에 연결된 입력 호 중에서 화살표 대신 작은 원을 가지는 입력 호는 시스템 동작의 우선 순위를 모델링하기 위한 것으로 스위칭 이론에 의한 not의 의미를 가진다. 즉, 그림 4에서 천이 막대 $T2$ 를 위한 상태 천이의 점화 가능 조건 $e1$ 과 $e2$ 중에서 $e2$ 는 입력 장소 B 가 토큰을 가지지 않는 조건이 된다. 모든 입력 호는 상태 천이가 발생할 때, 입력 장소로부터 제거할 토큰을 정의하는 점화 규칙을 가진다. 모든 출력



(a) 시그널의 비영속성을 모델링한 예



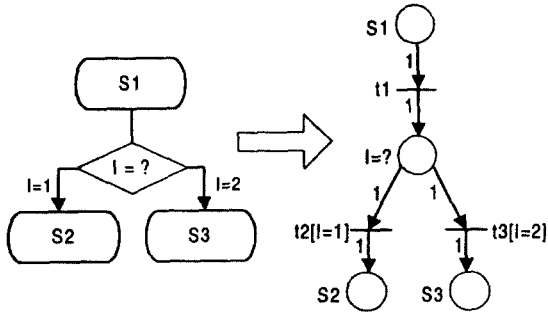
(b) 저장 시그널을 모델링한 예

IV. 동작 특성 분석

1. 도달성 그래프

SDL 명세로부터 수치 페트리 넷으로 모델링한 통신 프로토콜의 동작 기능 및 특성을 분석하기 위한 도구로는 시스템의 상태 천이를 나무(tree) 형태로 나타낼 수 있는 도달성 나무(reachability tree)를 이용하는 방법과 행렬 방정식(matrix equation)을 이용하는 방법이 있다. 도달성 나무에 의한 분석 방법은 장소(place)내의 토큰 수가 무한함을 나타내는 w 심볼로 인하여 도달성과 생존성 및 상태 천이 순서를 정확히 검증하는 데에 어려움이 있다. 행렬 방정식에 의한 분석 방법은 시스템 구조 및 천이 순서를 충분히 표현할 수 없는 단점이 있다[2].

III장 3절의 변환 법칙에 따라 SDL 명세로부터 얻어진 수치 페트리 넷은 상태 천이시 토큰 수가 SDL



(c) 확장 Finite State Machine을 모델링한 예

그림 6. SDL의 특성을 수치 페트리 넷으로 모델링한 예

영속성, 및 EFSM과 같은 SDL의 특성을 수치 페트리 넷으로 모델링한 예를 보인 것이다. 그림 7은 아래의 변환 법칙을 이용하여 그림 3의 시그널 변환의 논리 기능을 수치 페트리 넷으로 표현한 것이다.

- 프로세스에 할당되는 입력 큐는 장소(place)로 변환한다.
- 프로세스의 시작/종료 심볼, 프로시듀어의 시작/종료 심볼 및 상태 심볼은 장소로 변환한다.
- 프로세스의 각 상태 천이는 상태 천이 막대로 변환한다.
- 시그널의 비영속성은 상태 천이 막대로부터 큐 및 상태 장소로 연결되는 점화 가능 조건과 입력 상태 천이 함수로 변환한다.
- 프로세스의 제어 흐름은 연관된 장소에 연결되는 점화 가능 조건과 입/출력 상태 천이 함수로 변환한다.
- 타스크 심볼은 변환 법칙에서 제외한다.
- 입력 심볼은 큐 장소로부터 연관된 상태 천이 막대의 점화 가능 조건과 입력 상태 천이 함수로 변환한다.
- 저장 심볼은 큐 및 상태 장소로 연결되는 점화 가능 조건과 입/출력 상태 천이 함수로 변환한다.
- 출력 심볼은 상태 천이 막대로부터 목적지 프로세스의 큐 장소로 연결되는 출력 상태 천이 함수로 변환한다.
- 결정 심볼은 하나의 장소와 기억장치 점화 가능 조건을 가지는 제어 경로 수 만큼의 상태 천이 막대로 변환하며, 장소와 각 상태 천이 막대는 점화 가능 조건을 가진다.

