공학 연구논문집 제27권 제2호 pp. 33~44, 울산대학교
Journal of Enginering Research Vol. 27, No. 2, pp. 33~44, 1996
University of Ulsan.

확장 방향성 그래프를 이용한 실시간 병렬 태스크의 단계적인 구성

이종구 · 김규년
전자계산학과

<요 약>

실시간 소프트웨어 설계 시에는 시스템을 분석하고 이것을 병렬 태스크로 구성하게 된다. 시스템 분석의 결과로 control flow and data flow diagram(이하 C&DFD)이 산출물로 나오게 되고 이들을 이용하여 성능에 문제가 생기지 않을 조건하에서 병렬 태스크로 구성하게 된다. 이러한 병렬 태스크들의 구성 방법이 Gomaa에 의해 COncurrent Design Approach for Real-Time System(이하 CODARTS)방법론으로 제시되어 있다. 그러나 CODARTS에서 제시되는 병렬 태스크 구성 지침은 설계자의 경험에 의존적이기 때문에 이를 이용하여 시스템을 설계하기가 어려운 것이 현실이다. 본 연구는 CODARTS에서의 태스크 구성지침을 보다 정교하게 개선하고 방향성 그래프를 확장하여, 설계자의 경험에만 의존하지 않고도 일반적으로 적용 가능한 실시간 병렬 태스크 구성 지침을 제안한다.

A Stepwise Structuring of Real-Time Concurrent Task Using Extended Directed Graph

Lee, Jong-Gu · Kim, Kyoo-Nyun
Dept. of Computer Science

<Abstract>

When we design real-time software the target system is analyzed and then structured to concurrent tasks. As a result of the analysis, control flow and data flow

diagram(C&DFD) is produced. This diagram is structured into concurrent tasks under the condition that performance problem does not rise. The method that makes concurrent tasks is introduced as COncurrent Design Approach for Real-Time System(CODARTS) method by Gomaa, but designing systems using CODARTS is somewhat difficult because its rule is dependent on the designer's experience. In this study, we propose a generally well-adaptable guideline for structuring real-time concurrent tasks using extended directed graph. This guideline can be used without depending on the designer's experience.

1. 서 론

본 연구에서는 실시간 소프트웨어 설계 방법론에서 사용되는 병렬 태스크를 단계적으로 구성하는 지침을 제시하고 이에 대한 예를 보이고자 한다. Gomaa의 방법론에서는 우선 Real-Time Structured Analysis(이하 RTSA[2]) 또는 RTSA의 확장인 Concurrent Object-Based Real-Time Analysis(이하 COBRA)를 사용하여 목표 시스템을 부 시스템으로 분해한 C&DFD를 얻는다[1]. 여기서 분해된 각 부 시스템들은 모두 병렬적으로 실행할 가능성들을 가지고 있다. 그러나, 분해된 많은 부 시스템들이 모두 병렬 태스크로 구성된다면, 설계 단계에서는 좋으나 전체 시스템 구현 단계에서 상당량의 부하를 가지게 되어 오히려 시스템의 성능문제에 부딪히게 될 가능성을 가지게 된다. 따라서, 이러한 부하를 극복할 수 있도록 두 개이상의 부 시스템을 묶을 수 있는 지침이 있어야 된다. 예를 들어 어떤 부 시스템들이 순서적으로 실행된다면 이들은 굳이 각각 병렬 태스크로 구성 할 필요가 없다. Gomaa의 방법론에서는 CODARTS방법론을 사용하여 이러한 문제를 극복하고 있다[1]. CODARTS는 RTSA또는 COBRA의 결과물인 C&DFD의 부 시스템을 몇 가지 구성 지침에 의해 태스크로 구성한다. 이러한 구성된 태스크들을 표현한 다이어그램을 Task Architecture Diagram(이하 TAD)라고 한다. CODARTS방법론의 병렬 태스크 구성 지침의 많은 부분은 경험적이어서 이를 실제 적용하는데 있어서 경험이 있는 설계자의 감각이 절대적으로 필요하게 되어 자의적인 구성이 이루어질 가능성이 높아진다. 따라서, 본 연구에서는 경험적인 특성이 제거된 보다 객관적으로 적용 가능한 실시간 병렬 태스크의 구성 지침을 제안한다.

