論文 94-19-6-13

메시지 의미관계를 이용한 프로토콜 변환 방법에 관한 연구

正會員 權 雲 哲* 正會員 沈 泳 燮* 正會員 李 東 豪*

A Study on the Protocol Conversion Method using Semantic Relation between Messages

Woon Chul Kwon*, Young Seob Shim*, Dong Ho Lee* *Regular Members*

要 約

본 논문은 적합성을 만족하는 프로토콜 변환 방법에 메시지간의 의미관계를 추가하여 프로토콜 변환기를 생성하는 방법을 제시한다. 특히, 제안된 방법은 정형화된 기법과 하향식 방법을 도입하여 서로 다른 포로토콜로 구현된 컴퓨터 네트워크 사이의 프로토콜 불일치를 해결하는 효율적인 최다(maximal) 변환기를 유도한다. 따라서 제시된 방법을 사용하여 변환기를 유도할 경우, 불필요한 상태나 전이들의 상당 부분이 쉽게 제거된다.

ABSTRACT

This paper presents a method of constructing the protocol converter where the semantic relation between messages is added to the protocol conversion method satisfying with conformity property. This method comes from a formal method with top-down approach and then derives efficient maximal converter to overcome protocol mismatch between computer networks implemented with different protocols. When a converter is made with this method, the most useless portions of the maximal converter are removed easily.

I. 서 론

컴퓨터 통신이 다중화되고 보다 향상된 통신 서비스에 대한 수요가 늘어남에 따라, 컴퓨터 네트워크의 상호연동(internetworking)에대한 필요성이 증가하고 있다. 그러나, 서로 다른 네트워크에 접속된 사용자간의 통신은 어려움이 있다. 함께 동작하도록 설계되지 않은 시스템 사이에 통신을 할 경우 프로토콜 불일치가 발생한다.

프로토콜 불일치 문제를 해결하는 가장 명확한 방법은 한 프로토콜을 다른 프로토콜로 대치하거나 두 프로토콜을 어떤 공통의 프로토콜로 대치하는 것이다. 다른 해결 방법으로 기존의프로토콜을 대응 프로

*光云大學校 電子計算學科
Dept. of Computer Science, Kwangwoon University
論文番號 : 9422
接受日字 : 1994年 1月 19日

토콜로 변환시키는 엔티티(entity)를 두는 방법이 있다. 서로 다른 프로토콜의 변환을 수행하는 엔티티를 프로토콜 변환기라 한다.

일반적인 프로토콜 변환에 관한 문제의 분석은 Green[5]에 의해 시작되었다. Green은 프로토콜 변환이 계속 필요할 것이라 주장하면서 프로토콜 변환 분야에서 이론적 배경이 부족하다는 것을 지적하였다.

Lam[6]은 변환시스템의 오류없음을 결정하는 방법을 제시하고 제한된 프로토콜의 자동변환기 생성 방법을 제시했다. 두 프로토콜에서 의미적으로 공통인 이미지를 찾고 요구되는 나머지 기능을 변환기 구성을 통하여 완성하였다. Calvert와 Lam[1]은 보다 복잡한 변환기의 정확성에 대한 인식을 위해 프로토콜 검증에 사용된 투영기법을 어떻게 이용할 수 있는가를 보였다. 또한, Calvert와 Lam[2,3,4]은 하향식 접근방법을 사용하여 나눈몫(quotient)을 구하는 방법을 제시하였다. 이 방법은 변환시스템에서 주어진 서비스가 오로지 하나일때만 사용할 수 있다.

Okumura[7,8]는 주어진 두 프로토콜의 메시지사이의 의미관계를 규정하는 변환씨드(conversion seed)를 이용한 방법을 제안하였다.

Yao[9]는 하향식 접근방식을 사용하여, 최종연동시스템이 서비스 명세에 대해 적합성(conformity property)를 만족하도록 하는 방안을 제시하였다.

본 논문에서는 통신 프로토콜 및 프로토콜 변환 모델의 명세를 위한 정형모델(formal model)로 통신유한상태기계(CFSM : Communication Finite State Machine)를 사용한다. 하향식 접근방법을 사용하여 지정된 정형모델로 명세된 입력으로부터 프로토콜 변환기를 자동 생성한다. 그리고 서비스 중심의 프로토콜 변환에 대해서만 고려하며, 전체 변환시스템의 서비스 명세가 주어졌다는 가장아래 프로토콜 변환을 한다.

