SRL/ATM으로부터 Ada 실행코드 생성

고 현*[1], 김 광종[1], 이 연식[1], 이 문근[2]
군산대학교 컴퓨터정보과학과, 전북대학교 전자정보공학부
e-mail : kogo@cs.kunsan.ac.kr

Generation of Ada Executable Code from SRL/ATM

Hyun Ko*[1], Kwangjong Kim[1] ,Moon-Kun Lee[2], Yonsik Lee[1]
[1]Dept. of Computer Information Science, Kunsan National University
[2]Division of Electronics and Information Engineering, Chonbuk National University

요 약

본 논문은 순환공학 환경에서의 실시간 시스템 개발 및 검증을 위한 코드 생성기 구현과정에서 실시간 시스템에 대한 ATM(Abstract Timed Machine) 명세로부터 생성된 SRL(Software Representation Language) 중간코드로부터 Ada 실행코드 생성방법을 제시한다. 실시간 시스템을 명세, 분석, 검증하기 위한 정형기법인 ATM은 기존의 정형기법과는 달리 순환공학 환경에서의 실시간 시스템이 갖는 정적 및 동적 속성은 물론 특정 환경에서의 동적행위도 표현이 가능하므로, DoME/ATM 그래픽 명세 표기와 중간코드로부터 실행코드를 자동 생성함으로써 순환공학 환경에서의 실시간 시스템 개발 및 검증을 가능하게 한다. 따라서, 실행코드 자동 생성기를 구현하기 위하여 본 논문에서는 선행연구에 의한 DoME /ATM으로부터 변환된 SRL/ATM 코드로부터 Ada 실행코드를 생성하기 위하여 SRL/ATM 과 Ada의 관계를 분석하고 실행코드 생성을 위한 기본 규칙들을 정의하여, Ada 실행코드 생성기를 설계한다. 실행코드 생성기는 SRL 파스트리 생성기를 이용하여 구문분석을 통해 구문노드와 수식노드, 단말노드 등과 같은 구문적 요소들을 추출하여 어휘분석을 통해 얻어진 정보들과 추출된 구문 정보들을 기반으로 실제 Ada 실행코드를 생성한다.

1. 서론

실시간 시스템은 시스템의 논리적인 결과의 정확성에만 의존하지 않고 결과가 도출된 시간에도 많은 중요성을 부여하는 시스템으로 항공우주 산업, 국방시스템, 공정제어, 교통제어 시스템 등 우리 생활의 다양한 응용분야에 폭넓게 이용되고 있다. 이들 시스템들은 특성상 복잡하고 대규모의 시스템인 경우가 많으며, 이는 시스템의 유지보수의 어려움과 막대한 초기 시스템 개발 비용을 요구하게 되어 재사용 가능하고 신뢰성 있는 시스템을 개발하기 위한 방법이 필요하게 되었다. 따라서, 시스템을 명세하고 검증할 수 있는 정형기법을 사용하게 되었으나, 기존 정형기법들[1,2,3,4,5,6]의 경우 시스템 표현 및 개발 시적용 범위의 한계성으로 인해 새로운 정형기법이 요구되었다. ATM은 순공학과 역공학이 하나로 통합된 순환공학 환경 내에서 실시간 시스템을 명세, 분석, 검증할 수 있는 정형기법으로 많은 실시간 시스템의 속성 기술은 물론 순환공학 과정에서의 특정 환경과 동적 정보 등도 표현이 가능하다[7,8]. 이러한 ATM 정형기법을 적용하여 순환공학 환경에서 실시간 시스템을 개발하기 위해서는 명세를 위한 명세도구나 명세분석을 위한 분석기, 실행코드 생성기, 검증기 등과 같은 자동화 도구의 개발이 요구된다[7,8]. 따라서, 본 논문에서는 2 장에서 관련연구인 DoME/ATM과 매개언어인 SRL 그리고 Ada 실행코드를 생성하기 위한 정보추출 방법을 설명하고, 3 장에서는 SRL/ATM의 구조 및 제어흐름과 Ada와의 관계 및 실행코드 생성을 위한 기본 규칙들을 정의하고, SRL 분석기와 파스트리 생성기를 구현하여 SRL/ATM으로부터 실시간 시스템의 정적 및 동적 정보와 구조적 정보를 추출하여 이들의 구조적 매핑을 통한 Ada 실행코드의 생성방법을 제시한다. 마지막 4 장에서는 결론 및 향후 연구과제에 대해 기술한다.