2. CODARTS에서의 병렬 태스크 구성 지침

병렬 시스템에서 병렬 태스크의 수가 많아지면 태스크간의 동기화와 통신에 대한 추가 부하가 발생되기 때문에 시스템의 설계자는 성능을 고려하여 병렬 태스크의 수를 조절하며 설계를 수행해야 한다. Gomaa에의해 제시된 CODARTS는 아래에 제시된 몇 가지 지침에 의거하여 태스크의 수를 줄여나간다[1]. 아래에는 CODARTS의 병렬 태스크를 구성

하는 지침을 소개한다. 2.1절에서는 I/O와 관련된 병렬 태스크 구성 지침을 2.2절에서는 내부 병렬 태스크 구성 지침을 2.3절에서는 응집도에 기반을 둔 병렬 태스크 구성 지침을 2.4절에서는 우선 순위에 기반을 둔 병렬 태스크 구성 지침을 기술한다.

2.1. I/O와 관련된 병렬 태스크 구성 지침

I/O 와 관련된 병렬 태스크 구성 지침으로는 비동기 장치관련 I/O 태스크 지침, 주기적 I/O 태스크 지침, 자원 감독 태스크 지침이 있으며 I/O관련 각 부 시스템의 특성을 고려하여 구성한다. 비동기 장치관련 I/O 태스크 지침은 비동기 장치에 의해 활성화되는 부 시스템을 태스크로 구성하는데 적용되는 지침으로 각 부 시스템은 하나씩의 태스크가 할당된다. 주기적 I/O 태스크 지침은 타이머에 의해 활성화되는 부 시스템을 태스크로 구성하는데 적용되는 지침으로 각 부 시스템은 하나씩의 태스크가 할당된다. 일반적으로 센서를 샘플링하는 I/O 부 시스템에 적용된다. 자원 감독 태스크 지침은 자료를 소비하는 장치와 연결된 부 시스템이 복수개의 타 부 시스템에 의해 활성화될 때 장치와 연결된 부 시스템을 태스크로 구성하는데 적용된다. 이 부 시스템은 상이한 요구를 조절할 수 있어야 하기 때문이다.

2.2. 내부 병렬 태스크 구성 지침

내부 병렬 태스크 구성 지침은 I/O와 관련된 부 시스템을 제외한 모든 부 시스템을 구성하는데 적용된다. 세부 지침으로는 주기 태스크 지침, 비동기 태스크 지침, 제어 태스크 지침, 사용자 역할 태스크 지침이 있다. 주기 태스크 지침은 주기적으로 수행하는 부 시스템을 태스크로 구성하는데 적용된다. 때때로 이러한 주기적 성격을 가지는 부 시스템을 묶어서 하나의 태스크로 구성하기도 한다. 비동기 태스크 지침은 다른 내부 부 시스템에 의해 활성화되는 부 시스템을 태스크로 구성하는데 적용된다. 제어 태스크 지침은 제어 부 시스템을 태스크로 구성하는데 적용된다. 사용자 역할 태스크 지침은 키보드와 같은 입력 장치 또는 터미널과 같은 출력 장치와 연관되어 있는 태스크를 구성하는데 적용된다.

2.3. 응집도에 기반을 둔 병렬 태스크 구성 지침

모듈의 응집도 측정기준과 함수의 응집도 측정기준이 이미 소개되어 있고 이러한 응집도의 일반적인 속성 정의도 이미 소개되어 있다[3][4][5][6]. CODARTS에서는 이러한 응집도의 기본 개념을 이용하여 부시스템을 단일 태스크로 구성한다. 응집도 지침으로서는 시간 응집도 지침, 순차 응집도 지침, 제어 응집도 지침, 함수 응집도 지침이 있다. 시간 응집도 지침은 타이머의 발생 간격이 동일한 부 시스템이 실행 순서에 무관하게 수행될 수 있을 때 하나의 태스크로 구성시키는데 적용된다. 순서 응집도 지침은 여러 개의 부 시스템이 병렬적으로 수행할 수 없고 순차적으로만 수행될 수 있는 경우 이러한 부 시스템을 하나의 태스크로 구성하는데 적용된다. 제어 응집도 지침은 상태 전이를 수행하는 제어 부 시스템과 이 제어 부 시스템에 의해 활성화되는 부 시스템들을 하나의 태스크로 구성하는데 적용된다. 함수 응집도 지침은 응집도 지침 중에서 가장 약한 형태의 지침으

로 기능이 유사한 부 시스템을 구성하는데 적용된다.