그리고 적합성을 만족하는 프로토콜 변환 방법에서 메시지간의 의미관계를 부여한 의미사양을 입력으로 추가하여 프로토콜 변환기를 생성하는 방법을 제시하였다. 일반적인 프로토콜 변환에서 입력으로 들어온 명세-프로토콜 명세, 서비스 명세-로는 한 프로토콜의 메시지와 의미적으로 대응되는 다른 프로토콜의 메시지나 메시지 순서열을 알 수 없어서 의미사양을 추가시켰다. 의미사양은 두 프로토콜에서 동일한 의미를 갖는 메시지간의 대응관계를 명세한 것이다.

의미사양을 만족하지 않는 전이들을 제거하므로서 최적에 가까운 변환기를 생성할 수 있다. 하향식 방법을 사용한 기존의 방법들을 이용하여 생성된 변환기는 최다(maximal) 변환기이다. 최다 변환기에 존재하는 불필요한 상태나 전이들은 의미사양을 입력으로 추가시키므로서 쉽게 제거할 수 있음을 알 수 있다.

Ⅱ. 의미관계를 이용한 프로토콜 변환

1. 변환을 위한 정형모델

통신 프로토콜은 통신시스템의 매우 중요한 구성요소이며 네트워크 구성요소들의 상호협동과 메시지의 순차적인 전송을 관리하는 규칙의 집합으로 이루어져 있다. 통신 프로토콜을 표현하는 방법으로서 전이 모델의 한 종류인 통신유한상태기계(CFSM)를 본 논문에서 사용한다.

통신 프로토콜은 대부분 계층구조 형태이다. 계층사이의 상호작용은 서비스 접근점(SAP : Service Access Point)을 통하여 서비스 프리미티브(일종의 메시지)를 주고 받음으로 모델화된다. 따라서 통신 프로세스에 존재하는 전이(transition)를 두 종류로 구분할 수 있다. 첫번째는 동배 전이(peer transition)로서, 대응 프로토콜 엔티티간에 메시지를 주고 받음으로서 발생되는 전이이다. 두번째는 서비스 전이(service transition)로서, 상위층 프로토콜 엔티티(사용자)와 프로토콜 엔티티간에 서비스 프리미티브를 주고 받음으로서 발생되는 전이이다. 이 관계를 그림1에 나타내었다.

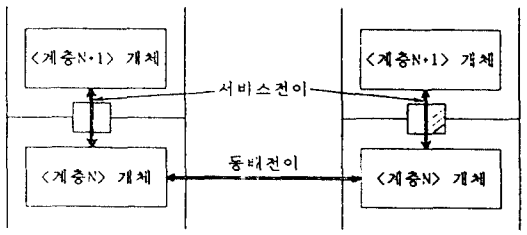


그림 1. 통신 프로토콜 시스템
Fig. 1. A Communication Protocol System

여러 프로토콜 엔티티들로부터 그들 사이의 전반적인 상호동작을 나타내는 글로발 시스템 엔티티를 구성하기 위해 '‖'라는 혼성연산자를 사용한다. 또한, 글로발 시스템 엔티티에서 그것을 구성하는 일부 프

로토콜 엔티티를 추출하기 위해 '|'라는 추출연산자를 사용한다.

통신 프로토콜 P에 속하는 통신 프로세서로 P_A와 P_B가 있고, 프로토콜 P의 전반적인 동작을 나타내는 시스템 엔티티를 P_G라고 하자. 이때, P_G를 '$P_A \| P_B$'로서 간단히 표기하며 P_G의 구성방법은 다음과 같다.

〈정의〉 $P_G = P_A \| P_B = (S_G, I_G, \sum_G, \lambda_G, F_G, T_G)$

$S_G \in S_A \times S_B$

$I_G = \langle I_A, I_B \rangle$

$\sum_G = (\sum_A \cup \sum_B) - (\sum_A \cap \sum_B)$

$\lambda_G = \lambda_P$

$F_G = \langle F_A, F_B \rangle$

$T_G \in S_G \times (\sum_G \cup \lambda_G) \times S_G$

$= \{(\langle a, b \rangle, m, \langle a', b' \rangle) : (m \in \lambda_G \wedge (\{a = a' \wedge b \xrightarrow{m} b') \vee (b = b' \wedge a \xrightarrow{m} a')\}) \wedge (m \in \sum_G \wedge ((a = a' \wedge b \xrightarrow{m} b') \vee b') \vee ((b = b' \wedge a \xrightarrow{m} a')))\}$