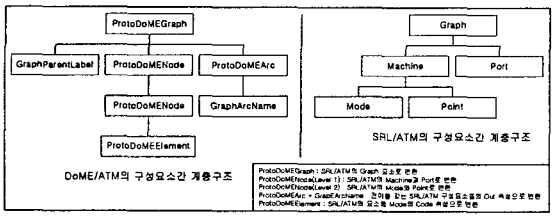
본 연구는 한국과학재단 특정기초연구(과제번호 1999-2-303-003-3) 지원으로 수행되었음

2. DoME/ATM과 SRL

2.1 DoME/ATM 스크립트 코드와 SRL/ATM의 구성요소

실시간 시스템에 대한 DoME/ATM 명세는 개발된 ATM 명세도구를 이용하여 시스템의 동적/정적 동작 및 시스템 요소들에 대한 아이템과 이들 아이템들 간의 관계로 표현된다. 그래픽컬한 DoME/ATM은 각 아이콘의 정보를 기준으로 DoME에서 제공되는 기능에 의해 자동으로 스크립트 코드(텍스트 GUI-ATM 명세)로 저장된다. 텍스트 형태의 스크립트 코드는 DoME/ATM에서의 아이템간 계층구조 및 통신에 따른 병행관계, 선후 시행관계를 표현한다. 이러한 DoME/ATM 명세로부터 실행코드를 생성하는 과정에서 불필요한 정보에 대한 정제 및 ATM 정형기법의 특성(그래픽 표기법)에 따른 비정형적 요소로 인한 명확한 정보추출이 어려우므로 중간 매개언어인 SRL의 생성이 필요하다. SRL(Software Representation Language)은 시스템을 구성하는 소프트웨어, 하드웨어, 통신망, 형상, 요구사항, 설계, 성능 등에 관한 정보를 저장소에 보관 및 관리하고 이를 재/역공학 과정에서 사용자의 특정 목적에 부합하게 표현하고 분석, 이해 및 평가를 하기 위한 메타언어이다. SRL은 실시간 시스템을 구문노드와 수식노드로 표현하며, 시스템의 중첩구조에 따라 구문노드는 형제노드와 자손노드로 표현되고, 구문의 상세 정보를 표현하기 위해 수식노드를 하위노드로 갖는 추상구문트리(Abstract Syntax Tree) 형태를 갖는다. 이러한 SRL의 생성은 DoME/ATM에서의 각 아이템 요소와 이들 요소간의 관계 정보추출을 통해 이루어진다. SRL/ATM은 DoME/ATM의 불필요한 정보를 제외한 구조 정보나 동적 및 정적 정보의 추출을 통해 명세분석 도구가 이를 기반으로 다양한 용도나 목적에 맞게 사용할 수 있도록 지원한다.

SRL/ATM에서는 DoME/ATM에서 ProtoDoMENode로 표현되었던 *Machine*, *Port*, *Mode*, *Point*를 각각 구분하여 표현하였으며, ProtoDoMEArc의 경우 이를 각 구성요소의 Out 속성으로 표현함으로써 제어의 흐름 파악에 있어 높은 이해도를 제공해 준다. 또한 각 구성요소들을 구분할 수 있는 식별정보의 추출을 통해 각 아이템들 간의 포함관계 및 상호작용 관계에 따른 명확한 계층구조를 표현한다[8]. 다음 [그림 1]은 DoME/ATM과 SRL/ATM을 구성하는 요소들 간의 계층구조를 나타낸다.



