(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년04월06일
(11) 등록번호 10-2237167
(24) 등록일자 2021년04월01일

(51) 국제특허분류(Int. Cl.)
G06F 16/00 (2019.01)
(52) CPC특허분류
G06F 16/2365 (2019.01)
(21) 출원번호 10-2015-7014922
(22) 출원일자(국제) 2013년12월09일
심사청구일자 2018년12월10일
(85) 번역문제출일자 2015년06월04일
(65) 공개번호 10-2015-0095648
(43) 공개일자 2015년08월21일
(86) 국제출원번호 PCT/US2013/073899
(87) 국제공개번호 WO 2014/093232
국제공개일자 2014년06월19일
(30) 우선권주장
61/735,451 2012년12월10일 미국(US)
(뒷면에 계속)
(56) 선행기술조사문헌
KR1020100015478 A*
US07215637 B1*
JP07036706 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
아브 이니티오 테크놀로지 엘엘시
미국 02421 매사추세츠주 렉싱턴 스프링 스트리트 201
(72) 발명자
스튜더 스캇
미국 01833 매사추세츠주 조지타운 필즈베리 레인 27
홀리 조셉 스케핑턴 3세
미국 02478 매사추세츠주 벨몬트 힐크레스트 로드 11
와이즈만 아미트
미국 01730 매사추세츠주 베드포드 레드코트 로드 15
(74) 대리인
유미특허법인

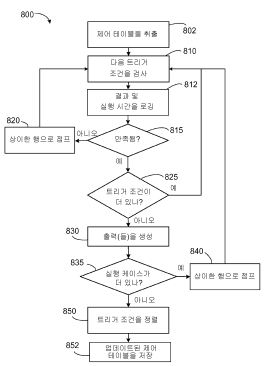
전체 청구항 수 : 총 60 항 심사관 : 이현중

(54) 발명의 명칭 **변환 생성용 시스템**

(57) 요 약

본 명세서는 변환을 규칙 세트에 기초하여 생성하는 것에 관련되는 기술을 설명한다. 일반적으로, 본 명세서에서 설명되는 하나의 양태는 실행 케이스를 포함하는 규칙 세트를 수신하는 단계를 포함하는 방법에서 구현될 수 있는데, 규칙 세트 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 하나 이상의 트리거 조건 모두가 만족되는 경우 생성될 출력의 사양을 포함한다. 이러한 방법은 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행들의 시퀀스를 포함하는 제어 구조를 생성하는 단계를 더 포함할 수도 있다. 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함할 수도 있다. 트리거 조건 중 적어도 하나에 대하여, 트리거 조건이 만족되지 않으면, 제어 구조는 행들의 시퀀스 내의 적어도 하나의 행을 스킵하도록 처리를 디렉팅할 수도 있다.

대 표 도 - 도8



(30) 우선권주장
61/751,814 2013년01월11일 미국(US)
13/958,037 2013년08월02일 미국(US)

명 세 서

청구범위

청구항 1

하나 이상의 데이터 처리 장치에 의하여 수행되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법으로서,

실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하는 단계로서, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하는, 규칙 세트를 수신하는 단계;

상기 규칙 세트에 적어도 부분적으로 기초하여 적어도 하나의 프로세서를 사용하는 제어 구조를 생성하는 단계로서,

상기 제어 구조는 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행들의 시퀀스를 정의하고, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 특정하고,

생성된 제어 구조는 상기 적어도 하나의 프로세서에 의해 실행시

하나의 행의 트리거 조건 중 하나가 만족되지 않을 경우 하나의 행으로부터 상이한 행으로 처리를 디렉팅하고,

상기 제어 구조 내의 제 1 행의 트리거 조건 중 적어도 하나에 대하여, 상기 제 1 행의 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조가 행들의 시퀀스 내에서 적어도 제 2 행을 스킵하는 처리를 디렉팅할 것으로 구성되는, 제어 구조를 생성하는 단계; 및

상기 제어 구조를 저장하거나 송신하는 단계

를 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 2

제 1 항에 있어서,

입력 데이터를 수신하는 단계;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 단계; 및

상기 제어 구조에 의하여 특정된 출력에 기초하여 데이터를 저장하거나 송신하는 단계를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 3

제 1 항에 있어서,

상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 생략된 트리거 조건은 실행 케이스에서 실행 케이스들의 시퀀스 내의 대응하는 실행 케이스 이전에 발생하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 4

제 1 항에 있어서,

행 내의 트리거 조건들의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 5

제 1 항에 있어서,

행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 6

제 1 항에 있어서,

상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 행에 대한 트리거 조건들의 시퀀스를 정렬하는 단계를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 7

제 1 항에 있어서,

행에 대한 트리거 조건들의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하는 단계를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 8

제 1 항에 있어서,

입력 데이터를 수신하는 단계;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 단계;

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하는 단계; 및

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하는 단계를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 9

제 1 항에 있어서,

행에 대한 트리거 조건들의 상기 시퀀스를 상기 트리거 조건에 대한 실패율(failure rate)에 기초하여 정렬하는 단계를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 10

제 1 항에 있어서,

입력 데이터를 수신하는 단계;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 단계;

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록(record)에 의하여 만족되는지 여부에 기초하여 업데이트하는 단계; 및

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하는 단계를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 11

제 1 항에 있어서,

상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코

딩하기 위한 방법.

청구항 12

제 1 항에 있어서,

상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 특정되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 13

제 1 항에 있어서,

상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 14

제 1 항에 있어서,

상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 15

제 1 항에 있어서,

상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법.

청구항 16

데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 소프트웨어를 저장하는 비휘발성 컴퓨터-판독가능 매체로서,

상기 소프트웨어는 적어도 하나의 프로세서를 포함하는 컴퓨팅 시스템에 의해 실행시 컴퓨팅 시스템으로 하여금:

실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하되, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하도록 하고;

상기 규칙 세트에 적어도 부분적으로 기초하여, 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 정의하는 제어 구조를 생성하도록 하고 - 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 특정하고,

생성된 제어 구조는 실행시 입력 데이터를 변환하기 위한 장래의 처리 도중에

하나의 행의 트리거 조건 중 하나가 만족되지 않을 경우 하나의 행으로부터 상이한 행으로 처리를 디렉팅하고,

상기 제어 구조의 제 1 행 내의 트리거 조건 중 적어도 하나에 대하여, 상기 제 1 행의 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조가 행들의 시퀀스 내에서 적어도 제 2 행을 스킵하는 처리를 디렉팅할 것으로 구성됨 -;

상기 제어 구조를 저장하거나 송신하도록 하는

명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 17

제 16 항에 있어서,

컴퓨터 시스템으로 하여금:

입력 데이터를 수신하도록 하고;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하도록 하고;

상기 제어 구조에 의하여 특정된 출력에 기초하여 데이터를 저장하거나 송신하도록 하는 명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 18

제 16 항에 있어서,

상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 상기 생략된 트리거 조건은 실행 케이스 내에서 실행 케이스들의 상기 시퀀스 내의 대응하는 실행 조건 이전에 발생하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 19

제 16 항에 있어서,

행 내의 트리거 조건들의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 20

제 16 항에 있어서,

행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 21

제 16 항에 있어서,

컴퓨팅 시스템으로 하여금:

상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 행에 대한 트리거 조건들의 시퀀스를 정렬하게 하는 명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 22

제 16 항에 있어서,

컴퓨팅 시스템으로 하여금:

행에 대한 트리거 조건들의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하게 하는 명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 23

제 22 항에 있어서,

컴퓨팅 시스템으로 하여금:

입력 데이터를 수신하도록 하고;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하도록 하고;

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하도록 하고;

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하도록 하는 명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 24

제 16 항에 있어서,

컴퓨팅 시스템으로 하여금:

행에 대한 트리거 조건들의 상기 시퀀스를 상기 트리거 조건에 대한 실패율에 기초하여 정렬하게 하는 명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 25

제 24 항에 있어서,

컴퓨팅 시스템으로 하여금:

입력 데이터를 수신하도록 하고;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하도록 하고;

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록에 의하여 만족되는지 여부에 기초하여 업데이트하도록 하고;

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하도록 하는 명령을 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 26

제 16 항에 있어서,

상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 27

제 16 항에 있어서,

상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 특정되는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 28

제 16 항에 있어서,

상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 29

제 16 항에 있어서,

상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 30

제 16 항에 있어서,

상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 소프트웨어를 저장하는 컴퓨터-판독가능 매체.