단, $a \in S_A$, $b \in S_B$, $s \in S_S$

S : 프로세스에 존재하는 상태집합
I : 프로세스의 초기상태 ($\in$S)
$\sum$: 프로세스에서 다른 프로세서로 보내는 메시지의 집합
λ : 서비스 프리미티브의 집합
F : 프로세스의 마지막 상태
T : 현재상태 s에서 메시지 m을 송수신한 후의 다음 상태 s'를 정의하는 상태전이 관계(s, m, s') s는 m의 head state s'는 m의 tail state

시스템 엔티티 P_G로부터 프로토콜 엔티티 P_A를 추출하는 것을 '$P_A = (P_G) | \sum_A \cup \lambda_A$'로써 표기한다. P_A를 추출하는 방법은 P_G에 있는 전이들 중에 P_A에 속하지 않는 메시지에 의해 이루어지는 전이들을 ε로 변환시킨 후, ε를 제거하는 알고리즘을 사용하여 P_A를 추출한다. P_G에서 P_B를 추출할 때에도 이와 같은 방법을 사용한다.

두개의 프로토콜과 서비스를 명세하는 유한상태기계가 프로토콜 변환의 입력으로 사용된다. 이러한 입력만으로는 프로토콜 P와 Q에 있는 메시지(기능)중에 의미론적으로 서로 관련있는 메시지가 무엇인가를 알 수 없다. 따라서, 각 프로토콜에 있는 메시지간의 관계를 입력으로 추가시킬 필요가 있다. 의미관계를 갖는 메시지들은 서로 비슷한 기능(의미)을 갖는다는 의미등가성(semantic equivalency)을 나타내는 것으로서, 다음과 같이 정의된다.

〈정의〉 의미 등가성(Semantic equivalency)

통신 프로세스 P_A와 Q_B의 메시지중에서 동일한 의미를 갖는 메시지간의 관계및 순서를 명세한 의미사양을 Rx라고 하자. 그리고 생성된 변환기 C에서 초기 상태를 시점으로 생성될 수 있는 메시지 순서열을 γ라고 하자. 의미사양 Rx에서 초기 상태를 시점으로 생성될 수 있는 메시지 순서열을 ρ라고 하자. 변환기 C가 의미등가성을 유지하려면 다음의 조건을 만족하여야 한다.

$\gamma | \sum_{Rx} \subseteq \rho \qquad x = 1, 2, \ldots, n$

2. 변환 방법

입력으로 의미사양이 추가된 경우, 생성된 프로토콜 변환기 C는 다음과 같은 세가지 조건을 만족하여야 정확하다.

(1) 적합성(Conformity) 조건 : $(P_A \| C \| Q_B) | \sum_{PA} \cup \sum_{QB}$가 서비스 명세 S와 동일해야 한다[9].
(2) 의미 등가성(Semantic equivalency) 조건 : 변환기 C에서 생성된 메시지 순서열을 의미사양 M의 메시지로 투영했을 때의 메시지 순서열을 μ라고 하자. 이때, 메시지 순서열 μ를 의미사양 M의 메시지 순서열로 받아들일 수 있어야 한다.
(3) 활동성(Liveness) 조건 : 시스템 엔티티 G의 어떤 상태 s에대해, 메시지 순서열의 head state가 s이고 tail state가 F_G인 메시지 순서열이 존재하여야 한다.

본 논문에서는 주어진 입력으로부터 생성된 시스템 그래프로부터 변환기 C를 점차적으로 유도한다. 프로토콜 P, Q 및 서비스 사양 S를 입력으로 하여 $P_A \| S \| Q_B$을 수행하여 시스템 그래프 G를 생성한다. $(G) | \sum_{PA} \cup \sum_{QB}$와 서비스 S는 동일하지만, G를 대응 전이로 투영한 것이 G에 존재하지 않는 경로를 포함하는 $P_A \| C' \| Q_B$의 C'를 생성할 수 있다. 결과적으로 C'를 변환기로 사용할 수 없다. 왜냐하면 $(P_A \| C' \| Q_B) | P_A \cup \sum_{QB}$가 서비스 사양 S와 동일하지 않을 수 있기 때문이다.

그러나, 시스템 그래프 G로부터 유도된 C'로 $P_A \|$

C′‖Q_B와 시스템 그래프 G가 서로 동일함을 보장하는 방법만 보일 수 있다면 G를 서비스 사양 S에 대한 검증을 위해 이용할 수 있다. 만약 G가 S에 대해 적합하다면 G로부터 유도된 변환기 C′가 주어진 목적 프로토콜에 대한 최종 변환기 역할을 할 수 있다. G가 $P_A \| ((G) | \lambda_{PA} \cup \lambda_{QB}) \| Q_B$와 같다는 성질을 갖는 시스템 그래프 G를 완전 동기 시스템 그래프(properly synchronized system graph)라고 한다. 반면에 이러한 성질을 갖지 않는 시스템 그래프 G를 비완전 동기 시스템 그래프(nonproperly synchronized system graph)라고 한다[9].