2.4. 우선 순위에 기반을 둔 병렬 태스크 구성 지침

일반적으로 태스크의 우선 순위는 구현 단계에서 고려되지만 활성 시간이 중요한 변수인 경우에는 태스크 구성 단계에서 고려된다. 이때 적용되는 구성 지침이다.

3. 병렬 태스크 구성을 위한 C&DFD의 기본 정의

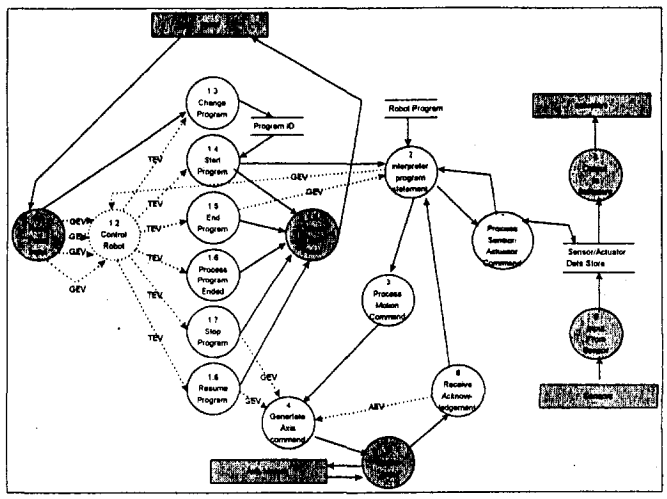


그림 1. 로봇 제어 시스템의 C&DFD

[그림 1]은 로봇 제어 시스템을 RTSA또는 COBRA지침을 적용하여 만든 C&DFD이다 [1]. C&DFD의 구조는 다음과 같이 정의한다.

S(C&DFD) = { TR, EV, MSG }
TR = { CTR, DTR, Source_DevTR, Sink_DevTR }
EV = { TEV, AEV, GEV, TmEV }
S : C&DFD가 표현하는 전체 시스템
TR : 부 시스템 [원]
EV : 이벤트 [점선 화살표]
MSG : 자료 흐름 [실선 화살표]
CTR : 제어 부 시스템 [전선 원]
DTR : 자료 부 시스템 [실선 원]
Source_DevTR : 자료를 발생시키는 장치 부 시스템 [흑색 실선 원]
Sink_DevTR : 자료를 소비하는 장치 부 시스템 [흑색 실선 원]
TEV : 트리거 이벤트 [TEV를 레이블로 가지는 점선 화살표]

AEV : 응답 이벤트 [AEV를 레이블로 가지는 점선 화살표]
GEV : 목적 이벤트 [GEV를 레이블로 가지는 점선 화살표]
TmEV : 타이머 이벤트 [번개표시 점선 화살표]

위의 정의를 사용하여 4장에서는 태스크 구성을 위한 단계적 적용 지침에 대해 설명한다.

4. 병렬 태스크 구성 지침

여기서 소개되는 단계적 병렬 태스크 구성 지침은 CODARTS에서 소개된 구성 지침을 정제하고 확장시킨 것으로 다단계 적용을 통해 부 시스템을 태스크로 구성할 수 있다는 특징을 가지고 있다. 5.1와 5.2를 제외한 모든 단계는 부 시스템 관계 테이블을 참조하여 태스크를 구성하고, 5.1와 5.2는 확장 방향성 그래프와 이를 사용하여 만든 인접 매트릭스 테이블을 사용하여 태스크를 구성한다. 2, 3에서는 I/O관련 태스크를 구성하고, 4, 5에서는 비동기 태스크를, 6-8.1에서는 제어 태스크를 구성하며, 8.2에서는 제어 태스크와 병렬적으로 수행 가능한 태스크를 구성한다.