청구항 31

데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템으로서,

실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하도록 구성되는 입력 디바이스 또는 포트로서, 상기 실행

케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하는, 입력 디바이스 또는 포트; 및

상기 규칙 세트에 적어도 부분적으로 기초하여, 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행들의 시퀀스를 정의하는 제어 구조를 생성하는 동작들을 수행하도록 구성되는 적어도 하나의 프로세서로서,

각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 특정하고,

생성된 제어 구조는 실행시 입력 데이터를 변환하기 위한 장래의 처리 도중에

하나의 행의 트리거 조건 중 하나가 만족되지 않을 경우 하나의 행으로부터 상이한 행으로 처리를 디렉팅하고,

상기 제어 구조의 제 1 행 내의 트리거 조건 중 적어도 하나에 대하여, 상기 제 1 행의 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조가 행들의 시퀀스 내에서 적어도 제 2 행을 스킵하는 처리를 디렉팅할 것으로 구성되는, 적어도 하나의 프로세서; 및

상기 제어 구조를 송신하도록 구성되는 출력 디바이스 또는 포트

를 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 32

제 31 항에 있어서,

입력 데이터를 수신하도록 구성되는 입력 디바이스 또는 포트;

동작들을 수행하도록 구성되는 적어도 하나의 프로세서로서, 상기 동작들을 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작을 포함하는, 적어도 하나의 프로세서; 및

데이터를 상기 제어 구조에 의하여 특정되는 출력에 기초하여 송신하도록 구성되는 출력 디바이스 또는 포트를 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 33

제 31 항에 있어서,

상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 상기 생략된 트리거 조건은 실행 케이스 내에서 실행 케이스들의 상기 시퀀스 내의 대응하는 실행 조건 이전에 발생하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 34

제 31 항에 있어서,

행 내의 트리거 조건들의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 35

제 31 항에 있어서,

행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 36

제 31 항에 있어서,

상기 동작들은:

상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 행에

대한 트리거 조건들의 시퀀스를 정렬하는 동작을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 37

제 31 항에 있어서,

상기 동작들은:

행에 대한 트리거 조건들의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하는 동작을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 38

제 37 항에 있어서,

상기 동작들은:

입력 데이터를 수신하는 동작;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작;

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하는 동작 및

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하는 동작을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 39

제 31 항에 있어서,

상기 동작들은:

행에 대한 트리거 조건들의 상기 시퀀스를 상기 트리거 조건에 대한 실패율에 기초하여 정렬하는 동작을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 40

제 39 항에 있어서,

상기 동작들은:

입력 데이터를 수신하는 동작;

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작;

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록에 의하여 만족되는지 여부에 기초하여 업데이트하는 동작; 및

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하는 동작을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 41

제 31 항에 있어서,

상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 42

제 31 항에 있어서,

상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 특정되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 43

제 31 항에 있어서,

상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 44

제 31 항에 있어서,

상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 45

제 31 항에 있어서,

상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 46

데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템으로서,

실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하도록 구성되는 입력 디바이스 또는 포트로서, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하는, 입력 디바이스 또는 포트;

상기 규칙 세트에 적어도 부분적으로 기초하여, 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행들의 시퀀스를 정의하는 제어 구조를 생성하기 위한 수단으로서,

각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 특정하고,

생성된 제어 구조는 실행시 입력 데이터를 변환하기 위한 장래의 처리 도중에

하나의 행의 트리거 조건 중 하나가 만족되지 않을 경우 하나의 행으로부터 상이한 행으로 처리를 디렉팅하고,

상기 제어 구조의 제 1 행 내의 트리거 조건 중 적어도 하나에 대하여, 상기 제 1 행의 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조가 행들의 시퀀스 내에서 적어도 제 2 행을 스킵하는 처리를 디렉팅할 것으로 구성되는, 제어 구조를 생성하기 위한 수단; 및

상기 제어 구조를 저장하도록 구성되는 데이터 스토리지 시스템

을 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 47

제 46 항에 있어서,

입력 데이터를 수신하도록 구성되는 입력 디바이스 또는 포트;

동작들을 수행하도록 구성되는 적어도 하나의 프로세서로서, 상기 동작들을 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작을 포함하는, 적어도 하나의 프로세서; 및

데이터를 상기 제어 구조에 의하여 특정되는 출력에 기초하여 송신하도록 구성되는 출력 디바이스 또는 포트를 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 48

제 46 항에 있어서,

상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 상기 생략된 트리거 조건은 실행 케이스 내에서 실행 케이스들의 상기 시퀀스 내의 대응하는 실행 조건 이전에 발생하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 49

제 46 항에 있어서,

행 내의 트리거 조건의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 50

제 46 항에 있어서,

행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 51

제 46 항에 있어서,

상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 행에 대한 트리거 조건들의 시퀀스를 정렬하기 위한 수단을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 52

제 46 항에 있어서,

행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하기 위한 수단을 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 53

제 52 항에 있어서,

입력 데이터를 수신하도록 구성되는 입력 디바이스 또는 포트;

동작들을 수행하도록 구성되는 적어도 하나의 프로세서로서, 상기 동작들은:

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작, 및

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하는 동작을 수행하는, 적어도 하나의 프로세서; 및

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하기 위한 수단을 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 54

제 46 항에 있어서,

행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실패율에 기초하여 정렬하기 위한 수단을 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 55

제 54 항에 있어서,

입력 데이터를 수신하도록 구성되는 입력 디바이스 또는 포트;

동작들을 수행하도록 구성되는 적어도 하나의 프로세서로서, 상기 동작들은:

트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작, 및

고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록에 의하여 만족되는지 여부에 기초하여 업데이트하는 동작을 포함하는, 적어도 하나의 프로세서; 및

상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하기 위한 수단을 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 56

제 46 항에 있어서,

상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 57

제 46 항에 있어서,

상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 특정되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 58

제 46 항에 있어서,

상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 59

제 46 항에 있어서,

상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

청구항 60

제 46 항에 있어서,

상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 컴퓨팅 시스템.

발명의 설명

기 술 분 야

[0001] 관련 출원들에 대한 상호 참조

[0002] 본 출원은, 2013 년 1 월 11 일에 출원된 미국 가출원 번호 제 61/751,814 호 및 2012 년 12 월 10 일에 출원된 미국 가출원 번호 제 제 61/735,451 호에 대한 우선권을 주장하는, 2013 년 8 월 2 일 출원된 미국 특허 출원 번호 제 13/958,037 호에 대한 우선권을 주장하는데, 이들의 그 전체 내용은 원용에 의해 본 명세서에 포함된다.

배 경 기 술

[0003] 본 명세서는 규칙 세트에 기초하여 데이터에 대한 변환을 생성하기 위한 시스템에 관련된다.

[0004] 복잡한 계산들은 흔히 디렉팅된 그래프("데이터흐름 그래프"라고 불림)를 통한 데이터 흐름이라고 표현될 수 있는데, 계산의 컴포넌트는 그래프의 꼭지점들 및 그래프의 링크(아크, 에지)에 대응하는 컴포넌트들 사이의 데이터 흐름에 연관된다. 컴포넌트는 데이터를 하나 이상의 입력 포트에서 수신하고, 데이터를 처리하며, 하나 이상의 출력 포트로부터 데이터를 제공하는 데이터 처리 컴포넌트 및 데이터 흐름의 소스 또는 싱크로서의 역할을 하는 데이터셋 컴포넌트를 포함할 수 있다. 이러한 그래프-기초 계산을 구현하는 시스템은 미국 특허 제 5,966,072 호인 "EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS"에서 설명된다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0005] 일반적 양태 1 에서, 하나 이상의 데이터 처리 장치에 의하여 수행되는, 데이터를 변환하기 위한 규칙 세트를 인코딩하기 위한 방법은: 실행 케이스의 시퀀스를 포함하는 규칙 세트를 수신하는 단계로서, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하는, 단계; 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하는 단계로서, 각각의 행은: 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성되는, 단계; 및 상기 제어 구조를 저장하거나 송신하는 단계를 포함한다.

[0006] 입력 데이터를 수신하는 단계; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 단계; 및 상기 제어 구조에 의하여 특정된 출력에 기초하여 데이터를 저장하거나 송신하는 단계를 더 포함하는, 양태 1 에 따르는 양태 2.

[0007] 상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 생략된 트리거 조건은 실행 케이스에서 실행 케이스의 시퀀스 내의 대응하는 실행 케이스 이전에 발생하는, 양태 1 및 양태 2 중 임의의 하나에 따르는 양태 3.

[0008] 행 내의 트리거 조건의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 양태 1 내지 양태 3 중 임의의 하나에 따르는 양태 4.

[0009] 행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 양태 1 내지 양태 4 중 임의의 하나에 따르는 양태 5.

[0010] 상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 행에 대한 트리거 조건들의 시퀀스를 정렬하는 단계를 더 포함하는, 양태 1 내지 양태 5 중 임의의 하나에 따르는 양태 6.

[0011] 행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하는 단계를 더 포함하는, 양태 1 내지 양태 6 중 임의의 하나에 따르는 양태 7.

[0012] 입력 데이터를 수신하는 단계; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 단계; 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하는 단계; 및 상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하는 단계를 더 포함하는, 양태 1 내지 양태 7 중 임의의 하나에 따르는 양태 8.

[0013] 행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실패율에 기초하여 정렬하는 단계를 더 포함하는, 양태 1 내지 양태 8 중 임의의 하나에 따르는 양태 9.

[0014] 입력 데이터를 수신하는 단계; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 단계; 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록(record)에 의하여 만족되는지 여부에 기초하여 업데이트하는 단계; 및 상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하는 단계를 더 포함하는, 양태 1 내지 양태 9 중 임의의 하나에 따르는 양태 10.

[0015] 상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 양태 1 내지 양태 10 중 임의의 하나에 따르는 양태 11.

[0016] 상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 규정되는, 양태 1 내지 양태 11 중 임의의 하나에 따르는 양태 12.

[0017] 상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 양태 1 내지 양태 12 중 임의의 하나에 따르는 양태 13.

[0018] 상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 양태 1 내지 양태 13 중 임의의 하나에 따르는 양태 14.

[0019] 상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 양태 1 내지 양태 14 중 임의의 하나에 따르는 양태 15.

[0020] 일반적 양태 16 에서, 데이터 처리 장치 및 데이터 처리 장치에 커플링된 메모리를 포함하는 시스템. 데이터 처리 장치에 의하여 실행되면 데이터 처리 장치가, 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하되, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하도록 하는 저장된 명령들을 가지는 메모리. 동작들은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하는 동작을 더 포함할 수도 있는데, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 동작들은 상기 제어 구조를 저장하거나 송신하는 동작을 더 포함할 수도 있다.

[0021] 동작들이 입력 데이터를 수신하는 동작; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작; 및 상기 제어 구조에 의하여 특정된 출력에 기초하여 데이터를 저장하거나 송신하는 동작을 더 포함하는, 양태 16 에 따르는 양태 17.

[0022] 상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 생략된 트리거 조건은 실행 케이스에서 실행 케이스의 시퀀스 내의 대응하는 실행 케이스 이전에 발생하는, 양태 16 및 양태 17 중 임의의 하나에 따르는 양태 18.