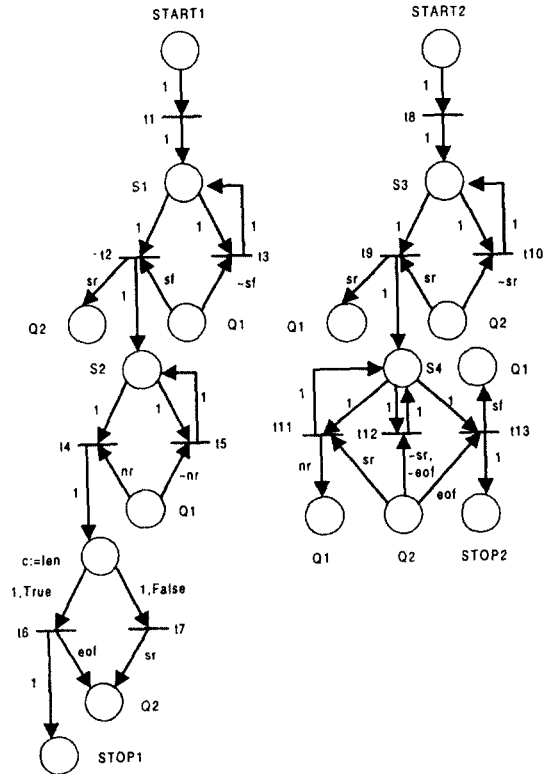


그림 7. 그림 3의 “시그널 변환의 논리 기능”을 수치 페트리 넷으로 표현한 예

개념에 의하여 항상 하나씩 증가 또는 감소하는 특성을 가진다. 본 절에서는 이러한 특성을 이용하여 도달성 나무에서 제시한 w 심볼로 인한 정보 손실을 막고, 시스템 상태 천이 순서를 보다 정확하게 표현할 수 있는 도달성 그래프(reachability graph)를 제안한다. 시스템의 동작 특성 및 상태 천이 순서를 보다 효과적으로 분석하기 위한 도달성 그래프는 다음과 같은 요소들로 구성되어 있다.

• 노드(node)

수치 페트리 넷으로 표현한 시스템 상태를 나타내는 토큰 분포

$$N = (P, PL, CL, M)$$

여기에서,

P : 장소(place)들의 유한 집합

PL : 부 노드 목록을 위한 지시자

CL : 자 노드 목록을 위한 지시자

M : 입력 큐 장소내의 토큰 분포

• 호(arc)

입력의 토큰 분포에서 다른 토큰 분포로의 상태 천이 $A = (EC(t), B(t), F(t))$

여기에서,

$EC(t)$: 상태 천이 t 를 위한 점화 가능 조건

$B(t)$: 상태 천이 t 를 위한 입력 상태 천이 함수

$F(t)$: 상태 천이 t 를 위한 출력 상태 천이 함수

• 토큰 갯수 심볼

토큰 분포에 나타난 각 토큰의 갯수에 대한 표현

$$\text{토큰 갯수} = \{ \#0, \#1, \#2 \}$$

여기에서,

$\#0$: 토큰 갯수가 "0"임을 의미하며, 노드의 토큰 분포에 나타나지 않음

$\#1$: 토큰 갯수가 "1"임을 의미

$\#2$: 토큰 갯수가 "2" 이상임을 의미

SDL 개념과 일치하는 그림 8의 도달성 그래프 알고리즘은 도달성 나무 알고리즘과 거의 유사하나, 토큰 갯수가 무한함을 나타내는 w 심볼을 제거하고 이에 따른 새로운 상태의 생성과 상태 천이 순서의 추적이 가능하도록 설계되었다.

• Generate Symbol Table and Extended Finite State Machine from SDL Specifications.

• In Petri Nets, system status is represented by marking.

• In the case of Numerical Petri Nets derived from SDL Specifications, marking is composed of state places which occupied control tokens, and the content of queue places.

• Let F is the set of frontier nodes to be processed.

• Let A is the set of state nodes which has been generated, previously.

While (F is non-empty)

{

• Let f is an element of F .

• The notation of $[F]Si$ means the identity of i th state place is occupied by a control token.