아래에는 병렬 태스크 구성 지침의 적용 단계를 보여주고 있다. []괄호로 표시된 문장은 CODARTS에서 적용되는 구성 지침을 나타낸다.

1. 부 시스템 관계 테이블을 구성한다. (부 시스템 관계 테이블은 각 TR간의 관계(자료, 메시지 등)를 설명하는 표로서 TR의 속성을 나타낸다. 따라서, 이하 단계에서 판단의 근거로 사용된다.)
2. 테이블을 이용하여 I/O와 관련되는 TR를 추출한다. (부 시스템 관계 테이블을 참조하여 마킹이 DEV인 모든 부 시스템을 I/O 관련 TR로 추출한다. 추출된 TR들은 3을 거쳐 병렬 태스크로 구성된다.)
3. 추출된 TR을 태스크로 구성한다.
   3.1. 출력이 MSG인 Source_DTR을 태스크로 선택 [비동기 장치 관련 I/O 태스크 지침 적용] (자료를 발생시키는 장치와 연결되어 있는 TR이 MSG를 발생시킬 때 이 TR은 비동기 장치 관련 I/O 태스크로 구성된다.)
   3.2. 입. 출력 MSG중 하나가 없는 TR을 태스크로 선택 [주기 I/O 태스크 지침 적용] (이 지침에 해당되는 TR은 분명히 하나 이상의 자료 저장소(data store)와 연결되어 있고, 타이머에 의해 자료 저장소와 상호작용을 하기 때문에 주기 I/O 태스크로 구성된다.)
   3.3. 상호작용 하는 DTR이 복수개 인 Sink_DevTR을 태스크로 선택 [자원 감독 I/O 태스크 지침 적용] (복수개의 DTR로부터 MSG를 전송 받아 이를 처리해야하기 때문에 이러한 것을 감독해야한다. 따라서 자원 감독 I/O 태스크로 구성된다.)
   3.4. 상호작용 하는 DTR이 단수개 인 Sink_DevTR은 호출 DTR에 결합 [제어 응집도 지침 적용] (이 TR은 항상 수동적인 역할을 수행하기 때문에 이 TR을 호출하는 DTR에 포함시킨다.)

4. 입출력 MSG가 모두 존재하는 DTR을 추출한다. (부 시스템 관계 테이블을 참조하여 입력과 출력이 모두 MSG인 DTR을 추출한다. 이 DTR은 5를 거쳐 병렬 태스크로 구성된다.)
5. 추출된 DTR을 응집도 지침에 의거하여 태스크로 구성한다. (확장 방향성 그래프와 인접 매트릭스 테이블을 이용하여 5.2와 5.3을 처리한다.)
   5.1. 함수 응집도 지침을 적용하여 태스크로 구성 [함수 응집도 지침] (5.3절에서 자세히 설명된다.)
   5.2. 순차 응집도 지침을 적용하여 태스크로 구성 [순차 응집도 지침] (5.4절에서 자세히 설명된다.)
   5.3. 시간 응집도 지침을 적용하여 태스크로 구성 [시간 응집도 지침] (타이머에 의해 활성화되고 그 빈도가 유사하다면 몇 개의 DTR을 묶어 하나의 병렬 태스크로 구성한다. 이때 부 시스템 관계 테이블을 참조한다.)
6. CTR을 태스크로 선택. [제어 응집도 지침 적용] (부 시스템 관계 테이블을 참조하여 TR의 형이 1(CTR)인 부 시스템을 병렬 태스크로 구성한다.)
7. 나머지 DTR을 추출한다. (부 시스템 관계 테이블 중에서 아직 태스크로 구성되지 못한 부 시스템을 추출한다. 이 부 시스템은 CTR에 의해 호출되는 것들이다.)
8. 추출된 DTR을 태스크로 구성한다.
   8.1. TEV에 의해 호출되는 DTR을 CTR에 결합 [제어 응집도 지침 적용] (활성화 순간이 TEV에 의해서 결정되는 DTR은 수동적인 면을 가지고 있기 때문에 CTR에 포함시킨다.)
   8.2. GEV에 의해 호출되는 DTR은 따로 모아 태스크로 구성 [함수 응집도 지침 적용] (GEV에 의해 호출되는 DTR은 활성화 순간이 CTR과 병렬적이기 때문에 따로 묶어서 태스크로 구성한다.)