비동기 상태(unsynchronized state)를 시스템그래프에서 제거하며 완전 동기 시스템 그래프를 얻는다. 비동기 상태에 대한 정의는 다음과 같다. 시스템 그래프 $G = P_A \| S \| Q_B$에 대해서, (단, a, s 및 b는 P_A, S 및 Q_B의 각 지역상태이다) a 또는 b가 서비스 전이 t의 head state이나 시스템 그래프 $G = P_A \| S \| Q_B$의 한 상태인 ⟨a, s, b⟩가 t의 head state가 아닌 경우에 그러한 상태를 비동기상태라고 한다.

그리고, 시스템 그래프를 생성하는 방법은 다음과 같다.

$$P_G = P_A \| S \| Q_B = (S_G, I_G, \Sigma_G, \lambda_G, F_G, T_G)$$

$$S_G \in S_A \times S_S \times S_B$$

$$I_G = \langle I_A, I_S, I_B \rangle$$

$$\Sigma_G = (\Sigma_A \cup \Sigma_B)$$

$$\lambda_G = \Sigma_S$$

$$F_G = \langle F_A, F_S, F_B \rangle$$

$$T_G \in S_G \times (\Sigma_G \cup \lambda_G) \times S_G$$

$$= \{(\langle a, s, b \rangle, m, \langle a', s', b' \rangle) : (m \in \lambda_G \wedge ((a = a' \wedge b \xrightarrow{m} b') \vee (b = b' \wedge a \xrightarrow{m} a'))) \vee (m \in \Sigma_G \wedge ((a = a' \wedge b \xrightarrow{m} b') \vee b') \vee (b = b' \wedge a \xrightarrow{m} a'))\}$$

단, $a \in S_A, b \in S_B, s \in S_S$

완전 동기 시스템과 의미관계의 개념을 이용하여, 프로토콜 엔티티 P_A와 Q_B간의 통신 중개 역할을 하는 프로토콜 변환기 C를 생성하는 방법은 다음과 같다.

(1) 시스템 그래프 $G = P_A \| S \| Q_B$를 구성한다. 단, 생성된 시스템 그래프는 아래의 세조건을 만족해야 한다.

a. $(G) | \sum P_A \cup \sum P_B = S$

b. $(G) | \sum P_A \cup \lambda P_B = P_A$

c. $(G) | \sum Q_B \cup \lambda Q_B = Q_B$

(2) 시스템 그래프에서 모든 비동기상태를 제거한다.

(3) 의미등가성을 유지하지 않는 전이들을 제거한다.

(4) 시스템 그래프에서 활동성을 유지하지 않는 상태들을 제거한다.

(5) 시스템 그래프가 서비스 S에 대해 만족하는지를 검증한다.

$(G) | \sum P_A \cup \sum P_B = S$

(6) G가 S를 만족한다면 G로부터 변환기 C를 생성한다.

$C = (G) | \lambda P_A \cup \lambda Q_B$

(7) 변환기 C에 있는 '+'와 '−'기호를 각각 '−'와 '+'로 대치 시킨다.

Ⅲ. 적용 및 결과 분석

1. 적용 사례

본 절에서는 제시된 프로토콜 변환 방법을 적용한 사례를 들어본다. 예제에서 사용한 프로토콜은 전형적인 AB(Alternate Bit) 프로토콜과 NS(Non-Sequenced) 프로토콜이다. AB 프로토콜에서는 메시지 순서 번호를 사용하지만, NS 프로토콜은 메시지 순서번호를 사용하지 않는다.

AB 프로토콜의 명세는 그림 2에, NS 프로토콜의 명세는 그림 3에 도시되어 있다. 여기에서 메시지의 수신을 '−' 기호로 명시하며, 송신은 '+' 기호로 명시한다. 그리고 노드는 상태를, 간선은 전이를, 레이블은 송수신되는 메시지를 각각 나타낸다.