• The notation of $[F]Qi(no)$ means the number of i th colored token of the queue place in frontier node F .

For all transitions which are enabled in marking F , create a new node N .

If (marking $[F]Qi(no) = \text{many}$)

{

new marking $[N]Qi(no) = \text{many}$;

}

Else if (exists an internal node I on the path from the root node to F , and marking $[I]Si = \text{marking}[N]Si$ for each state place, and $[I]Qk(no) < [N]Qk(no)$ for each token type in a queue place)

{

new marking $[N]Qk(no) = \text{many}$;

}

Else

{

i th new marking $[N]I = \text{next marking}([F], ij)$

}

If (marking $[F]Si = \text{marking}[N]Si$ for each state place, and marking $[n]Qk(no) = \text{many} - 1$ for each colored token in a queue place)

{

delete marking $[N]Qk(no)$ in the new node N ;

}

• It means that a state transition concerned with an arbitrary status can trigger more one times.

For a new node N , construct parent list and children list.

If (N is an error status) generate error report;

Else

{


```

    if (N is not in A or F) add N to F;
    delete f from F;
    add f to A;
}
}
    
```

그림 8. 도달성 그래프 알고리즘

SDL로 명세한 통신 프로토콜의 동작은 프로세스들의 자극/반응을 나타내는 시그널을 통한 상호 통신으로 생각할 수 있고, 시그널의 저장을 위하여 SDL에서는 각 프로세스에 큐가 배정된다. 프로세스의 동작은 큐내의 시그널 도착 순서에 의하여 규정되어지기 때문에 큐내의 정보는 동작 상태를 규정할 수 있는 중요한 정보이다.

그러나, 시스템 명세 단계에서 큐내의 시그널 도착 순서를 고려하기에 상당한 어려움이 따른다. 따라서, 제안한 도달성 그래프는 큐내의 시그널 존재 유무만을 표현하여 현재의 시스템 상태로부터 상태 천이가 가능한 모든 상태를 생성함으로써 도달 가능한 모든 상태 천이 순서를 생성한다. 또한, 상태 천이 순서를

추적하기 위하여 시스템 상태는 노드 명과 부 노드(parent node) 지시자 및 자 노드(child node) 지시자를 가진다. 도달성 그래프의 각 노드는 주어진 프로세스들의 상태와 큐내의 시그널 정보로써 시스템 상태를 표현하며, 호는 시스템 상태 천이를 유발하는 수치 페트리 넷의 입/출력 상태 천이 함수를 나타낸다.

그림 9는 SDL 명세로부터 도달성 그래프를 생성하기 위한 내부 흐름도를 보인 것이며, 그림 10은 그림 7의 시스템 초기 상태로 Q1과 Q2에 각각 시그널 *sf*와 *sr*을 주었을 때 천이할 수 있는 모든 시스템 상태를 생성한 도달성 그래프이다. 그림 10의 각 호에 표시한 정수 1은 제어 토큰을 나타내며, 문자열은 시그널 토큰이나 결정에 대한 응답 값을 나타낸다. 샵 심볼, #은 시그널 토큰의 수를 표시하며, #1은 시그널 토큰의 수가 1개임을 나타내고 #2는 시그널 토큰의 수가 2개 이상임을 표시한 것이다.