위의 여러 단계 중에서 5는 CODARTS의 병렬 태스크 구성 지침 중에서 응집도에 기반을 둔 구성 지침과 대응된다. 그러나 단계적 병렬 태스크 구성 지침의 5는 응집도 지침을 구체적으로 처리할 수 있도록 지원하여 CODARTS의 태스크 구조화 추상성을 배제 시켰다. 이 5의 처리과정은 다음장에서 자세히 설명한다.

5. 단계적 병렬 태스크 구성의 5단계 지침

이 단계에서는 방향성 그래프와 인접 매트릭스 테이블을 사용한다. 이것은 응집도에 의한 태스크 구성을 구체화할 수 있도록 해준다. 5.1절에서는 확장 방향성 그래프를 추출하는 지침을, 5.2절에서는 인접 매트릭스 테이블 구성하는 지침을 설명하고, 5.3절과 5.4절에서는 이러한 것을 이용한 함수, 순차 응집도 지침을 설명한다.

5.1. 확장 방향성 그래프를 추출한다.

4에서 구해진 DTR과 그것들의 관계를 이용하여 확장 방향성 그래프를 구성한다. 확장 방향성 그래프의 구조는 다음과 같다.

G = {V, E}

E = {msg_edge, tm_edge, ack_edge, sink_edge}

G는 확장 방향성 그래프이고, V는 vertex를 E는 edge를 나타낸다. E의 요소로는 MSG의 전송을 나타내는 msg_edge와 타이머 이벤트의 전송을 나타내는 tm_edge와 응답 이벤트의 전송을 나타내는 ack_edge 그리고 MSG를 소비하는 자료 소비장치 TR과 관련된 sink_edge로 구성된다. 이 확장 방향성 그래프는 edge의 속성이 다양하고, 표시되지 않는 외부장치와 연결되는 tm_edge와 sink_edge를 가지고 있다. 표기법은 [표 1]과 같다.

요소	V	msg_ edge	tm_ edge	ack_ edge	sink_ edge
표기법	고유 번호	→		ACK	→ Sink #

[표 1] 확장 방향성 그래프에서 사용되는 표기법
(#은 Dev_TR의 고유 번호를 의미한다.)

[그림 2]의 왼쪽 그래프는 4에서 구해진 DTR들의 C&DFD이고 오른쪽 그래프는 C&DFD를 확장 방향성 그래프로 표현한 것이다. 그러나, 4에서 구해지는 DTR을 대상으로 하기 때문에, MSG흐름 내부에 다른 TR(예를 들어, I/O관련 Dev_TR)이 존재할 경우, MSG흐름은 단절되게되어 MSG흐름의 의미를 상실하게 된다. 이러한 것을 극복하기 위해서는 [그림 2]의 오른쪽 그림과 같이 단절선을 연결해 주어야 한다. 그리고 자료를 소비하는 DevTR은 sink_edge의 표기법에 따라 확장 방향성 그래프에 표기된다. 이러한 방향성 그래프가 만들어 졌다면, 인접 매트릭스 테이블을 구성할 수 있다.

5.2. 인접 매트릭스 테이블

열은 MSG또는 EV를 생성하는 쪽의 DTR 과 타이머 이벤트이고 행은 이러한 것을 소비하는 쪽의 DTR 과 자료 소비 장치로서 DTR간의 상호관계를 설명하는 표이다. 마킹 0은 아무런 관계가 없음을 나타내고, 마킹 1은 MSG를 주고받음을, 마킹 2는 AEV를 주고받음을 나타내고 마킹 3은 외부장치와의 관계를 나타낸다. 특히 첨자가 0인 것(3(0))은 타이머 이벤트가 발생됨을 나타내고, 첨자가 0이 아닌 것(3(5))은 MSG를 소비하는 고유번호가 5인 Sink_DevTR과 관계를 나타낸다. [표 2]는 [그림 2]의 오른쪽 그래프의 인접 매트릭스를 표현한 것이다.