[0023] 행 내의 트리거 조건의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 양태 16 내지 양태 18 중 임의의 하나에 따르는 양태 19.

[0024] 행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 양태 16 내지 양태 19 중 임의의 하나에 따르는 양태 20.

[0025] 동작들이 상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 행에 대한 트리거 조건들의 시퀀스를 정렬하는 동작을 더 포함하는, 양태 16 내지 양태 20 중 임의의 하나에 따르는 양태 21.

[0026] 동작들이: 행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하는 동작을 더 포함하는, 양태 16 내지 양태 21 중 임의의 하나에 따르는 양태 22.

[0027] 동작들이: 입력 데이터를 수신하는 동작; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작; 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하는 동작; 및 상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하는 동작을 더 포함하는, 양태 16 내지 양태 22 중 임의의 하나에 따르는 양태 23.

[0028] 동작들이: 행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실패율에 기초하여 정렬하는 동작을 더 포함하는, 양태 16 내지 양태 23 중 임의의 하나에 따르는 양태 24.

[0029] 동작들이: 입력 데이터를 수신하는 동작; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작; 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록(record)에 의하여 만족되는지 여부에 기초하여 업데이트하는 동작; 및 상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하는 동작을 더 포함하는, 양태 16 내지 양태 24 중 임의의 하나에 따르는 양태 25.

[0030] 상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 양태 16 내지 양태 25 중 임의의 하나에 따르는 양태 26.

[0031] 상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 규정되는, 양태 16 내지 양태 26 중 임의의 하나에 따르는 양태 27.

[0032] 상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 양태 16 내지 양태 27 중 임의의 하나에 따르는 양태 28.

[0033] 상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 양태 16 내지 양태 28 중 임의의 하나에 따르는 양태 29.

[0034] 상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 양태 16 내지 양태 29 중 임의의 하나에 따르는 양태 30.

[0035] 일반적 양태 31 에서, 처리 디바이스에 의하여 실행가능하고 이러한 실행 시에 처리 디바이스가, 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하게 하되, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하게 하는 명령들을 포함하는 소프트웨어를 저장하는 컴퓨터 판독가능 스토리지 매체. 동작들은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하는 동작을 더 포함할 수도 있는데, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 동작들은 상기 제어 구조를 저장하거나 송신하는 동작을 더 포함할 수도 있다.

[0036] 동작들이 입력 데이터를 수신하는 동작; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작; 및 상기 제어 구조에 의하여 특정된 출력에 기초하여 데이터를 저장하거나 송신하는 동작을 더 포함하는, 양태 31 에 따르는 양태 32.

[0037] 상기 행 중 적어도 하나는 대응하는 실행 케이스에 대한 트리거 조건을 생략하고, 생략된 트리거 조건은 실행 케이스에서 실행 케이스의 시퀀스 내의 대응하는 실행 케이스 이전에 발생하는, 양태 31 및 양태 32 중 임의의 하나에 따르는 양태 33.

[0038] 행 내의 트리거 조건의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스인, 양태 31 내지 양태 33 중 임의의 하나에 따르는 양태 34.

[0039] 행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력 표현으로 디렉팅하는 코드의 일부인, 양태 31 내지 양태 34 중 임의의 하나에 따르는 양태 35.

[0040] 동작들이 상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하

여 행에 대한 트리거 조건들의 시퀀스를 정렬하는 동작을 더 포함하는, 양태 31 내지 양태 35 중 임의의 하나에 따르는 양태 36.

[0041] 동작들이: 행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실행 시간에 기초하여 정렬하는 동작을 더 포함하는, 양태 31 내지 양태 36 중 임의의 하나에 따르는 양태 37.

[0042] 동작들이: 입력 데이터를 수신하는 동작; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작; 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간을 상기 입력 데이터로써 상기 트리거 조건을 실행하는 데에 걸린 시간에 기초하여 업데이트하는 동작; 및 상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실행 시간에 기초하여 정렬하는 동작을 더 포함하는, 양태 31 내지 양태 37 중 임의의 하나에 따르는 양태 38.

[0043] 동작들이: 행에 대한 트리거 조건의 상기 시퀀스를 상기 트리거 조건에 대한 실패율에 기초하여 정렬하는 동작을 더 포함하는, 양태 31 내지 양태 38 중 임의의 하나에 따르는 양태 39.

[0044] 동작들이: 입력 데이터를 수신하는 동작; 트리거 조건을 상기 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 동작; 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율을 상기 트리거 조건이 상기 입력 데이터 내의 기록(record)에 의하여 만족되는지 여부에 기초하여 업데이트하는 동작; 및 상기 제어 구조 내의 행에 대한 트리거 조건에 대한 포인터를 업데이트된 실패율에 기초하여 정렬하는 동작을 더 포함하는, 양태 31 내지 양태 39 중 임의의 하나에 따르는 양태 40.

[0045] 상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 더 포함하는, 양태 31 내지 양태 40 중 임의의 하나에 따르는 양태 41.

[0046] 상기 규칙 세트는 그래픽 사용자 인터페이스를 통하여 규정되는, 양태 31 내지 양태 41 중 임의의 하나에 따르는 양태 42.

[0047] 상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현되는, 양태 31 내지 양태 42 중 임의의 하나에 따르는 양태 43.

[0048] 상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력 표현에 의하여 표현되는, 양태 31 내지 양태 43 중 임의의 하나에 따르는 양태 44.

[0049] 상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력 표현이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)인, 양태 31 내지 양태 44 중 임의의 하나에 따르는 양태 45.

[0050] 상기 제어 구조는 복수 개의 처리 디바이스에 의하여 병렬적으로 실행되는 변환의 일부인, 양태 1 내지 양태 15 중 임의의 하나에 따르는 양태 46.

[0051] 상기 제어 구조는 복수 개의 처리 디바이스에 의하여 병렬적으로 실행되는 변환의 일부인, 양태 16 내지 양태 30 중 임의의 하나에 따르는 양태 47.

[0052] 상기 제어 구조는 복수 개의 처리 디바이스에 의하여 병렬적으로 실행되는 변환의 일부인, 양태 31 내지 양태 45 중 임의의 하나에 따르는 양태 48.

[0053] 일 양태에서, 일반적으로, 규칙 세트에 기초하여 변환을 생성하기 위한 방법은 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하는 단계를 포함하고, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함한다. 이러한 방법은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하는 동작을 더 포함할 수도 있는데, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 이러한 방법은 상기 제어 구조를 저장하거나 송신하는 단계를 더 포함할 수도 있다.

[0054] 일반적으로, 본 명세서에서 설명되는 기술 요지의 하나의 양태는 데이터 처리 장치 및 데이터 처리 장치에 커플

링된 메모리를 포함하는 시스템에서 구현될 수 있다. 데이터 처리 장치에 의하여 실행되면 데이터 처리 장치가, 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하되, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하도록 하는 저장된 명령들을 가지는 메모리. 동작들은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하는 동작을 더 포함할 수도 있는데, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 동작들은 상기 제어 구조를 저장하거나 송신하는 동작을 더 포함할 수도 있다.

[0055] 일반적으로, 본 명세서에서 설명된 기술 요지의 하나의 양태는 처리 디바이스에 의하여 실행가능하고 이러한 실행 시에 처리 디바이스가, 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하게 하되, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함하게 하는 명령들을 포함하는 소프트웨어를 저장하는 컴퓨터 판독가능 스토리지 매체에서 구현될 수 있다. 동작들은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하는 동작을 더 포함할 수도 있는데, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 동작들은 상기 제어 구조를 저장하거나 송신하는 동작을 더 포함할 수도 있다.

[0056] 일반적으로, 본 명세서에서 설명된 기술 요지의 하나의 양태는 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하도록 구성되는 입력 디바이스 또는 포트를 포함하는 시스템에서 구현될 수 있는데, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함한다. 이러한 시스템은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행의 시퀀스를 포함하는 제어 구조를 생성하기 위한 수단을 더 포함할 수도 있는데, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 이러한 시스템은 제어 구조를 저장하도록 구성되는 데이터 스토리지 시스템을 포함할 수도 있다.

[0057] 일반적으로, 본 명세서에서 설명된 기술 요지의 하나의 양태는 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하도록 구성되는 입력 디바이스 또는 포트를 포함하는 시스템에서 구현될 수 있는데, 상기 실행 케이스의 시퀀스 내의 적어도 하나의 실행 케이스는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함한다. 이러한 시스템은 동작들을 수행하도록 구성되는 적어도 하나의 프로세서를 포함할 수도 있는데, 동작들은 상기 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행들의 시퀀스를 포함하는 제어 구조를 생성하는 동작을 포함하고, 각각의 행은 하나 이상의 트리거 조건들의 시퀀스 및 대응하는 실행 케이스에 대한 출력을 특정하는 정보를 포함하고, 상기 생성된 제어 구조는 상기 트리거 조건 중 하나가 만족되지 않을 경우 상이한 행에서 계속하게 처리를 디렉팅하도록 구성되며, 상기 생성된 제어 구조는, 입력 데이터를 변환하기 위한 장래의 처리 도중에, 상기 트리거 조건 중 적어도 하나가 만족되지 않을 경우 상기 제어 구조 내의 상기 트리거 조건 중 적어도 하나에 대하여, 상기 제어 구조가 상기 행의 시퀀스 내의 적어도 하나의 행을 스킵하게 처리를 디렉팅하도록 구성된다. 이러한 시스템은 제어 구조를 송신하도록 구성되는 출력 디바이스 또는 포트를 포함할 수도 있다.