2. 동작 특성 및 분석 기법

페트리 넷의 서술 능력을 개선한 수치 페트리 넷은 페트리 넷의 동작 규칙을 유지함으로써 도달

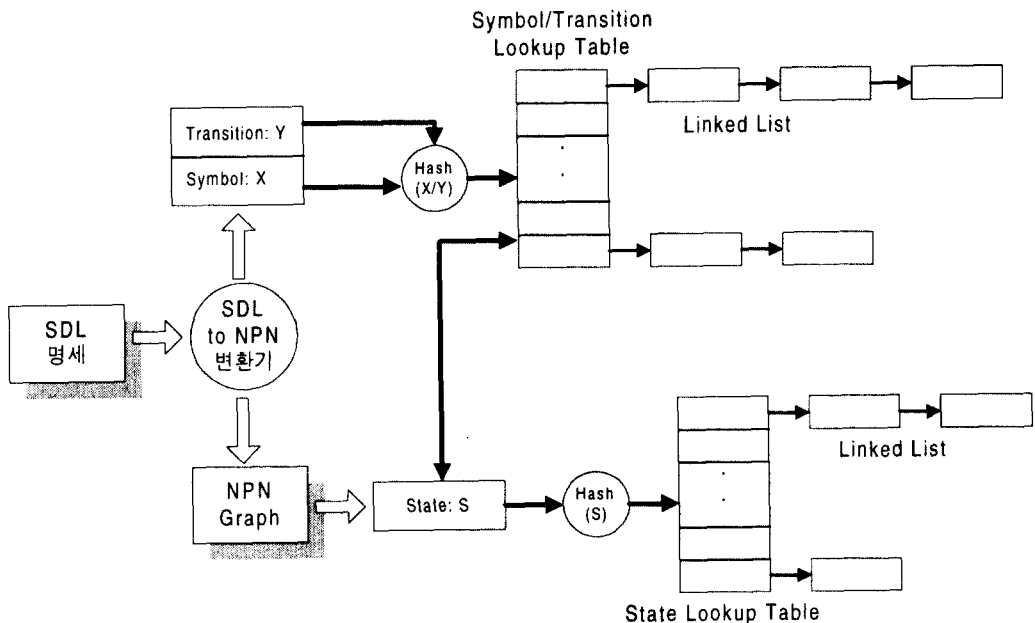


그림 9. 도달성 그래프 생성을 위한 내부 흐름도

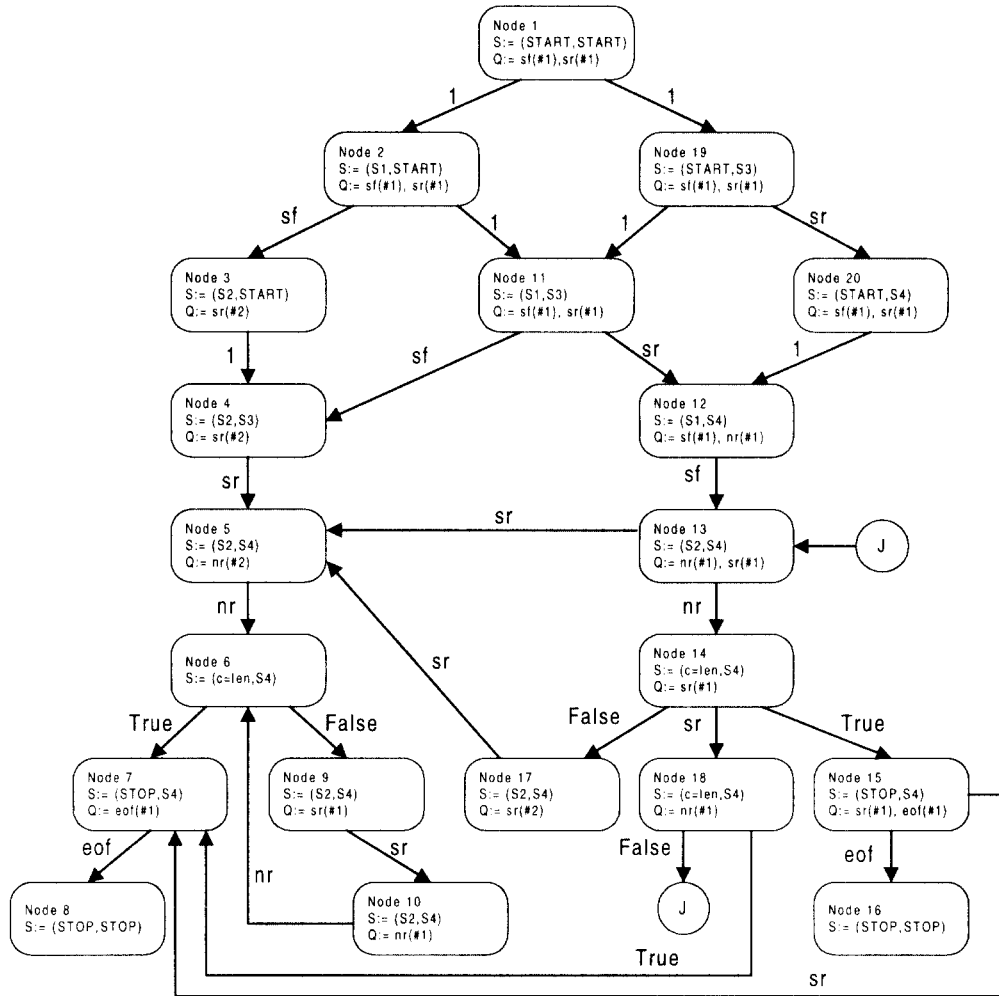


그림 10. 그림 7의 Q1과 Q2에 각각 시그널 sf와 sr을 주었을 때의 도달성 그래프

성 및 생존성과 같은 동작 특성을 분석할 수 있다[1]. 수치 페트리 넷으로 표현한 시스템의 동작 특성을 분석하기 위하여 제안한 도달성 그래프는 페트리 넷의 도달성 나무 알고리즘의 무한 토큰 갯수를 나타내는 w 심볼을 보완하고, 중복된 시스템 상태를 보유함으로써 상태 전이 순서를 보다 정확하게 추적할 수 있다. 따라서, 도달성 그래프를 이용한 동작 특성 분석 기법은 도달성 및 생존성과 같은 특성을 도달성 나무 기법과 동일하게 적용하여 분석하게 된다[2]. 수치 페트리 넷으로 표시한 시스템의 동작 오류 가

능성을 보여주는 오류 메시지는 발생 가능한 모든 상태 전이 경로를 보여주는 도달성 그래프의 구성중에 생성된다. 이러한 오류 메시지는 SDL로 명세한 통신 프로토콜의 동작 특성을 분석하는 자료로 사용할 수 있다. 도달성 그래프를 이용하여 분석 가능한 동작 특성은 다음과 같다.

• Deadlock 가능성(no further progression)

도달성 그래프에서의 deadlock 가능성은 임의의 시스템 상태가 이미 생성된 다른 상태와 중복되지 않음