[그림 1] DoME/ATM과 SRL/ATM에서의 구성요소간 계층구조

2.2 Ada 실행코드 생성을 위한 SRL/ATM의 정보추출

SRL/ATM에서의 정보추출은 SRL/ATM의 각 구성요소로 구분된 아이템들의 구조 정보와 관계 정보, 그리고 실제 실시간 시스템 동작시 처리되는 작업의 특성이나 시스템이 갖는 환경적 요소 등과 같은 실행코드 생성에 필요한 정보들의 추출을 통해 이루어진다. SRL에서의 머신은 다른 독립된 머신과의 통신 및 이벤트 호출에 따른 병행적 시행관계와 개념적 종속관계를 형성한다. 이러한 관계정보들은 실제 다른 머신을 호출하는 호출자 머신 내부에서의 전이 이동 시 시간적 흐름에 따른 호출 정보의 추출을 통해 얻어진다. 호출 정보로는 전이 이동 시의 조건이나 시간제약 등과 같은 제약조건들과 전이 이동에 있어서의 해당 두 모드, 즉 전이를 내보내는 모드와 전이가 들어가는 목적모드에 대한 정보들이 해당된다. 이러한 정보추출 외에도 실제 실행코드에서의 실행문이나 예외를 제외한 일반적인 구문에 해당하는 모드내 Element와 같은 정적 정보추출도 필요하다.

3. SRL/ATM에 대한 Ada실행코드 생성

3.1 SRL/ATM과 Ada와의 관계

SRL/ATM은 스크립트 코드로부터의 구조 형태 및 구성요소들 간의 연관관계에 대한 정적 및 동적 정보 추출을 통해 DoME/ATM을 표현한다. SRL/ATM은 이러한 구조와 관계 정보를 이용하여 실행코드의 전체 프로그램 모듈구조 및 내부 구조 형태와 직접적으로 매핑될 수 있다. SRL/ATM 구성요소 중 Graph는 전체 실시간 시스템에서 하나의 부분적 역할을 수행하는 실제 시스템으로 나타낼 수 있고, Machine은 하드웨어 측면에서의 태스크 단위와 대응되는 것으로 실제 Ada 프로그래밍 언어에서의 프로그램을 구축하는 기본단위 프로그램인 Package, Function, Procedure, Task, Generic, Protected, Entry 등과 같은 모듈구조로 표현할 수 있다. 각 Machine은 처리작업의 특성에 따라 적합한 모듈구조로 표현할 수 있고, 따라서 각 독립된 머신들간의 활성통신 및 데이터 통신관계도 각 모듈간의 호출관계로 표현될 수 있다.

ATM에서의 각 머신간 통신과 제어흐름은 Ada 실행코드의 우선실행 순위와 관련하여 실제 작업처리 과정에 따라 Ada 코드가 실행되는 것과 같다. 이는 머신의 활성 또는 활성종료 등과 같은 동기/비동기화 통신과 대응하여 모듈의 시작과 종료가 결정되며, 모드내의 작업처리에 따른 상태변화는 순차적인 Ada 구문이나 서브블록의 실행 등으로 나타낼 수 있다. 작업처리에 있어 정해진 시간 범위에 따른 제약이나 다음 상태로의 전환을 위한 조건만족, 조건에 따른 선택적 상태로의 전환 등은 각각 Ada 구문의 서브블록에 해당하는 반복구문, 조건구문, 선택구문 등으로 변환되어 진다. 또한 이러한 Ada 구문들의 코드생성 시의 생성 위치는 SRL의 구성요소들이 갖는 속성에 따라 결정되거나 구성요소의 실제 속성값의 구문분석을 통해 생성위치가 결정되어진다.

3.2 SRL/ATM에 대한 Ada 실행코드 생성을 위한 기본규칙

SRL/ATM으로부터 추출된 구조정보 및 동적 및 정적 정보를 이용하여 구조적 매핑 방법을 통한 Ada 실행코드로의 변환에 있어 실제 추출된 구조정보만으로 Ada의 모듈구조로 매핑시키는 것은 매우 어렵다. 이는 각 모듈이 작업의 특성에 따라 결정되어짐으로써 DoME/ATM 명세 뿐만 아니라 여기서 추출된 정보를 이용해 표현한 SRL/ATM에서도 각 머신이 처리하는 작업의 특성을 파악하는 것은 매우 힘들기 때문이다. 따라서 구조적 매핑을 통한 Ada로의 변환을 위해서는 다음과 같은 조건을 만족하도록 하여 실제 SRL/ATM에 대한 Ada 코드변환기에서 외부적으로 결정사항을 입력 해주어야 한다.

조건 1 . ATM 명세에서 하나 또는 다수의 머신이 존재할 경우, 각 머신 마다 처리 작업의 특성에 맞는 모듈 구조를 가진 기본적인 프로그램 단위를 결정해주어야 한다.