[0058] 양태들은 후속하는 피쳐들 중 하나 이상을 포함할 수 있다. 입력 데이터가 수신될 수도 있고 트리거 조건이 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사될 수도 있다. 제어 구조에 의하여 특정되는 출력에 기초하는 데이터는 저장되거나 송신될 수도 있다. 상기 행 중 적어도 하나는 대응하는 실행 케이스

에 대한 트리거 조건을 생략할 수도 있는데, 상기 생략된 트리거 조건은 실행 케이스 내에서 실행 케이스들의 상기 시퀀스 내의 대응하는 실행 조건 이전에 발생한다. 행 내의 트리거 조건의 상기 시퀀스는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내의 트리거 조건으로 각각 디렉팅하는 코드의 일부의 시퀀스일 수도 있다. 행 내의 출력을 특정하는 정보는 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내의 출력으로 디렉팅하는 코드의 일부일 수도 있다. 행에 대한 트리거 조건들의 시퀀스는 상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행에 기초하여 정렬될 수도 있다. 행에 대한 트리거 조건들의 시퀀스는 트리거 조건에 대한 실행 시간에 기초하여 정렬될 수도 있다. 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간은 입력 데이터로써 트리거 조건을 실행시키는데 걸리는 시간에 기초하여 갱신될 수도 있다. 제어 구조 내의 행에 대한 트리거 조건으로의 포인터는 업데이트된 실행 시간에 기초하여 정렬될 수도 있다. 행에 대한 트리거 조건들의 시퀀스는 트리거 조건에 대한 실패율에 기초하여 정렬될 수도 있다. 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율은 트리거 조건이 입력 데이터 내의 기록에 의하여 만족되는지 여부에 기초하여 업데이트될 수도 있다. 제어 구조 내의 행에 대한 트리거 조건으로의 포인터는 업데이트된 실패율에 기초하여 정렬될 수도 있다. 상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 포함할 수도 있다. 규칙 세트는 그래픽 사용자 인터페이스를 통하여 특정될 수도 있다. 상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조 내의 단일 트리거 조건에 의하여 표현될 수도 있다. 상기 규칙 세트 내의 상이한 실행 케이스에 대한 적어도 두 개의 출력들은 결합되고 상기 제어 구조의 행 내의 단일 출력에 의하여 표현될 수도 있다. 상기 제어 구조는 상기 트리거 조건에 대응하는 노드 및 상기 제어 구조의 상기 행 내의 출력이 있는 비순환성 디렉팅된 그래프(acyclic directed graph)일 수도 있다.

발명의 효과

[0059] 양태들은 후속하는 장점들 중 하나 이상을 포함할 수 있다.

[0060] 몇몇 구현형태는 규칙 세트에 기초한 변환을 위한 기록당 처리 시간을 단축할 수도 있다. 몇몇 구현형태는 규칙 세트에 기초한 변환을 위한 컴파일레이션(compilation) 및 기동 시간을 단축할 수도 있다. 몇몇 구현형태는 규칙 세트에 기초한 변환을 위한 컴파일레이션 동안의 메모리 사용을 감소시킬 수도 있다. 몇몇 구현형태는 물리적 엔티티, 예컨대 무엇보다 비행기, 차량, 컴퓨터, 빌딩, 또는 다른 기반구조를 나타내는 데이터의 더 효율적인 처리를 제공할 수도 있다. 몇몇 구현형태는 방대한 양의 데이터를 처리하는 사용자의 인지적 부담(cognitive burden)을 절감시킬 수도 있는데, 예를 들어 방대한 양의 데이터(예를 들어, 수백만 또는 수십억 개의 기록)의 처리가 사용자에 의하여 더 용이하게 특정될 수도 있어서 사용자가 데이터의 처리를 더 용이하게 이해하고 따라서 규칙 세트 사양에 대한 효율적 구조를 발견하는 것에 대하여 우려하지 않고 특정 애플리케이션의 그 사용자의 도메인 지식을 적용하도록 한다.

[0061] 본 발명의 다른 특징들, 및 이점들은 후속하는 상세한 설명 및 청구항들로부터 명확하게 될 것이다.

도면의 간단한 설명

[0062] 도 1 은 예시적인 데이터흐름 그래프의 개략도이다.

도 2a 및 도 2b 는 스프레드쉬트-기초 규칙 엔트리에 대한 예시적인 그래픽 사용자 인터페이스의 예이다.

도 3a 는 규칙 세트에 대한 고유한 트리거 조건의 목록의 일 예를 예시한다.

도 3b 는 규칙 세트에 대한 고유한 출력의 목록의 일 예를 예시한다.

도 4a 및 도 4b 는 비순환성 디렉팅된 그래프로서 표현되는 규칙 세트에 기초한 변환에 대한 예시적인 제어 구조를 도시한다.

도 5 는 디스플레이된 테이블로서 표현되는 규칙 세트에 기초한 변환에 대한 예시적인 제어 구조를 도시한다.

도 6 은 그래프-기초 계산을 실행하기 위한 시스템의 블록도이다.

도 7 은 규칙 세트에 기초하는 변환을 생성하고 실행하기 위한 예시적인 프로세스의 흐름도이다.

도 8 은 규칙 세트에 기초한 변환을 실행하기 위한 예시적인 프로세스의 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0063] 그래프-기초 계산이 큰 집합의 데이터를 처리하기 위하여 사용될 수도 있다. 예를 들어, 신용 카드 발급자들은 그래프-기초 계산을 사용하여 수백만 장의 신용 카드에 대한 트랜잭션 데이터를 처리하여 보상 포인트를 발급하고 보상 포인트의 상환을 위한 제안에서 제시할 제휴사들의 제품을 선택할 수도 있다. 다른 예에서, 항공사는 그래프-기초 계산을 사용하여 수백만 명의 항공사 승객에 대한 빈번한 비행 마일리지 계좌를 업데이트할 수도 있다. 다른 예에서, 은행은 그래프-기초 계산을 사용하여 다양한 소스로부터의 소비자 데이터를 처리하고 특정 소비자에 대한 이용가능한 데이터에 의존하는 최대량까지의 대출에 대한 대출 승인을 생성할 수도 있다. 다른 예에서, 항공사는 그래프-기초 계산을 사용하여 비행기 군의 유지 보수를 추적하고 제어할 수도 있다. 다른 예에서, 렌트카 회사는 그래프-기초 계산을 사용하여 차량 군의 유지보수를 추적하고 제어할 수도 있다. 다른 예에서, 온라인 서비스 제공자는 그래프-기초 계산을 사용하여 웹 서버의 하나 이상의 클러스터에 대한 유지 보수 및/또는 부하 밸런싱을 추적하고 제어할 수도 있다. 다른 예에서, 도시는 그래프-기초 계산을 사용하여 도로 트래픽 데이터에 기초하여 도로 신호등을 제어할 수도 있다. 다른 예에서, 무선 네트워크 운영자는 그래프-기초 계산을 사용하여 개인용 무선 통신 디바이스(예를 들어, 스마트 폰 또는 태블릿 디바이스)에 대한 무선 네트워크 액세스 및 대역폭 할당을 제어할 수도 있다. 이것들은 그래프-기초 계산의 애플리케이션의 몇 가지 예시들일 뿐이고 많은 다른 애플리케이션들이 가능하다.

[0064] 그래프-기초 계산은 입력 데이터 내의 세트 내의 기록(또는 다른 데이터 엘리먼트)에 적용되는 변환에 대응하는 하나 이상의 컴포넌트를 포. 일반적으로, 변환은 입력 기록을 처리하여 하나 이상의 출력 기록을 생성(예를 들어, 생성 또는 업데이트)한다. 예를 들어, 신용 카드 발급자는 신용 카드 계정 기록과 연관된 트랜잭션 기록을 처리하여 제안된 신용 카드 트랜잭션에 대한 트랜잭션 승인 또는 거절 기록을 생성할 수도 있다. 어떻게 변환이 입력 기록을 처리하여 출력 기록을 생성하는지에 대한 세부사항은 복잡할 수 있고 특정 애플리케이션에 관련된 많은 양의 도메인 지식에 의존할 수도 있다. 소프트웨어 개발 또는 코딩 지식이 별로 없는 사용자(예를 들어, 운영자 또는 개발자)가 특정 애플리케이션의 그들의 지식에 기초하여 변환을 용이하게 구성할 수 있도록 허용하는 것이 유용할 수도 있다. 이해하기 쉬운 인터페이스를 통해서 특정된 규칙들에 기초하여 효율적인 변환을 생성하는 시스템이 이러한 사용자의 인지 부하(cognitive load)를 감소시키는 것을 도울 수도 있다.

[0065] 몇 가지 구현형태들에서, 사용자는 잠재적으로 복잡한 변환을 스프레드쉬트-기초 그래픽 사용자 인터페이스(GUI)를 통해서 구성하도록 허용될 수도 있다. 예를 들어, 사용자는 각각의 실행 케이스에 대한 스프레드 시트-기초 그래픽 사용자 인터페이스 내에 행들을 생성함으로써 실행 케이스들을 포함하는 규칙 세트를 특정할 수도 있다. 각각의 실행 케이스는 입력 데이터 및 하나 이상의 출력에 대하여 테스트될 수도 있는, 트리거 조건이라고 불리는 하나 이상의 조건을 포함할 수도 있다. 어떤 실행 케이스에 대한 트리거 조건의 모두가 입력 데이터에 의하여 만족된다면, 그 실행 케이스에 대한 하나 이상의 출력이 생성될 수도 있다. 예를 들어, 트리거 조건 및 출력은, 소프트웨어 코딩 기술은 부족하지만 도메인 지식을 가지고 있으며 스프레드쉬트를 편안하게 다루는 사용자가 규칙 세트를 특정하도록 하는 스프레드쉬트-기초 GUI의 열들에 대응할 수도 있다. 몇 가지 구현형태들에서, 스프레드쉬트-기초 GUI가 아닌 포맷들이 규칙 세트를 특정하기 위하여 사용될 수도 있다(예를 들어, 데이터 조작 언어(DML)로).