SND \ RCV	1	2	3	4	5	7	sink
1	0	1	0	0	0	0	0
2	0	0	1	0	0	0	0
3	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0
7	0	0	0	1	2	0	0
Timer Event	3(0)	0	0	0	0	0	0

[표2] 그림2의 오른쪽 그래프의 인접 매트릭표스 표현

[표 2]의 인접 매트릭스 테이블을 이용하여 5.1와 5.2를 구체적으로 처리할 수 있다. 우선 5.1인 함수 응집도 지침을 고려하여 보고, 5.2인 순차 응집도 지침을 고려해보자.

5.3. 함수 응집도 지침을 이용한 태스크구성

함수 응집도 지침도 두 가지로 나누어 고려해야 한다. 기본자료로 [그림 3]과 [표 3]을 이용한다.

1. 순차적인 함수 응집도 지침에 의해 태스크를 구성한다.
1.1. Sink행을 제외한 열중에서 마킹 1의 개수가 2이상인 열의 SND DTR을 구한다.
1.2. 구해진 DTR의 RCV DTR들이 병렬적인 수행이 불가능하다면 SND DTR과 RCV DTR은 하나의 태스크로 구성 될 수 있다.
예) 마킹 1의 개수가 2이상인 SND DTR은 1 DTR이고, 그것의 RCV DTR인 2,3,4 DTR은 병렬 실행이 불가능하다. 1, 2, 3, 4 DTR는 하나의 태스크로 구성된다.

SND \ RCV	1	2	3	4	5	6	7	Sink
1	0	1	1	1	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	3(8)
6	0	0	0	0	0	0	0	3(8)
7	0	0	0	0	0	0	0	3(8)
Timer Event	0	0	0	0	0	0	0	0

[표 3] 그림3의 오른쪽 그래프의 인접 매트릭스 표현

2. 클러스터링된 함수 응집도 지침에 의해 태스크로 구성한다.
2.1. Sink행의 마킹이 3인 것들을 모은다.
2.2. 마킹의 첨자가 동일한 것들을 모아서 하나의 태스크로 구성 될 수 있다.

예) 마킹 3[4]가 같은 1,2,3 DTR을 모아서 하나의 태스크로 구성된다.

5.4. 순차 응집도 지침을 이용한 태스크구성

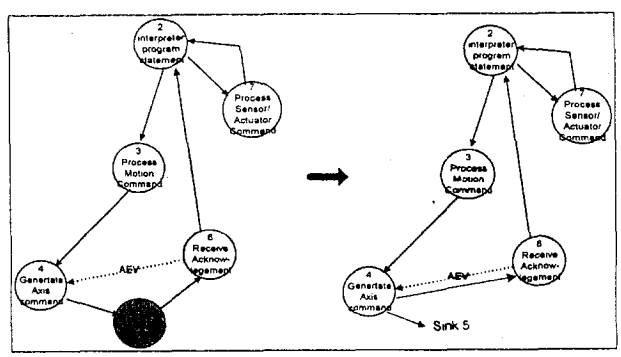
기본 자료로 [그림 2]와 [표 2]를 이용한다. 순차 응집도 지침은 두 가지로 나누어 고려해야 한다.

1. 응답 이벤트에 의해 동작하는 순차성 부 시스템의 집합을 태스크로 구성한다.
   1.1. 마킹이 2인 부분의 전송DTR와 수신DTR을 구한다.
   1.2. 수신DTR로부터 전송DTR까지 1로 마킹된 인접DTR을 찾아낸다.
   예) 마킹이 2인 부분의 전송DTR은 7 DTR이고, 수신DTR은 5DTR이다. 5,7 DTR이 하나의 태스크로 구성된다.
2. 타이머 이벤트에 의해 동작하는 순차성 부 시스템의 집합을 태스크로 구성한다.
   2.1. 마킹이 3(0)인 부분의 수신DTR을 구한다.
   2.2. 수신DTR로부터 인접DTR이 없을 때까지 1로 마킹된 인접DTR을 찾아낸다.
   예) 마킹이 3(0)인 수신DTR은 1 DTR이다. 1,2,3 DTR이 하나의 태스크로 구성된다.