또한 구조적 매핑 방법을 통해 SRL/ATM에 대한 Ada 실행코드 생성을 위해서는 다음과 같은 정의 사항들을 만족해야 한다.

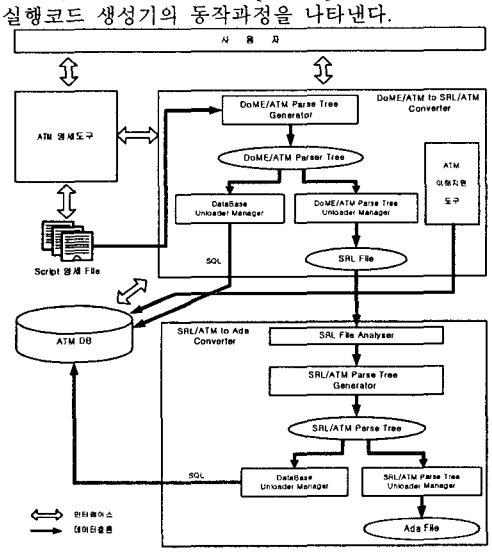
정의 **1**. 각 *Machine* 은 반드시 하나 이상의 Port 와 PortConnector 를 가진다.
정의 **2**. 각 Machine 은 반드시 하나만의 StartPoint 와 TerminationPoint 를 가진다.
정의 **3**. 각 Mode 는 반드시 하나 이상의 In 전이와 Out 전이 가진다.
정의 **4**. SRL/ATM 에서의 Machine 간 동기/비동기화 활성통신에 있어 호출 태스크와 피호출 태스크의 관계에 따라 Ada 모듈 구조간의 종속관계가 성립된다.
정의 **5**. SRL/ATM 에서의 Machine 간 동기/비동기화 활성통신 순서에 따라 Ada 모듈 구조의 생성 위치가 결정된다.
정의 **6**. SRL 코드 상의 전이에 의한 Loop 구조는 해당 실행코드 내 반복 구문의 규칙을 적용하여 표현한다.
정의 **7**. SRL 코드 상의 전이에 의한 분기 구조는 해당 실행코드 내 선택 구문 또는 분기 구문의 규칙을 적용하여 표현한다 (단 Task 의 활성을 위한 동기/비동기화 호출 제외).
정의 **8**. SRL 코드 상의 전이에 의한 분기 구조에 대한 Ada 코드의 선택조건 구문은 SRL의 Condition과 Constraint 속성을 함께 사용하거나 또는 Communication과 Constraint 속성을 함께 사용해 표현한다 (단, 선택 조건문에서 *Communication*과*Condition*은 단독 사용가능하며, Constraint 속성은 단독으로 사용할 수 없다).

3.3 SRL/ATM에 대한 Ada 실행코드 생성기 설계

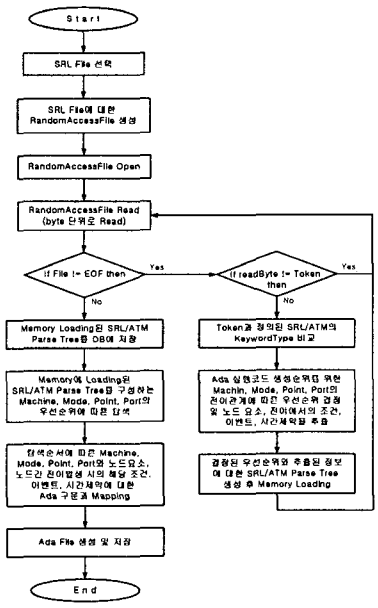
DoME/ATM으로부터 Ada실행코드를 생성하기 위한 전체 시스템의 구조는 상위부분(DoME/ATM to SRL/ATM Converter)과 하위부분(SRL/ATM to Ada Executable Code Generator)으로 구성되며 다음 [그림 2]와 같다.