두 프로토콜의 서비스 명세는 그림 4과 같으며 의미사양은 그림 5와 같다. 두 프로토콜에서 사용된 메

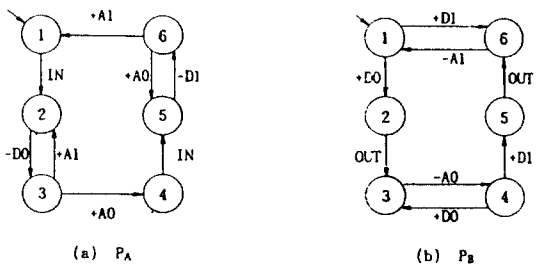


그림 2. AB 프로토콜

Fig. 2. Alternate Bit Protocol

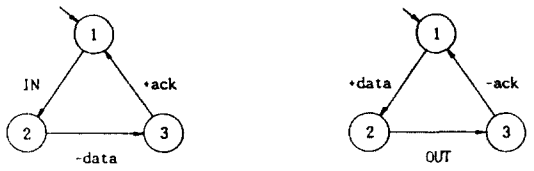


그림 3. NS 프로토콜
Fig. 3. Non-Sequenced Protocol

시지중 데이타 메시지는 'D0', 'D1' 및 'data'이고 확인 응답 메시지는 'A0', 'A1' 및 'ack'이다. 따라서 송신자로부터 Dx를 받아서 수신자로 data를 보내며, 수신자로부터 ack를 받아서 송신자로 Ax를 보낸다.