6. 단계적 병렬 태스크 구성 지침을 적용한 로봇 제어 시스템[1]의 TAD

단계적 병렬 태스크 구성 지침을 로봇 제어 시스템의 설계에 적용해보자. [그림 1]의 C&DFD를 바탕으로 하여 부 시스템 관계 테이블을 작성한다. [표 4]는 로봇 제어 시스템의 부 시스템 관계 테이블이다.

3.1에 의해 1.1, 5 TR은 비동기 장치관련 I/O 태스크로 구성되고, 3.2에 의해 8, 9 TR은 주기 I/O 태스크로 구성되고, 3.3에 의해 1.9 TR은 자원 감독 I/O 태스크로 구성된다 5.1에 의해 2, 3, 7, TR은(함수 응집도 지침) 태스크로 구성된다. 5.2에 의해 4, 6 TR은(순차 응집도 지침) 태스크로 구성된다. 6에 의해 1.2 TR이 제어 태스크로 구성되고, 8.1에 의해 1.3, 1.4, 1.5, 1.6, 1.7, 1.8 TR이 1.2 TR에 포함된다.



<그림 4> 로봇제어 시스템의 확장 방향성 그래프로의 변환

로봇 제어 시스템의 부 시스템 관계 테이블																	
	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2	3	4	5	6	7	8	9
EV in		1.1 2	1.2	1.2	1.2	1.2	1.2	1.2		1.5		1.7 1.8 6					
EV out	1.2	1.3(T) 1.4(T) 1.5(T) 1.6(T) 1.7(T) 1.8(T)			2		4	4		1.2				4(A)			
MSG in	DE V		1.1						1.4 1.5 1.6 1.7 1.8	1.4 6 7	2	3	4 DEV	5	2		DEV
MSG out	1.3			2 1.9	1.9	1.9	1.9	1.9	DEV	3 7	4	5	6 DEV	2	2	DE V	
타이머	Null	Null	Null	Null	Null	Null	Null	Null	Null	Null	Null	Null	Null	Null	Null	any	any
TR의 종류	3	1	2	2	2	2	2	2	4	2	2	2	3 4	2	2	4	3

[표 4] 로봇 제어 시스템의 부 시스템 관계 테이블
(TR의 종류 1) CTR 2) DTR 3) Source_DevTR 4) Sink_DevTR
A는 AEV를 T는 TEV를 의미한다.)

	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2	3	4	5	6	7	8	9
2단계	*								*				*			*	*
3.1 단계	*												*				
3.2 단계																*	*
3.3 단계									*								
3.4 단계																	
4 단계										*	*	*		*	*		
5.1 단계																	
5.2 단계												*		*			
5.3 단계										*	*				*		
6 단계		*															
7 단계			*	*	*	*	*	*									
8.1 단계			*	*	*	*	*	*									
8.2 단계																	

[표 5] 부 시스템의 각 단계별 적용표

SND \ RCV	2	3	4	6	7	sink
2	0	1	0	0	1	0
3	0	0	1	0	0	0
4	0	0	0	1	0	3(5)
6	1	0	2	0	0	0
7	1	0	0	0	0	0
Timer Event	0	0	0	0	0	0

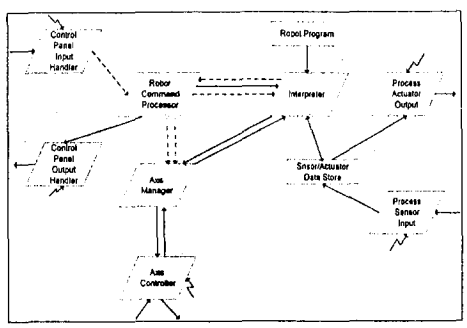
[표 6] 그림4의 인접 매트릭스 표현

부 시스템	태스크 이름
1.1	Control Panel Input Handler
1.2 ~ 1.8	Robot Command Processor
1.9	Control Panel Output Handler
4, 6	Axis Manager
5	Axis Controller
2, 3, 7	Interpreter
8	Process Actuator Output
9	Process Sensor Input