SRL/ATM에서 Ada 실행코드의 생성과정은 SRL/ATM 코드 변환기에서 생성된 SRL 파일을 입력받아 SRL/ATM 분석기를 통해 구성요소들의 구조 및 관계 정보를 추출한다. 또한, 제어 흐름에 따른 시간제약, 조건에 대한 속성 정보를 추출하고 어휘 분석을 통한 통신 이벤트나 모드 간의 전이 관계를 추적하여 실행코드의 생성 위치 및 노드의 탐색순위를 결정하게 된다. 이렇게 추출된 정보는 SRL/ATM 파스트리 생성기를 통해 파스트리를 생성한 후, SRL/ATM 파스트

리 Unloader를 통해 탐색 우선순위에 따라 트리를 구성하는 노드들을 탐색하여 각 해당 Ada 구문을 생성하게 된다. 또한, SRL/ATM 파스트리로부터 데이터베이스 Unloader를 통해 추출된 정보와 우선순위를 ATM 데이터베이스에 저장한다. Ada 실행코드의 생성위치는 통신이나 모드 간 전이에 대한 관계정보와 요소들의 속성에 의해 결정되어 해당 위치에 실행코드가 생성되어 진다. 다음 [그림 3]은 SRL/ATM에 대한 Ada 실행코드 생성기의 동작과정을 나타낸다.



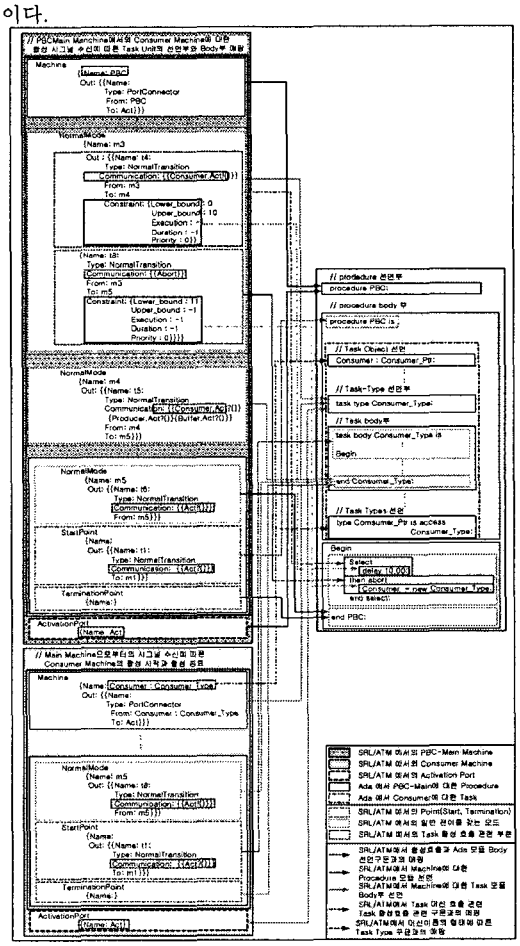
[그림 2] 실행코드 생성 시스템 구성도



[그림 3] SRL/ATM에 대한 Ada 코드 생성기의 동작 흐름도

3.4 SRL/ATM에 대한 Ada 실행코드 생성 예

SRL/ATM에서 Ada 실행코드 생성 시 정의 1 과 정의 2 에서 명시한 바와 같이 머신은 반드시 하나의 Port와 PortConnector, StartPoint와 TerminationPoint를 가져야 한다. 이는 실제 머신간의 통신과 제어 흐름을 위해 반드시 존재해야 하는 요소들이다. SRL 코드에서의 Machine과 Port 관련부분은 모듈 정의 코드와 매핑될 수 있고, StartPoint와 TerminationPoint는 모듈 Body부의 정의로 표현될 수 있다. 또한, SRL/ATM 각 머신들은 동기/비동기화 활성 통신을 통한 개념적 종속관계를 형성하게 됨으로써 실제 Ada 실행코드 생성 시 임의의 한 머신으로부터의 다른 독립적으로 존재하는 머신을 활성시키는 경우, 호출자 태스크인 Ada 모듈 내부에 피호출자 태스크인 모듈이 생성되어 진다. 또한, 동기/비동기화 활성 통신에서 Ada 모듈 중 하나인 Task 모듈에 해당하는 머신을 활성시키는 경우, 호출자 태스크 즉, 메인 머신에서의 모드간 전이이동 시에 발생하는 활성호출은 피호출자 머신에서의 활성 시그널 수신 여부에 따라 활성과 활성취소로의 선택적 전이 이동으로 표현된다. 이 때, 이러한 형태를 Ada 구문으로의 변환함에 있어 Task 모듈의 활성호출과 관련된 구문적 규칙인 Select ... then abort ... end Select 구문을 적용을 하게 된다. 다음 [그림 4]는 동기/비동기화 활성통신에 따른 호출 태스크와 비호출 태스크 간의 종속관계 및 활성과 활성취소에 따른 구문적 규칙을 적용한 예이다.