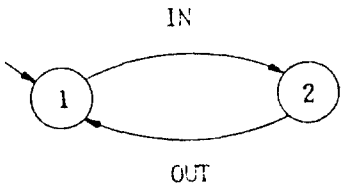


그림 4. 서비스 명세
Fig. 4. Service Specification

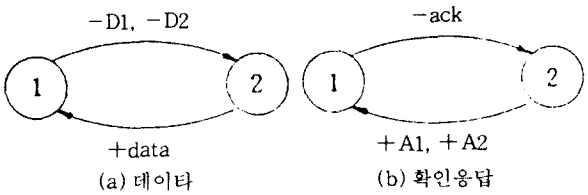


그림 5. 의미사양
Fig. 5. Semantic Specification

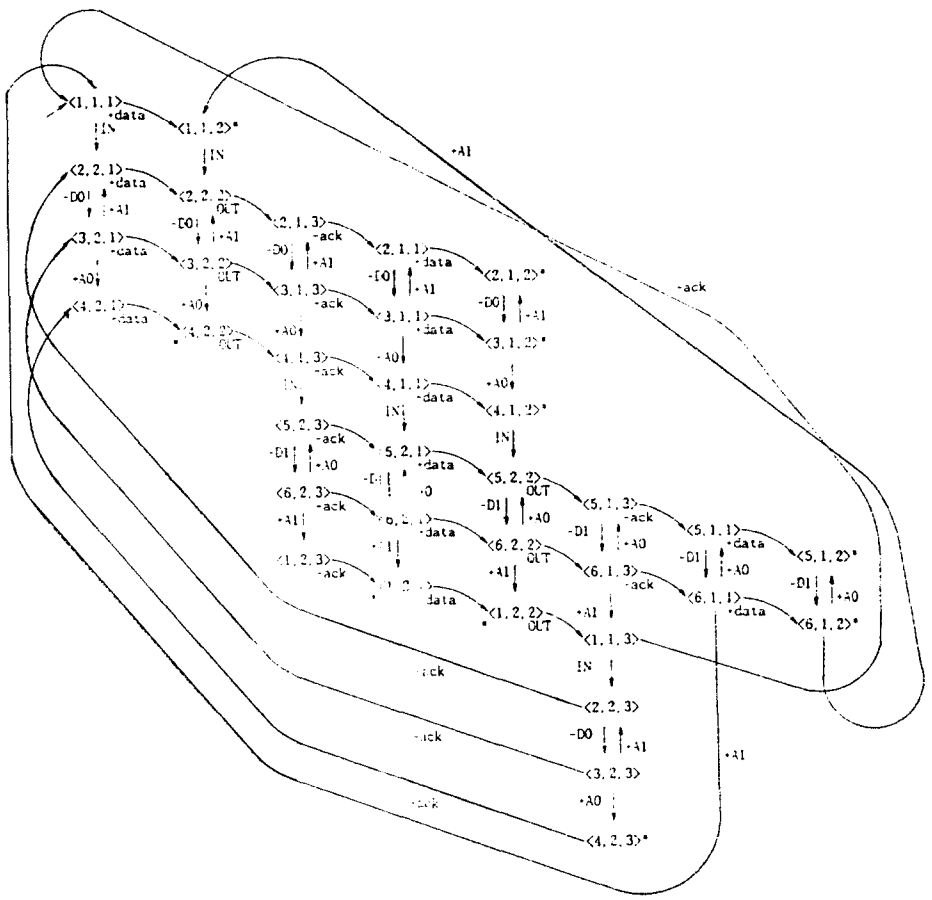


그림 6. 시스템 그래프 $G = P_A \| S \| Q_B$
Fig. 6. System Graph $G = P_A \| S \| Q_B$

이와 같은 입력으로 $P_A \| S \| Q_B$를 수생하여 생성된 시스템 그래프 G가 그림 6에 도시되어 있다. 그림 10에서 '*'으로 표시된 상태는 비동기상태이다. 시스템 그래프 G의 〈4, 2, 1〉상태에서 〈5, 2, 1〉상태로 전이하지 못하는 이유는 서비스 명세에 의해 어떤 'IN'이란 사건이 발생된 후, 'OUT'이라는 사건이 발생한 후에라야 다른 'IN'이란 사건이 발생될 수 있기 때문이다. 그리고 G의 〈1, 1, 2〉상태가 비동기 상태인 이유는 Q_B의 상태는 OUT의 head state인 2상태이나, 〈1, 1, 2〉상태는 OUT이 아닌 IN의 head state이기 때문이다.

그림 7은 비동기상태를 모두 제거한 시스템 그래프로부터 의미등가성을 파괴하는 상태와 전이들을 제거한 후의 시스템 그래프 G′를 나타낸 것이다. G′에는 활동성 성질에 어긋나는 상태가 존재하지 않으므로, G′는 최종적으로 얻은 완전 동기 시스템 그래프이다.

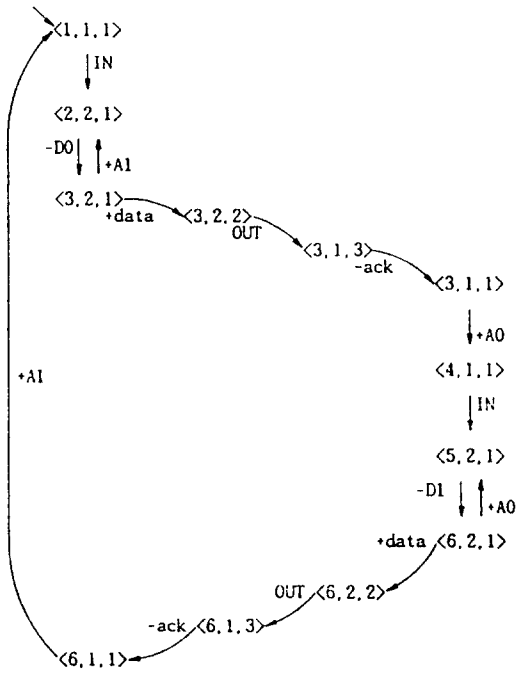


그림 7. 의미 등가성을 만족하는 시스템 그래프
Fig. 7. System Graph be satisfied with semantic equivalence

그림 8은 시스템 그래프 G′에서 추출된 최종 프로토콜 변환기이다. 최종 프로토콜 변환기는 시스템 그래프 G′에 있는 서비스 전이들을 ε로 한 뒤 ε를 제거한 후, 메시지 앞에 표시된 '+'와 '−' 기호를 각각 '−'와 '+' 기호로 대치시킨다. 그런 후 FSM의 최소화 과정을 거쳐 최종적인 결과를 얻는다.

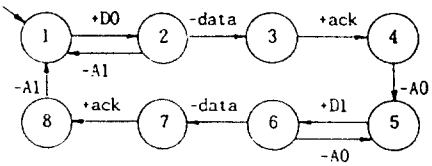


그림 8. 최종 프로토콜 변환기
Fig. 8. Final Protocol Converter

2. 결과 분석

본 논문에서 제시한 방법이 기존의 적합성을 만족하는 방법과 비교하여 상태와 전이의 수가 어느 정도 감소하는가를 알아보기 위해 네가지 프로토콜을 대상으로 하여 12종류의 변환기를 생성하였다. 적용한 프로토콜은 AB 프로토콜, NS 프로토콜, 폴링모델 프로토콜, ack-nack 모델 프로토콜이다. AB 프로토콜과 NS 프로토콜의 명세는 그림 2과 3에 나타나 있다. 폴링모델 프로토콜에 대한 명세는 그림 9과 같고 ack-nack 모델 프로토콜에 대한 명세는 그림 10에 방향성 그래프로 나타나 있다. 4가지 프로토콜로 12개의 프로토콜 변환기를 생성하였을때, 각 변환기에 존재하는 상태와 전이의 갯수를 비교한 것이 표 1과 2에 나타나 있다. 표의 변환기 부분에서 화살표를 기준으로 좌측은 송신측 프로토콜, 우측은 수신측 프로