면서 더 이상 상태 천이가 불가능한 노드로 규정할 수 있다. Deadlock 상태는 도달성 그래프 구성중에 초기 노드로부터 상태 천이가 불가능한 노드까지의 경로를 부 노드 지시자와 자 노드 지시자를 이용하여 추적함으로써 시스템의 deadlock 가능성을 보여주게 된다.

• 미 사용 상태 천이 함수(inexecutable transition)

그림 9에서 보인 것처럼, SDL로 명세한 시스템은 상태 심볼과 상태 천이 함수를 해쉬(hash) 함수를 이용하여 표시하게 된다. 그러나, 도달성 그래프 구성중에 한번도 사용하지 않은 상태 천이 함수가 존재한다면 이는 설계상의 오류로 볼 수 있다. 미 사용 상태 천이 함수를 위한 오류 메시지는 도달성 그래프를 구성한 다음 상태 천이 함수에 대한 해쉬 표를 검증함으로써 가능하다.

• 루프-백(loopback)

도달성 그래프 구성중에 임의의 시스템 상태 노드가 이미 생성된 노드이면서, 다른 시스템 상태로 천이할 수 있는 천이 경로가 없을 경우, 이러한 상태를 루프-백으로 규정할 수 있다. 루프-백은 시스템 동작상의 오류로 볼 수 있으나, 항상 오류로 간주할 수는 없다. 따라서, 도달성 그래프에서는 루프-백이 발생하는 시작 노드와 종료 노드의 상태 천이 경로를 부 노드 지시자와 자 노드 지시자를 이용하여 제공함으로써 시스템 설계자가 오류에 대한 판단을 할 수 있도록 한다.

• 도달성(reachability)

주어진 초기 상태로부터 임의의 시스템 상태에 대한 도달성 유무 및 상태 천이 경로는 그림 9의 도달성 그래프를 위한 해쉬 표를 이용하여 검증이 가능하다. 시스템 설계자는 이러한 정보를 이용하여 시스템의 동작 특성을 설계 단계에서 검증하는 것이 가능하다.