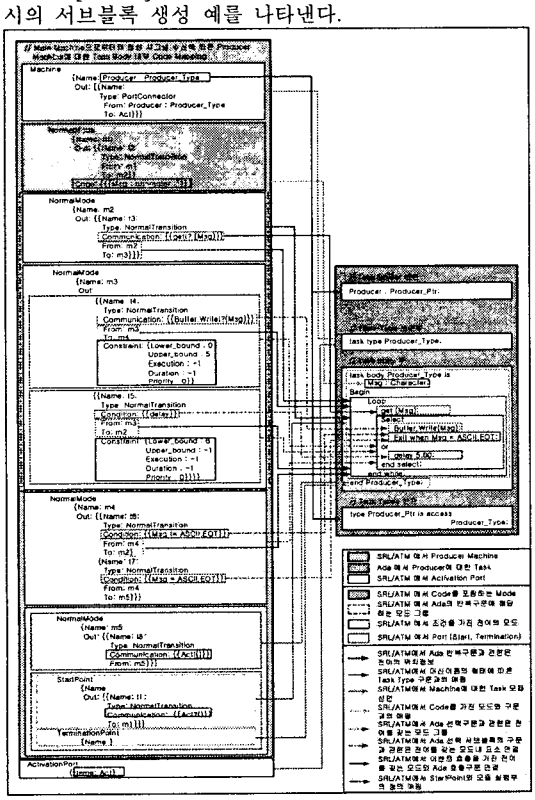
[그림 4] 동기/비동기화 활성에 따른 종속관계 및 Task 활성에 대한 구문규칙 적용 예

SRL의 모드는 다음 상태로의 모드 이동을 통해 계속적인 작업처리를 수행한다. 모드에서 다음 모드로의 연결은 전이에 의해 이루어지며, 전이는 때에 따라 이벤트 호출이나 시간제약, 조건 등을 갖는다. 머신에서의 제어흐름은 이러한

제약사항들에 따라 현재 모드에서 이미 지나쳐온 모드로 다시 되돌아가 반복적인 작업을 수행하기도 하며, 다음 모드로의 계속적 진행을 위해 주어진 제약사항을 만족함으로써 결과적으로 작업의 종료점에 도달할 수 있다. 이러한 구조는 실행코드의 반복구문 구조와 동일하다. 따라서 이러한 전이 이동에 따른 경우에는 해당 모드간 이동에 대해 반복구문의 규칙을 적용함으로써 코드 변환을 수행할 수 있다. 또한 한 모드가 두 개의 모드로 선택적으로 분기되는 경우에는 Ada에서의 조건구문이나 선택구문을 규칙을 적용할 수 있다. 이때, Ada 실행코드의 반복구문이나 조건구문, 선택구문과 대응되는 모드간 연결관계는 각각 조건이나 시간제약, 또는 조건에 대한 이벤트 호출 등과 관련된 레이블을 갖는 전이가 반드시 존재해야 한다.

SRL에서의 Code 는 Code 속성값이 일반적인 실행문이나 변수 선언과 같은 내용을 포함할 경우, Ada 해당 모듈에서 명세부, 즉 선언부분에 위치할 수 있지만, 포함내용이 수식계산을 위한 구문을 나타내는 경우에는 이를 모듈의 Body 구현부에 생성하게 된다.

다음 [그림 5]는 SRL/ATM에 대한 Ada 실행코드로의 변환시의 서브블록 생성 예를 나타낸다.