[0066] 사용자가 특정된 규칙 세트를 가지면, 입력 데이터 기록을 처리할 수 있는 하나 이상의 변환이 규칙 세트에 기초하여 생성될 수도 있다. 규칙 세트는 트리거 조건 및/또는 다중 실행 케이스 내에서 발생하는 출력을 포함할 수도 있다. 변환은, 변환을 컴파일하고 이것을 입력 데이터 기록에 적용하기 위하여 요구되는 메모리 및 처리 시간을 감소시키기 위하여 규칙 세트 내의를 활용하는 방식으로 생성될 수도 있다. 예를 들어, 트리거 조건이 다중 실행 케이스에서 발생할 경우, 입력 데이터가 그 트리거 조건을 만족하지 못한다면 변환은 그러한 트리거 조건을 가진 잔여 실행 케이스들을 검사를 스킵할 수도 있다. 규칙 세트 내의 리던던시를 활용하는 것은, 변환이 더 적은 메모리 및 프로세서 사이클을 사용하면서 큰 세트의 입력 데이터에 대해 실행되도록 허용할 수도 있다. 변환은 부분적으로 변환의 실행 흐름을 제어하는 제어 구조로서 인코딩될 수도 있다. 제어 구조는 행이라고 불리는 트리거 조건의 논리적 그루핑 및 그 규칙 세트로부터의 실행 케이스에 각각 대응하는 출력을 포함할 수도 있다. 변환의 실행 도중에, 트리거 조건의 평가가 통과되면, 그 행 내의 트리거 조건들의 시퀀스 내의 다음 트리거 조건이 평가된다. 만일 어느 행 내의 마지막 트리거 조건이 입력 데이터로써 평가할 때 통과된다면, 적어도 부분적으로 행에 의하여 특정되는 대응하는 출력(들)이 생성된다. 만일 입력 데이터로써 평가될 경우 트리거 조건이 실패하면(예를 들어, 트리거 조건이 만족되지 않으면), 제어 구조는 처리를 상이한 실행 케이스에 대응하는 상이한 행으로 디렉팅할 수도 있다. 예를 들어, 트리거 조건의 고장 시에, 실행 흐름은 점프에 의하여 다수 개의 실행 케이스를 스킵하여 그 제어 구조 내의 행들의 시퀀스에서 다음이 아닌 행 내의 트리거 조

건의 평가를 시작할 수도 있다. 몇 가지 구현형태들에서, 제어 구조는 고유한 트리거 조건 및 고유한 출력의 목록을 참조하여, 이러한 트리거 조건 및 출력이, 심지어 이들이 규칙 세트에 걸쳐 여러 번 발생하는 경우에도 변환(들)의 인코딩 중에 한번만 저장될 필요가 있도록 한다.

[0067] 몇 가지 구현형태들에서 트리거 조건 및 규칙 세트의 사양 내에서 발생하는 트리거 조건으로의 참조는 대응하는 변환(들)에 대하여 제어 구조로부터 생략될 수도 있다. 예를 들어, 트리거 조건은, 동일한 트리거 조건이 트리거 조건의 현재의 실례가 평가되기 이전에 필수적으로 평가될 규칙 세트 내의 다른 곳에서 발생할 경우에 그 제어 구조의 행으로부터 생략될 수도 있다. 트리거 조건을 행으로부터 생략하면 메모리 소비 및 대응하는 변환에 대한 처리 시간을 감소시킬 수도 있다.

[0068] 몇 가지 구현형태들에서, 제어 구조의 행 내의 또는 이에 의하여 참조되는 트리거 조건은 트리거 조건의 파라미터에 기초하여 정렬될 수도 있다. 이러한 정렬은 변환(들)이 인가될 기록들에 대한 평균 처리 시간을 감소시키기 위하여 수행될 수도 있다. 예를 들어, 트리거 조건들의 시퀀스는 트리거 조건의 실행 시간, 트리거 조건이 평가 시에 실패하는 경우에 스킵될 행들의 개수(예를 들어, 점프 사이즈), 또는 그 변환(들)으로써 이전에 처리된 바 있는 입력 데이터에 대한 실패의 빈도에 기초하여 정렬될 수도 있다. 몇 가지 구현형태들에서 제어 구조 내의 행들의 시퀀스는 트리거 조건에 대한 점프 사이즈를 증가시키도록 정렬될 수도 있다(예를 들어, 공통 트리거 조건을 가지는 실행 케이스들을 함께 그루핑함으로써). 몇 가지 구현형태들에서, 행 내의 트리거 조건들의 시퀀스 및/또는 행들의 시퀀스를 변경하면 이전에 프로세스된 데이터와 유사한 통계를 가지는 큰 세트의 데이터 상에 대응하는 변환을 실행하기 위하여 요구되는 처리 시간을 감소시킬 수도 있다.

[0069] 본 명세서에서, 용어 "규칙 세트"는 하나 이상의 규칙의 콜렉션을 지칭하고, 여기에서 각각의 규칙은 하나 이상의 실행 케이스로 이루어진다. 하나의 규칙 내의 실행 케이스들은 입력 데이터의 처리 도중에 그 실행 케이스가 평가되는 순서(order)를 결정할 수도 있는 순서(ordering)를 가질 수도 있다. "실행 케이스"는 하나 이상의 출력과 연관된 하나 이상의 트리거 조건의 콜렉션이다. "트리거 조건"은 임의의 다른 실행 케이스에 대한 트리거 조건과 함께 실행 케이스가 작동할지(fire) 여부를 결정하기 위하여 사용되는 조건이다. 어떤 실행 케이스에 대한 트리거 조건 모두가 입력 데이터에 의하여 만족된다고 결정되면, 실행 케이스는 "작동하고(fire)", 이것은 그 실행 케이스에 대한 하나 이상의 출력들이 생성된다는 것을 의미한다. 출력은 이것이 규칙 세트의 일부로서 특정된 일정한 파라미터에 기초하여 생성된다는 의미에서 정적일 수도 있고, 또는 출력은 이것이 부분적으로 입력 데이터 값 및/또는 중간 결과 값에 기초하여 생성된다는 의미에서 동적일 수도 있다. 규칙들은 단일-작동식(single-fire) 또는 다중-작동식일 수도 있다. 단일-작동 규칙에서는, 자신의 만족되는 트리거 조건들 모두를 가지는 제 1 실행 케이스만이 작동한다. 다중-작동 규칙에서는, 그 규칙을 이루는 실행 케이스들 모두가 검사되고 출력은 자신의 개별적인 트리거 조건들이 만족되는 대상인 모든 실행 케이스에 대하여 생성된다. 한 규칙 세트 내의 다중 규칙들이 시퀀스 내의 데이터 흐름들로 적용될 수도 있다. 예를 들어, 규칙 세트로부터의 제 1 규칙에 기초한 변환은 다중 입력 데이터 소스로부터의 기록을 취하고 출력 기록을 생성할 수도 있다. 이러한 출력 기록이 이제 입력 기록으로서 제 2 변환으로 그 규칙 세트로부터의 제 2 규칙에 기초하여 전달되어 제 2 세트의 출력 기록을 생성할 수도 있다.

[0070] 도 1 은 하나 이상의 변환을 포함하는 예시적인 데이터흐름 그래프(100)의 개략도를 도시한다. 데이터는 하나 이상의 데이터 소스로부터 하나 이상의 데이터 싱크로의 데이터의 흐름을 처리하는, 데이터흐름 그래프(100)의 데이터 처리 컴포넌트의 시퀀스를 통하여 전달된다. 데이터흐름 그래프 내의 다양한 데이터 처리 컴포넌트 중 임의의 것은 별개의 처리 디바이스에서 실행중인 프로세스에 의하여 구현될 수 있고, 또는 다중 데이터 처리 컴포넌트들이 단일 처리 디바이스에서 실행 중인 하나 이상의 프로세스에 의하여 구현될 수 있다. 몇 가지 구현형태들에서, 입력 데이터 기록은 그들이 도착할 때에(예를 들어, 신용 카드 트랜잭션에 대한 요청에 응답하여) 연속적으로 처리될 수도 있다. 몇 가지 구현형태들에서, 데이터는 데이터흐름 그래프(100)에 의하여 처리될 한 세트의 입력 데이터 기록들을 식별하는 묶음 내에서 처리될 수도 있다.

[0071] 데이터흐름 그래프(100)에 의한 데이터의 묶음의 처리는 사용자 입력 또는 몇몇 다른 이벤트, 예컨대 타이머의 만료에 의하여 개시될 수도 있다. 한 묶음의 데이터의 처리가 시작되면, 입력 데이터 기록들이 하나 이상의 입력 데이터 소스로부터 독출된다. 예를 들어, 입력 데이터는 컴퓨터-판독가능 저장 디바이스, 예컨대 데이터 스토리지 컴포넌트(110)에 의하여 표현되는 것에 저장되는 하나 이상의 파일로부터 독출될 수도 있다. 입력 데이터 기록들은 서버에서 실행 중인 데이터베이스, 예컨대 데이터 스토리지 컴포넌트(112)로부터도 역시 독출될 수도 있다. 결합 컴포넌트(120)는 데이터(예를 들어, 기록)를 시퀀스 내의 다중 데이터 소스로부터 독출하고 입력 데이터를 이산 작업 유닛들의 시퀀스 내에 배치한다. 작업 유닛은, 예를 들어 입력 기록에 기초하여 선결정된 포맷으로 저장되는 기록을 나타낼 수도 있고, 또는 예를 들어 처리될 트랜잭션을 나타낼 수도 있다. 작업

유닛(예를 들어, 기록)은 차례대로 데이터흐름 그래프 내의 다음 컴포넌트로 전달된다.