V. 결 론

SDL로 명세한 통신 프로토콜은 일반적으로 대규모이고 복잡하지만 상대적으로 높은 신뢰성을 요구하는 특성이 있다. 이러한 특성으로 인하여 시스템 구현 이전에 설계 단계에서의 동작 특성을 분석하는 것은 구현하고자 하는 시스템의 신뢰성을 높일 수 있음은 물론, 시스템 개발 비용의 절감을 기대할 수 있다. 본 논문에서는 SDL 표현 프로토콜의 동작 특성

을 분석하기 위하여 SDL의 페트리 넷 모델링 개념과 수치 페트리 넷으로의 변환 법칙을 제시하였다. 또한, 동작 특성의 분석을 위하여 도달성 나무를 개선한 도달성 그래프 알고리즘을 제안하였으며, 도달성 그래프를 이용하여 분석 가능한 동작 특성 및 분석 기법을 기술하였다. 그러나, 페트리 넷에서의 동작 특성 분석 기법은 도달성 그래프를 구성할 때, 도달 가능한 모든 시스템 상태를 생성하기 때문에 시스템 상태의 검색 시간 및 메모리 사용을 폭발적으로 요구할 수 있다. 본 논문에서는 해쉬 표를 이용한 도달성 그래프 구성 방법을 사용함으로써 검색 시간 및 메모리 요구 사항을 고려하였으나, 이를 최적화할 수 있는 분석 기법과 객체 지향 SDL 개념을 위한 모델링 기법의 연구가 계속되어야 할 것이다.

참 고 문 헌

1. F. J. W. Symons, Introduction to Numerical Petri Nets, a General Graphical Model of Concurrent Processing Systems, Australia Telecommunication Journal, vol. 14, pp. 28-33, 1980.
2. J. L. Peterson, Petri Net Theory and Modeling of System, Prentice Hall Inc., pp. 1-290, 1981.
3. C. Block, K. C. Kang, E. Chikofsky and D. Teichroew, Usage of the System Encyclopedia Manager (SEM) System with the CCITT Functional Specification and Description Language(CCITT/SDL), Univ. of Michigan, 2nd CHILL Conference, Lisle, Illinois, U. S. A, 1983.
4. CCITT, SDL Newsletter No. 10, Annex 1, 2, 3, Red Book Z. 100 - Z. 104, Geneva, 1987.
5. Eirik Vefsnmo and A.S. Computas, DASOM-A SDL TOOL, in SDL 87: State of Art and Future Trends, ed. R. Saracco and P.A.J. Tilanus, North-Holland, pp. 35-42, 1987.
6. SESSION-Dynamic Analysis, in SDL 89: The Language at Work, ed. O. Fergemand and M. M. Marques, North-Holland, pp. 209-279, 1989.
7. Jin P. Hong, W. Choi, Yeon S. Shin, Hwan C. Kim and Jung H. Choi, Integrated SDL Environment, SDL 89: The Language at Work, ed. O.

- Fergemand and M. M. Marques, North-Holland, pp. 117-126, 1989.
8. Hwan C. Kim, W. Choi, Chu H. Yim, and Jin P. Hong, The Automated Verification of SDL Specifications Using Numerical Petri-nets, SDL 91: Evolving Methods, ed. O. Faergemand and R. Reed, Elsevier Science Publishers B. V. (North-Holland), pp. 83-94, 1991.
 9. Z100(1993), CCITT functional Specification and Description Language(SDL), ITU-T, June 1994.
 10. Hwan-Chul Kim, The Modeling of Switching Systems Specified in SDL Using Petri Nets, International Technical Conference on Circuits/Systems, Computers, and Communications, pp. 919-922, 1996.



김 환 철(Hwan Chul Kim) 정회원
1984년 2월: 경북대학교 전자공학과 전자계산기 전공 공학사

1986년 2월: 한국과학기술원 전산학과 공학석사

1986년 2월~1991년 2월: 한국전자통신연구원 연구원

1991년 3월~현재: 한국전자통신연구원 선임연구원
관심분야

- 소프트웨어 공학
- 소프트웨어 개발 방법론
- 분산처리시스템
- 분산 데이터베이스 시스템
- 위성통신 서비스 시스템
- 시스템 엔지니어링