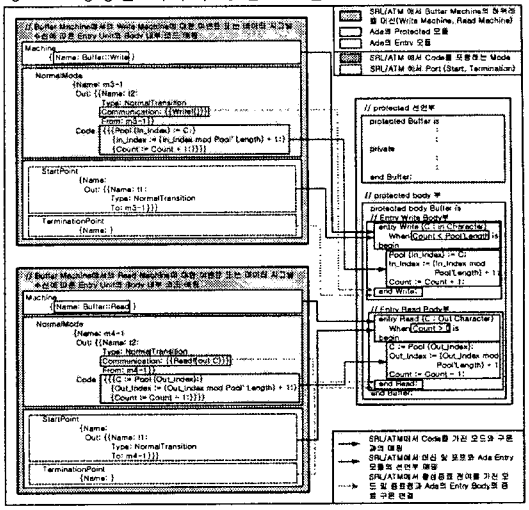


[그림 5] SRL/ATM의 제어흐름에 따른 Ada 실행코드의 서브블록 생성 예

다음 [그림 6]은 SRL/ATM에서 Code의 특성에 의해 Ada 코드 생성위치가 결정되어지는 코드 생성 예이다.

4. 결론 및 향후연구 과제

본 논문은 순환공학 환경에서의 실시간 시스템 개발 및 검증을 위한 실행코드 생성 시스템 구현과정에서 실시간 시스템에 대한 ATM 명세로부터 변환된 SRL/ATM에 대응하는 Ada 실행코드를 생성하는 방법을 제시하였다. 순환공학 환경을 위한 ATM 코드 생성 시스템은 실시간 시스템에 대한 DoME/ATM 명세로부터 분석기에 의한 명세분석을 통해 추출된 정보를 ATM DB에 저장하여 ATM 이해 지원 도구와 같은 분석도구에 제공함으로써 다양한 명세분석을 용이하게 지원할 수 있다. 또한, 추출된 정보를 이용하여 효율적인 실행코드 생성을 위하여 중간코드인 SRL 코드로 변환하였고,



[그림 6] SRL에서의 Code 특성에 따른 Ada 모듈구조 내에서의 코드 생성 예

SRL/ATM 분석기를 이용하여 각 구성요소의 속성 및 요소간 구조 정보, 노드 탐색 순위, 코드 생성 위치 등의 정보 추출을 통해 시스템 개발자의 요구에 맞는 다양한 실행코드를 생성할 수 있다.

향후 과제로는 본 논문에서 제시한 실행코드 생성규칙을 기반으로한 순공학 과정에서의 요구명세로부터 효율적 실행코드를 생성하기 위한 Ada 실행코드 생성 시스템의 구현이 요구되며, 추후에는 실행코드 생성을 위해 제시되었던 제약조건을 완벽히 만족시킴으로써 다양한 실행코드를 자동으로 생성할 수 있는 자동화 언어 생성 시스템의 개발에 관한 연구가 지속되어야 한다. 또한 역공학 측면의 검증을 위한 구현 언어로부터 ATM과 같은 명세도구로의 역변환시스템 개발과 실제 개발된 ATM과 같은 정형기법들을 시스템 개발에 효율적으로 적용하기 위한 다양한 자동화 도구들의 개발이 요구된다.

참고문헌

[1] A. Shaw, "*Communicating Real-Time State Machines*," IEEE Transactions on Software Engineering, Vol. 18, No. 9, pp. 805-816, September, 1992

[2] C. A. R. Hoare, *Communicating Sequential Processes*, Prentice-Hall, 1985

[3] D. Harel, "*Statecharts: A Visual Formalism for Complex System, Science of Computer Programming*," Vol. 8, pp. 231-274, 1987

[4] S. jahanian and A. Mok, *Modechart : A Specification Language for Real Time Systems*, IBM Technical Report: RC 15140, November, 1989

[5] Sitaram C. V. Raju, *An Automatic Verification Technique for Communicating Real-Time State Machines*, Dept.of CS&E at University of Washington, TR 93-04-08, 1993

[6] Stuart Bennett, *Real-Time Computer Control – An introduction*, Prenticce Hall, 1994

[7] 노경주, 박지연, 이문근, "실시간 시스템의 순환공학을 위한 정형기법 : 추상시간기계," 한국정보과학회 학술발표논문집(A), 제 27 권, 제 1 호, pp. 477-479, 2000

[8] 고현, 조상규, 이연식, "순환공학 환경에서의 실시간 시스템 개발 및 검증을 위한 코드 변환기 설계," 한국정보처리학회 학술발표논문집(상), 제 8 권, 제 1 호, pp. 193-196, 2001