[0072] 예시적인 데이터흐름 그래프(100)도 역시 변환 컴포넌트(130 및 140)를 포함한다. 변환 컴포넌트(130)에 의하여 실행되는 변환은 단일-작동 규칙에 기초한다. 즉, 오직 하나의 실행 케이스는 각각의 작업 유닛(예를 들어, 결합 프로세스로부터의 입력 데이터의 기록)에 대하여 작동할 것이다. 변환 컴포넌트(130)는 다음 데이터 처리 컴포넌트, 이러한 경우에는 변환 컴포넌트(140)로 전달되는 출력 기록을 생성한다.

[0073] 변환 컴포넌트(140)에 의하여 실행되는 변환은 다중-작동 규칙에 기초한다. 변환 컴포넌트(140)에 의하여 생성되는 출력 데이터 기록은 한 입력 작업 유닛에 대하여 작동된 실행 케이스들의 각각에 대응하는 출력 값의 목록을 포함할 수도 있다.

[0074] 예를 들어, 변환 컴포넌트(130)는 다양한 데이터 소스로부터의 신용 카드 트랜잭션 기록에 대응하는 결합 프로세스(120)로부터의 입력 데이터 기록을 처리할 수도 있다. 변환 컴포넌트(130)는 신용 카드에 할당된 보상 포인트의 양을 반영하는 출력 기록을 그 기록 내에 반영된 트랜잭션의 결과로서 생성할 수도 있다. 이러한 예에서, 변환 컴포넌트(140)는 이제 신용 카드 계정에 할당된 보상 포인트를 처리하여 대응하는 신용 카드 계정 보유자에 제시될 하나 이상의 제품 제공(offering)을 생성할 수도 있다.

[0075] 변환 컴포넌트(130) 및 변환 컴포넌트(140)는 변환 컴포넌트(130)에 대응하는 단일-작동 규칙 및 변환 컴포넌트(140)에 대응하는 다중-작동 규칙 모두를 포함하는 규칙 세트에 대응하는 더 큰 컴포넌트(150)로서 함께 그룹화될 수도 있다.

[0076] 작업 유닛이 데이터흐름 그래프의 데이터 처리 컴포넌트를 통해서 진행함에 따라, 각각의 작업 유닛과 연관된 결과 출력 기록은 데이터 싱크(170)로 전송되기 이전에 그들이 축적되는 데이터 큐(160)로 전달된다. 데이터 싱크(170)는 예를 들어 작업 유닛 또는 작업 유닛에 기초한 몇몇 축적된 출력을 저장하는 데이터 스토리지 컴포넌트일 수 있고, 또는 데이터 싱크(170)는 작업 유닛이 공개되는 큐(queue) 또는 최종 결과를 수신하기 위한 몇몇 다른 타입의 싱크일 수 있다. 몇 가지 구현형태들에서, 묶음 처리는 그 묶음 내의 모든 작업 유닛에 대한 결과들이 데이터 싱크(170)로 전송된 바 있을 경우에 종료된다. 이러한 포인트에서, 데이터흐름 그래프 내의 컴포넌트들은 종결될 수도 있다.

[0077] 도 2a 는 스프레드쉬트-기초 규칙 엔트리에 대한 예시적인 GUI(200)의 예이다. GUI(200)는 신용 카드 계정과 연관된 하나 이상의 기록에서 이용가능한 트랜잭션 데이터에 기초하여 신용 카드 계정에 대한 "보상 포인트" 값을 결정하는 단일-작동 규칙을 특정하기 위하여 사용자에 의하여 구성된 바 있다. GUI(200)는 그 규칙에 대한 5 개의 규칙 케이스를 특정하는 5 개의 행을 포함한다. GUI(200)는 그 규칙의 실행 케이스에 대한 트리거 조건 및 출력을 특정하고, 그리고 변환의 디버깅 또는 튜닝을 용이화하기 위하여 사용자에 의하여 사용될 수도 있는 규칙에 기초하여 변환의 테스트 실행으로부터의 메타데이터를 디스플레이하기 위하여 사용자에 의하여 사용된다. 예를 들어, GUI(200)는 도 6 의 애플리케이션 전문가 환경(622)을 통하여 사용자에게 제공될 수도 있다. 몇 가지 구현형태들에서, GUI(200)를 통하여 특정된 규칙 세트는 도 6 의 실행 환경(604)의 네트워크 인터페이스를 통하여 수신될 수도 있다.

[0078] 제 1 열(204)은 신용 카드 계정에 대한 평균 월별 요금을 반영하는 입력 데이터 기록 내의 변수에 적용되는 트리거 조건을 특정한다. 제 1 열 제 3 행의 다운-화살표(206)는 제 3 실행 케이스에 대한 제 1 트리거 조건이 제 2 실행 케이스에 대한 제 1 트리거 조건에 대한 트리거 조건과 동일하다는 것을 표시한다. 몇 가지 구현형태들에서, 사용자는 다운-화살표 아이콘을 수동으로 선택하여 다운-화살표를 스프레드쉬트-기초 GUI(200)의 하나 이상의 셀 내에 삽입할 수도 있고, 따라서 다운-화살표가 통과하는 실행 케이스에 대한 트리거 조건이 다운-화살표의 시작 바로 위의 트리거 조건과 동일하다는 것을 특정한다. 몇 가지 구현형태들에서, 사용자는 수동으로 스프레드쉬트-기초 GUI(200)의 인접한 셀에 동일한 트리거 조건을 수동으로 입력할 수도 있고 GUI(200)는 트리거 조건들이 동일하다는 것을 자동적으로 인식하고 이러한 반복을 표시하는 다운-화살표를 생성할 수도 있다.

[0079] GUI(200)의 제 2 열(208)은 신용 카드 계정이 활성화된 바 있는 년수를 반영하는 신용 카드 계정에 대한 입력 데이터 기록으로부터 유도되는 변수에 적용되는 트리거 조건을 특정한다. 열(208)은 한 쌍의 매칭 트리거 조건을 각각 표시하는 두 개의 다운-화살표(210)를 더 포함한다.

[0080] GUI(200)의 제 3 열(214)은 실행 케이스에 대한 트리거 조건의 모두가 평가되고 만족된다고 결정되는 경우 생성되는 출력들을 특정한다. 열(214)은 첫 번째 두 개의 실행 케이스들이 동일한 출력을 가지는 것을 표시하는 다운-화살표(216)를 더 포함한다. 사용자가 GUI(200)를 통하여 이것이 단일-작동 규칙(218)이라는 것을 특정한 바 있기 때문에, 실행 케이스는 하나가 작동하는 것이 발견될 때까지 한 번에 하나씩 평가될 수도 있다(예를 들

어, 실행 케이스 트리거 조건들 모두는 입력 데이터의 작업 유닛에 의하여 만족됨). 실행 케이스가 작동되면, 그러한 실행 케이스에 대하여 특정된 출력은 생성될 수 있을 것이고 이러한 규칙에 기초한 변환이 그 작업 유닛의 처리를 완료할 것이다. 여기에서 변환은 데이터흐름 내의 다음 작업 유닛을 처리하는 것을 개시하거나 또는 종결할 수도 있다.

[0081] GUI(200)의 마지막 행은 자신의 트리거 조건 셀들 모두가 키워드 "임의의 것"(230)으로 설정되게 하는데, 이것은 대응하는 트리거 조건이 존재하지 않는다는 것 또는 등가적으로 이러한 트리거 조건들이 언제나 참으로 평가된다는 것을 표시한다. 이러한 제 5 행이 트리거 조건을 가지지 않고 마지막에 평가되기 때문에, 이러한 제 5 실행 케이스에 대한 대응하는 출력이 디폴트 출력으로서 특정된다.

[0082] 제 4 열(220)은 변환의 추적된 테스트 실행으로부터의 메타데이터를 규칙의 디버깅 또는 튜닝을 용이화하기 위하여 사용자에 의하여 사용될 수도 있는 규칙에 기초하여 디스플레이한다. 일반적으로, 상이한 변환이 적어도 3 개의 상이한 동작 모드인: 생성 모드, 기록 테스트 모드, 및 파일 테스트 모드에 대한 규칙 세트에 기초하여 생성될 수도 있다. 생성 모드 변환은 규칙 세트의 로직을 구현하고 이것을 존재한다고 해도 적은 추가적 코드가 있는 입력 데이터로 적용한다. 기록 테스트 모드 변환은 개개의 작업 유닛(예를 들어, 계정, 트랜잭션, 항공사, 차량, 컴퓨터, 모바일 디바이스, 빌딩, 등을 나타내는 입력 데이터 기록)에 대한 변환의 실행을 통한 스테핑(stepping)을 가능하게 하도록 인코딩된다. 규칙 세트의 필수적 로직을 인코딩하는 것에 추가하여, 기록 테스트 모드 변환은 상세한 로깅 메시지(예를 들어, 각각의 입력 필드, 출력, 트리거 조건 결과(참 또는 거짓), 룩업 키, 룩업 필드, 및 어떤 실행 케이스가 작동되었는지, 및 몇몇 중간 파라미터의 값을 반영함)를 생성하는 코드를 포함할 수도 있다. 파일 테스트 모드 변환이 그 변환을 큰 묶음의 테스트 데이터에 적용하는 것 그리고 많은 작업 유닛에 대한 테스트 결과를 요약하는 로그를 리뷰하는 것을 가능하게 하도록 인코딩된다. 규칙 세트의 필수적 로직을 인코딩하는 것에 추가하여, 파일 테스트 모드 변환은 로깅 메시지(예를 들어, 각각의 실행 케이스가 작동되었던 횟수 및/또는 각각의 트리거 조건이 평가되고 통과되고 및/또는 실패된 횟수를 반영함)를 생성하는 코드를 포함할 수도 있다. 이러한 예에서, 열(220)은 수 천 개의 작업 유닛(예를 들어, 신용 카드 계정에 대한 기록)이 처리되었던 파일 테스트 모드 변환의 실행 도중에 실행 케이스가 작동했었던 횟수의 각각의 실행 케이스에 대한 카운트를 디스플레이한다.