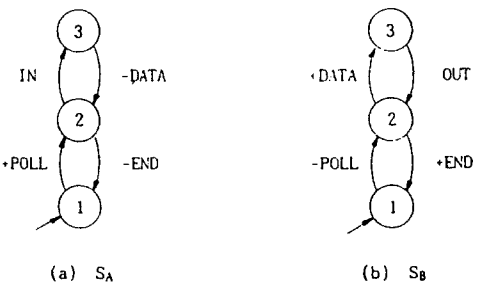


그림 9. 폴링 모델 프로토콜
Fig. 9. Polling model protocol

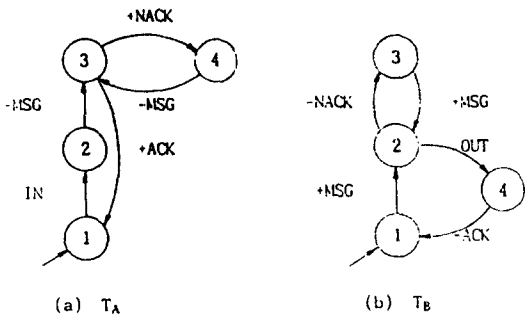


그림 10. ack-nack 모델 프로토콜
Fig. 10. ack-nack model protocol

표 1. 상태 갯수 비교표
Table 1. comparison table of the number of state

변 환 기	SR_N	ST_a	ST_b	향상도(%)
ABP → NSP	2	15	8	46.7
ABP → POLL	1	16	12	25.0
ABP → ACK	2	18	9	50.0
NSP → POLL	2	14	8	42.9
NSP → ACK	1	8	6	25.0
NSP → ABP	2	10	5	50.0
POLL → ABP	1	21	13	38.1
POLL → NSP	1	14	9	35.7
POLL → ACK	1	15	10	33.3
ACK → ABP	2	16	8	50.0
ACK → NSP	2	9	4	55.6
ACK → POLL	1	8	9	25.0
평균 향상도				40.2

* SR_N : 의미관계의 수
* ST_a : 적합성을 고려한 방법에 의한 상태의 갯수
* ST_b : 의미관계를 고려한 방법에 의한 상태의 갯수
* 향상도(%) : $(ST_a - ST_b)/ST_a * 100$

토콜을 각각 나타낸다.

수신측과 송신측이 바뀌는 경우라도 거의 비슷한 향상 결과를 얻었으며 두 프로토콜의 의미관계가 많을수록 보다 향상되는 것을 알 수 있다. 즉, 의미관계가 하나인 경우에는 25%~30%정도 향상되었고 의미관계가 두 개인 경우에는 50%~60%정도 향상되었다.

결과적으로 제시된 방법을 적용하여 최종적으로 생성된 프로토콜 변환기에 존재하는 상태와 전이의 갯수가 기존의 하향식 방법을 사용하여 생성된 프로토콜 변환기에 있는 갯수보다 상당히 적다.

표 2. 전이 갯수 비교표
Table 2. comparison table of the number of transition

변 환 기	SR_N	TR_a	TR_b	향상도(%)
ABP → NSP	2	23	10	56.7
ABP → POLL	1	38	27	28.9
ABP → ACK	2	34	13	61.8
NSP → POLL	2	24	10	58.3
NSP → ACK	1	15	11	26.7
NSP → ABP	2	14	6	57.1
POLL → ABP	1	36	18	50.0
POLL → NSP	1	21	13	38.1
POLL → ACK	1	22	13	40.9
ACK → ABP	2	30	10	66.7
ACK → NSP	2	14	8	42.9
ACK → POLL	1	13	10	23.1
평균 향상도				47.5

* SR_N : 의미관계의 수
* TR_a : 적합성을 고려한 방법에 의한 전이의 갯수
* TR_b : 의미관계를 고려한 방법에 의한 전이의 갯수
* 향상도(%) : $(TR_a - TR_b)/TR_a * 100$

Ⅳ. 결 론

본 연구는 적합성을 만족하는 프로토콜 변환 방법에서 메시지간의 의미관계를 부여한 의미사양을 입력으로 추가하여 프로토콜 변환기를 생성하는 방법을 제시하였다. 본 연구에서는 통신 프로토콜 및 프로토콜 변환모델을 표현하기 위한 정형모델로서 통신유한 상태기계를 사용하였으며 프로토콜 변환기 생성방법을 정형적으로 기술하기 위해 필요한 혼성연산자와 추출연산자를 사용하였다. 그리고 두 프로토콜에 있는 메시지중에 의미론적으로 서로 관련있는 메시지들의 관계를 의미사양으로 표현하였다. 또한 제시한 방법은 서비스 중심의 프로토콜 변환에 대해서만 고려하며, 전체 변환시스템의 서비스 명세가 주어졌다는 가정 아래 프로토콜 변환을 하였다.