[표 7] 부 시스템과 태스크의 사상관계

[표5]는 부 시스템이 적용되는 단계를 도표로 보여주고 있다. 5를 제외한 단계들은 부 시스템 관계 테이블을 참조하여 작업을 진행하지만 5는 부 시스템 관계 테이블을 참조하지 않고 다른 지침을 사용한다. 5를 진행하기 위해서는 C&DFD를 확장 방향성 그래프로 변형해야한다. [그림 4]는 C&DFD를 확장 방향성 그래프로 변형시키는 지침을 보여주고 있다. 이것의 인접 매트릭스 표현이 [표 6]에서 보여진다. [표 6]에서 마킹 1의 개수가 2개이상인 SND DTR은 2 DTR이다. 그리고 이것의 RCV DTR인 3, 7은 병렬적 수행할 수 없기 때문에 2, 3, 7 DTR은 하나의 태스크로 구성한다(순차적 함수 응집도). [표 6]을 참조하여 응답 이벤트에 의해 동작되는 부 시스템의 집합을 태스크로 구성한다(순차 응집도 지침). [표 6]에서 마킹이 2인 부분의 전송DTR은 6 DTR이고, 수신DTR은 4 DTR이다. 그래서 4, 6 DTR은 하나의 태스크로 구성된다. 부 시스템과 태스크의 사상은 [표 7]과 같다.

이렇게 구해진 병렬 태스크를 다이어그램으로 표현하면 [그림 5]와 같은 TAD가 표현된다.



[그림 5] 로봇제어 시스템의 Task Architecture Diagram(TAD)

7. 결론 및 추후 연구

Gomaa의 방법론에서 소개된 CODARTS방법론은 태스크를 구성하는데 체계적인 지침을 제시하고 있지만, 내용이 광범위하고 추상적인 부분이 없지 않았다. 이러한 면들은 경험이 많은 설계자에 의해 극복될 수 있는 부분이지만 설계자가 다를 경우 또 다른 형태의 태스크 구성 표현이 존재하게되어 해석자마다 다른 관점에서 설계가 이해될 수 있고 또한 설계의 일관성이 결여될 수 있다. 그래서 본 연구에서는 이미 존재하는 CODARTS방법론을 적용단계별로 정리하여 단일한 설계를 도출할 수 있도록 하였고, 일관적인 해석을 가능하도록 하였다. 특히 응집도를 기반으로한 태스크 구성 지침은 다소 추상적이기 때문에 설계자가 이해하기 어려운 면이 있었다. 그러나 이러한 것들도 방향성 그래프이론을 적용하여 구체화시킬 수 있고, 이러한 시도로 인해 CODARTS의 추상적인 많은 부분이 구체화될 수 있다.

본 연구에서 제시한 병렬 태스크 구성 지침은 실시간 병렬 소프트웨어 시스템의 설계를 자동화하는데 이용될 수 있으며, 현재로서는 단일 컴퓨터 환경에서 수행되는 실시간 병렬 시스템을 다루고 있지만, 네트워크로 연결된 다중 컴퓨터 환경에서 수행되는 실시간 병렬 시스템의 설계로의 확장이 가능할 것이다.

또한 이러한 병렬 태스크를 구성하는 정량적인 측정도구를 제시하여 수치에 의한 태스크의 구성 및 재구성을 가능하게 함으로서 유연하고 정당한 병렬 태스크 구성이 가능하게 하는 연구가 수행되어야 할 것이다.

참 고 문 헌

[1] Gomaa. H, "Software Design Methods for Concurrent and Real-Time System", ADDISION-WESLEY, 1993

[2] Gomaa, H, "The Calibration and Validation of a Hybrid Simulation/Regression Model of a Batch Computer System" Software, Practice and Experience Vol.8, No.1, 1978.

[3] Page-Jones, M, "The Practical Guide to Structured Systems Design", Second Edition Prentice Hall, 1988.

[4] J. Bieman and L.M. Ott, "Measuring Functional Cohesion", IEEE Trans. Software Eng, vol.20, no.8, Aug. 1994.

[5] L. Briand and S. Morasca, "Property-Based Software Engineering Measurement", IEEE Trans. Software Eng, vol.22, no.1, JAN. 1996.

[6] 송영제, "C언어로 구현한 소프트웨어 엔지니어링", 홍릉과학출판사, 1991.