[0083] 도 2b 는 스프레드쉬트-기초 규칙 엔트리에 대한 예시적인 GUI(250)의 예이다. GUI(250)는 도 2a 의 예시적인 GUI(200)에서 특정된 규칙에 기초한 변환을 사용하여 결정된 "보상 포인트" 값 및 신용 카드 계정과 연관된 다른 데이터에 기초하여 신용 카드 계정 보유자에게 제공될 제품 제공(product offerings)의 목록을 생성하는 다중-작동 규칙을 특정하기 위하여 사용자에 의하여 구성된 바 있다. GUI(250)는 그 규칙에 대한 5 개의 규칙 케이스를 특정하는 5 개의 행을 포함한다. GUI(250)는 그 규칙의 실행 케이스에 대한 트리거 조건 및 출력을 특정하고, 그리고 변환의 디버깅 또는 튜닝을 용이화하기 위하여 사용자에 의하여 사용될 수도 있는 규칙에 기초하여 변환의 테스트 실행으로부터의 메타데이터를 디스플레이하기 위하여 사용자에 의하여 사용된다. 예를 들어, GUI(250)는 도 6 의 애플리케이션 전문가 환경(622)을 통하여 사용자에게 제공될 수도 있다. 몇 가지 구현 형태들에서, GUI(250)를 통하여 특정된 규칙 세트는 도 6 의 실행 환경(604)의 네트워크 인터페이스를 통하여 수신될 수도 있다.

[0084] 제 1 열(254)은 신용 카드 계정에 대한 "보상 포인트"를 반영하는 변수에 적용되는 트리거 조건을 특정한다. 이러한 "보상 포인트" 값은 도 2a 의 GUI(200)에서 특정된 규칙에 기초하는 변환에 의하여 설정되고 데이터흐름 내의 기록에 기록될 수도 있다. 제 1 열의 제 2 및 제 3 행을 통과하는 다운-화살표(256)는 제 2 및 제 3 실행 케이스에 대한 제 1 트리거 조건이 제 1 실행 케이스에 대한 제 1 트리거 조건에 대한 트리거 조건과 동일하다는 것을 표시한다.

[0085] GUI(200)의 제 2 및 제 3 열(258)은 신용 카드의 타입 및 신용 카드 계정의 보유자의 거주국의 인구를 반영하는, 신용 카드 계정과 연관된 입력 데이터 기록으로부터 유도되는 다른 변수에 적용되는 트리거 조건을 특정한다. 열(258)은 한 쌍의 매칭 트리거 조건을 각각 표시하는 다운-화살표를 더 포함한다.

[0086] GUI(250)의 제 4 열(264)은 실행 케이스에 대한 트리거 조건의 모두가 평가되고 만족된다고 결정되는 경우 생성되는 출력들을 특정한다. 사용자가 GUI(250)를 통하여 이것이 다중-작동 규칙(268)이라는 것을 특정한 바 있기 때문에, 실행 케이스는 모두 평가될 수도 있다. 실행 케이스가 작동되면, 그러한 실행 케이스에 대하여 특정된 출력이 생성될 것이고 현재 작업 유닛에 대한 출력값들의 목록에 첨부될 수도 있다. 모든 실행 케이스가 평가된 바 있다면, 변환은 데이터흐름 내의 다음 작업 유닛의 처리를 시작하거나 종결할 수도 있다.

[0087] 제 5 열(270)은 변환의 추적된 테스트 실행으로부터의 메타데이터를 그 변환의 디버깅 또는 튜닝을 용이화하기

위하여 사용자에 의하여 사용될 수도 있는 규칙에 기초하여 디스플레이한다. 이러한 예에서, 열(270)은 수 천 개의 작업 유닛(예를 들어, 신용 카드 계정에 대한 기록)이 처리되었던 파일 테스트 모드 변환의 실행 도중에 실행 케이스가 작동했었던 횟수의 각각의 실행 케이스에 대한 카운트를 디스플레이한다.

[0088] 도 3a 는 규칙 세트에 대한 고유한 트리거 조건의 목록(300)의 일 예를 예시한다. 몇 가지 구현형태들에서, 하나 이상의 변환(들)이 규칙 세트에 기초하여 생성되는 경우, 변환 생성 프로세스의 일부는, 변환을 위한 제어 구조에 의하여 참조될 수도 있는 고유한 트리거 조건들의 목록을 생성하여 그렇지 않으면 그들이 발생하는 각각의 실행 케이스에 대한 트리거 조건의 사본 카피를 저장하기 위하여 사용될 수도 있는 메모리를 절약할 수도 있다. 몇 가지 구현형태들에서, 규칙 사양에 기초하여 변환을 생성할 수도 있으며 목록(300)을 포함하는 규칙의 일부는 그 규칙을 특정하는 애플리케이션 전문가에 의하여 사용되는 데이터에 대한 명칭을 변환 인코딩에서 사용되는 기술적 백엔드 명칭(backend names)으로 변환하는 것을 포함할 수도 있다. 예를 들어, 변환 동작은 고정된 키 또는 변수명의 다른 매핑에 기초하여 수행될 수도 있다. 이러한 예에서, 규칙 세트는 도 2a 의 GUI(200)를 통하여 특정된 규칙(규칙 1) 및 도 2b 의 GUI(250)를 통하여 특정된 규칙(규칙 2)으로 이루어진다.

[0089] 고유한 트리거 조건의 목록은 그 규칙 세트에서 한 번 이상 발생하는 각각의 트리거 조건의 단일 카피를 포함할 수도 있다. 이러한 예에서, 각각의 트리거 조건은 데이터 조작 언어(data manipulation language; DML) 표현으로서 인코딩된다. 트리거 조건의 DML 인코딩이 도 3a 의 제 1 열(310)에서 예시된다. 트리거 조건에 대한 다른 인코딩 포맷들도 가능하다(예를 들어, C, C++, 자바(Java), 또는 코볼(Cobol) 코드).

[0090] 몇 가지 구현형태들에서, 고유한 트리거 조건의 목록은 변환 내의 트리거 조건의 발생의 역방향 룩업을 용이화하는 사용 포인터(usage pointer)의 목록을 더 포함할 수도 있다. 예를 들어, 도 3a 의 제 2 열(320)은 규칙 1 및 규칙 2 를 포함하는 규칙 세트에 대한 사용 포인터의 목록을 예시한다. 각각의 포인터는 rule_id(규칙 1 또는 규칙 2), row_id(예를 들어, 특정 실행 케이스에 대응함), 및 column_id(예를 들어, 그 실행 케이스 내의 트리거 조건 시퀀스 포지션에 대응함)를 식별하는 숫자들의 삼중항이다. 이러한 예에서, 첫 번째 다섯 개의 고유한 트리거 조건은 규칙 1 에서 발생하고 다음 일곱 개의 고유한 트리거 조건은 규칙 2 에서 발생한다. 몇 가지 구현형태들에서, 사용 포인터는 고유한 트리거 조건들의 목록에 포함되지 않는다.

[0091] 몇 가지 구현형태들에서, 목록(300)은 목록(300) 내의 표현에 의하여 특정되는 복잡한 계산의 결과를 캐싱하기 위한 데이터 구조를 포함할 수도 있고, 따라서 결과가 재사용될 수도 있고 동일한 입력에 기초한 결과의 재계산은 입력 데이터가 변환의 실행 도중에 같다고 인식되는 경우에는 회피될 수도 있다.

[0092] 고유한 트리거 조건의 목록(300)은 매우 다양한 포맷 또는 데이터 구조에서 저장될 수도 있다(예를 들어, 링크된 목록 또는 인덱스된 어레이로서). 이러한 예에서, 고유한 트리거 조건의 목록(300)은 인덱스된 어레이로서 저장되어 변환에 대한 제어 구조 내의 이것의 인덱스에 의한 트리거 조건으로의 참조에 기초하여 트리거 조건의 룩업을 용이화한다.

[0093] 도 3b 는 규칙 세트에 대한 고유한 출력의 목록의 일 예를 예시한다. 몇 가지 구현형태들에서, 하나 이상의 변환(들)이 규칙 세트에 기초하여 생성되는 경우, 변환 생성 프로세스의 일부는, 변환을 위한 제어 구조에 의하여 참조될 수도 있는 고유한 출력들의 목록을 생성하여 그렇지 않으면 그들이 발생하는 각각의 실행 케이스에 대한 출력들의 사본 카피를 저장하기 위하여 사용될 수도 있는 메모리를 절약할 수도 있다. 이러한 예에서, 규칙 세트는 규칙 1 및 규칙 2 로 이루어진다.

[0094] 고유한 출력들의 목록은 그 규칙 세트에서 한 번 이상 발생하는 각각의 출력의 단일 카피를 포함할 수도 있다. 이러한 예에서, 각각의 출력은 데이터 조작 언어 표현으로서 인코딩된다. 그 출력들의 DML 인코딩이 도 3b 의 제 1 열(360)에서 예시된다. 출력에 대한 다른 인코딩 포맷들도 가능하다(예를 들어, C, C++, 자바(Java), 또는 코볼(Cobol) 코드).

[0095] 몇 가지 구현형태들에서, 고유한 출력들의 목록은 변환 내의 출력의 발생의 역방향 룩업을 용이화하는 사용 포인터의 목록을 더 포함할 수도 있다. 예를 들어, 도 3b 의 제 2 열(370)은 규칙 1 및 규칙 2 를 포함하는 규칙 세트에 대한 사용 포인터의 목록을 예시한다. 이러한 예에서, 첫 번째 네 개의 고유한 출력들은 규칙 1 에서 발생하고 다음 다섯 개의 고유한 출력들은 규칙 2 에서 발생한다. 몇 가지 구현형태들에서, 사용 포인터는 고유한 출력들의 목록에 포함되지 않는다.