제시한 방법의 타당성을 위해 네가지 프로토콜을 대상으로 12종류의 프로토콜 변환기를 생성해 보았다. 결과적으로 변환기의 크기를 나타내는 상태와 전이의 갯수가 기존의 방법에 비해 상당히 감소되었으며, 감소되는 정도가 의미관계의 수와 밀접한 관련이 있다는 것을 알 수 있었다.

동일한 의미를 갖는 메시지간의 대응관계를 명세한 의미사양을 추가시키므로서 최적에 가까운 변환기의 생성이 가능하였다. 하향식 방법을 사용한 기존

의 방법들을 이용하여 생성된 최다변환기에 존재하는 불필요한 상태나 전이들은 의미사양을 입력으로 추가시키므로서 쉽게 제거할 수 있음을 알 수 있었다.

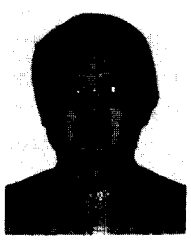
프로토콜 변환기 설계시 발생하는 일반적인 문제점의 하나로 상태의 폭발이 있다. 즉, 복잡한 프로토콜에 대해 프로토콜 변환작업을 할 경우, 본 연구의 방법을 사용하여 상태의 갯수가 감소하였다 하더라도 변환기를 유도하는 과정에서 존재하는 내부상태가 폭발적으로 증가하는 경우가 있다. 본 연구의 범위로 이 부분까지 포함하지 않았으므로 이의 해결 방안이 앞으로 강구되어야 한다.

참 고 문 헌

1. K.L. Calvert and S.S. Lam, "An Exercise in Deriving a Protocol Conversion," Proceedings of SIGCOMM '87 symposium, stowe, VT, pp. 151-160, 1987.
2. K.L. Calvert and S.S. Lam, "Deriving a Protocol Converter : a Top-Down Dethod," Proceedings of SIGCOMM '89 symposium, Austin, TX, pp. 247-258, Sep. 1989.
3. K.L. Calvert and S.S. Lam, "Formal Methods for Protocol Conversion," IEEE Journal on Selected Areas of Communications, Vol. 8, No. 1, pp. 127-142, Jan. 1990.
4. K.L. Calvert and S.S. Lam, "Adaptors for Protocol Conversion," IEEE INFOCOM '90, pp. 552-560, June 1990.
5. P.E. Green, "Protocol Conversion," IEEE Transaction on Communications, Vol. 34, No. 3, pp. 257-268, March 1986.
6. S.S. Lam, "Protocol Conversion," IEEE Transaction on Software Engineering, Vol. 14, No. 3, pp. 353-362, March 1988.
7. K. Okumura, "A Formal Protocol Conversion Method," Proceedings of the ACM SIGCOMM '86 symposium, Vermont, pp. 30-37, August 1986.
8. K. Okumura, "Generation of Proper Adaptors and Converters from a Formal Service Specification," IEEE INFOCOM '90, pp. 564-571, June 1990.
9. Y.W. Yao and M.T. Liu, "Constructing Protocol Converters with Guaranteed Service," IEEE INFOCOM '91, pp. 960-969, April 1991.

본 논문은 한국과학재단 연구과제임 911-1106-009-1

權 雲 哲(Woon Chul Kwon) 정회원
1990년 : 광운대학교 전자계산학과 졸업(이학사)
1992년 : 광운대학교 전자계산학과 졸업(이학석사)
※주관심분야 : 컴퓨터 네트워크, 프로토콜 공학



李 東 豪(Dong Ho Lee) 정회원
1957년 1월 15일생
1979년 2월 : 서울대학교 전자공학과 졸업(공학사)
1983년 2월 : 서울대학교 컴퓨터공학과 졸업(공학석사)
1988년 2월 : 서울대학교 컴퓨터공학과 졸업(공학박사)
1984년 9월 ~ 현재 : 광운대학교 전자계산학과 부교수
※주관심분야 : 프로토콜 공학, 네트워크 성능평가

沈 泳 燮(Young Seob Shim) 정회원
1988년 : 원광대학교 컴퓨터공학과 학사
1990년 : 광운대학교 전자계산학과 석사
1993년 ~ 현재 : 광운대학교 전자계산학과 박사과정
※주관심분야 : 컴퓨터 네트워크, 프로토콜공학, 운영체제