[0096] 고유한 출력들의 목록(350)은 매우 다양한 포맷 또는 데이터 구조에서 저장될 수도 있다(예를 들어, 링크된 목록 또는 인덱스된 어레이로서). 이러한 예에서, 고유한 출력의 목록(350)은 고유한 트리거 조건의 목록(300)과 공동으로 인덱스되는(예를 들어, 해제 인덱스 값 간격(disjoint index value intervals)으로) 인덱스된 어레이

로서 저장된다.

[0097] 변환 생성 프로세스의 일부는 변환의 실행 흐름을 제어하는 제어 구조의 생성이고 고유한 트리거 조건들의 목록 및/또는 고유한 출력들의 목록을 참조할 수도 있다. 본 명세서에서, 용어 "제어 구조"는 매우 다양한 인코딩 포맷을 지칭하고 이중 인덱스된 2 차원의 어레이로 한정되지 않는다. 예를 들어, 도 4a 및 도 4b 의 비순환성 디렉팅된 그래프 및 도 5 에서 예시되는 이중으로 링크된 목록은 변환의 실행 흐름을 제어하는 제어 구조의 예들이다. 제어 구조는 실행 케이스들에 대응하는 행들을 가진다. 변환 제어 구조의 콘텍스트에서, 용어 "행"은 하나 이상의 트리거 조건 및 하나 이상의 출력의 논리적 그루핑을 지칭하는데, 여기에서 그 행 내의 트리거 조건 모두가 만족되는 것으로 결정된 바 있으면, 그 행에 대한 출력(들)이 실행된다. 이러한 콘텍스트에서 용어 "행"은 디스플레이된 테이블의 수평 서브세트로 한정되지 않는다.

[0098] 도 4a 및 도 4b 는 비순환성 디렉팅된 그래프로서 표현되는 규칙 세트에 기초한 변환에 대한 예시적인 제어 구조(400)를 도시한다. 이러한 예에서, 변환이 기초하는 규칙 세트는 규칙 1 및 규칙 2 를 포함한다. 이러한 예에서, 변환은 데이터흐름 그래프(100)의 컴포넌트(150) 내에서 구현될 수도 있고, 또는 등가적으로, 변환은 규칙 2 에 기초하며 컴포넌트(140)에서 구현되는 제 2 변환과 직렬인 규칙 1 에 기초하며 컴포넌트(130)에 구현되는 제 1 변환으로서 구현될 수도 있다. 도 4a 및 도 4b 에서, 각각의 노드는 도 3a 와 관련하여 설명된 바와 같은 사용 포인터(rule_id, row_id, column_id)에 의하여 명명된다.

[0099] 제어 구조(400) 내의 노드들은 트리거 조건 또는 출력에 대응한다. 하나의 트리거 조건에 대응하는 노드는 그 노드로부터 나오는 두 개의 에지를 가진다. 이러한 두 개의 에지 중 하나는 그 노드에 대한 대응하는 트리거 조건이 입력 데이터에 적용될 때 참이라고 결정되는 경우에 추종된다. 이러한 두 개의 에지 중 두 번째는 그 노드에 대한 대응하는 트리거 조건이 입력 데이터에 적용될 때 거짓이라고 결정되는 경우에 추종된다. 하나의 출력에 대응하는 노드는 대응하는 출력이 생성된 이후에 언제나 추종되는, 그 노드로부터 나오는 하나의 에지를 가진다. 제어 구조(400) 내의 행은 이전의 트리거 조건 노드로부터 나오는 "참" 에지에 의하여 연속적으로 연결되는 하나 이상의 트리거 조건 노드의 시퀀스를 포함할 수도 있다. 어느 행에 대한 시퀀스 내의 마지막 트리거 조건 노드는 자신의 "참" 에지에 의하여 그 행에 대한 하나 이상의 출력 노드들 중 첫 번째 것으로 연결될 수도 있다. 어느 행 내의 출력 노드로부터 나오는 에지는 그 행 내의 추가적 출력 노드에 연결될 수도 있다. 어느 행 내의 마지막 출력 노드로부터 나오는 에지는 상이한 실행 케이스 또는 규칙에 대응하는 다른 행 내의 어느 노드로 연결될 수도 있고, 또는 실행 흐름을 작업 유닛에 대한 변환 처리의 끝(448)으로 디렉팅할 수도 있다. 어느 행 내의 트리거 조건으로부터 나오는 "거짓" 에지는 상이한 실행 케이스 또는 규칙에 대응하는 다른 행 내의 어느 노드로 연결될 수도 있고, 또는 실행 흐름을 작업 유닛에 대한 변환 처리의 끝(448)으로 디렉팅할 수도 있다.

[0100] 제어 구조(400)는 변환의 실행 흐름에 대한 기동 포인트(402)를 인코딩한다. 변환은 제 1 실행 케이스(404)에 대한 제 1 트리거 조건을 평가함으로써 시작한다.

[0101] 몇 가지 경우들에서, "거짓" 에지는 실행 흐름이 현재의 행에 인접하지 않은 행으로 점프하고 그렇게 함에 있어서 몇몇 실행 케이스의 평가를 스킵하도록 할 수도 있다. 행들을 스킵하는 것은, 이것이 작업의 유닛(예를 들어, 계정, 트랜잭션, 항공사, 차량, 컴퓨터, 모바일 디바이스, 빌딩, 등을 나타낼 수도 있는 입력 데이터 기록에 대응함)을 처리하기 때문에 복잡도를 감소시키고 변환의 처리 시간을 감소시킬 수도 있다.

[0102] 이 예에서, 트리거 조건 노드(420)를 포함하는 노드들 중 일부는 그들로 연결되는 에지를 가지지 않는다. 어느 노드로 연결되는 에지가 부족하다는 것은 대응하는 트리거 조건 또는 출력이 그 변환이 기초하는 규칙 세트의 로직 하에서는 처리될 필요가 전혀 없을 것이라는 것을 반영한다. 결과적으로, 이러한 무연결 노드 및 그들의 대응하는 트리거 조건들은 제어 구조(400)로부터 생략될 수도 있다. 불필요한 트리거 조건의 이러한 생략이 도 4b 의 제어 구조(450)에서 예시된다.

[0103] 몇 가지 경우들에서, 노드들은 결합되고 제어 구조 내에서 단일 노드에 의하여 표현될 수도 있다. 예를 들어, 노드(454 및 456)는 노드(454)에 의하여 참조되는 트리거 조건 및 노드(456)에 의하여 참조되는 트리거 조건의 논리적 AND의 평가를 디렉팅하는 단일 트리거 조건 노드에 의하여 결합될 수도 있다. 이것은 양자의 트리거 조건이 참이면 출력 노드(458)가 다음에 처리되고 이러한 트리거 조건들 중 어느 것이 거짓이면 노드(460)에 대한 트리거 조건이 그 다음에 처리되기 때문에 가능하다. 이러한 방식으로, 상기 규칙 세트 내의 실행 케이스에 대한 적어도 두 개의 트리거 조건들은 결합되고 상기 제어 구조(450) 내의 단일 트리거 조건에 의하여 표현될 수도 있다. 만일 결합된 노드들이 이러한 트리거 조건 중 유일한 실례에 대응한다면, 트리거 조건은 고유한 트리거 조건의 목록 내의 단일 엔트리 내에 역시 결합될 수도 있다. 이와 유사하게, 출력 노드(470 및 472)는 그들

의 대응하는 출력들이 언제나 함께 생성되기 때문에 결합될 수도 있다. 이러한 예에서, 출력 노드(470 및 472)를 결합함으로써 그들의 대응하는 행들이 그 변환이 기초하고 있는 규칙 세트로부터의 두 개의 실행 케이스에 대응하는 단일 행으로 결합될 수도 있다. 이러한 노드 결합 기법은 생성되고 실행되어야 하는 제어 흐름 코드의 양을 감소시킴으로써 변환에 대한 메모리 사용 및 처리 시간을 감소시킬 수도 있다.

[0104] 도 5 는 디스플레이된 테이블로서 표현되는 규칙 세트에 기초한 변환에 대한 예시적인 제어 구조(500)를 도시한다. 제어 구조(500)는 예를 들어, 이중으로 링크된 목록(예를 들어, 행들의 링크된 목록으로서, 각각의 행이 그 행에 대한 트리거 조건 및 출력을 포함함)으로서 저장될 수도 있다. 이러한 예에서, 제어 구조(500)는 변환에 대한 실행 흐름을 규칙 1 및 규칙 2 를 포함하는 규칙 세트에 기초하여 제어한다. 이러한 예에서, 제어 구조(500)는 고유한 트리거 조건의 목록(300) 내의 트리거 조건 및 고유한 출력의 목록(350) 내의 출력을 참조한다.

[0105] 도 5 의 제 1 열(510)은 제어 구조(500)의 행들에 대한 인덱스(예를 들어, 실행 케이스 번호)를 나타낸다. 이러한 인덱스는 제어 구조 내의 행을 참조하여 실행 흐름에서의 점프를 용이화하여 불필요한 처리를 회피하기 위하여 사용될 수도 있다. 제 2 열(520)은 각각의 행에 대한 트리거 조건들의 시퀀스를 나타낸다. 예를 들어, 제어 구조(500)는 고유한 트리거 조건의 목록(300) 내의 트리거 조건을 참조하고 더 나아가 트리거 조건이 결과에 기초하여 평가된 이후에(예를 들어, 통과 또는 실패/ 참 또는 거짓) 변환의 처리를 더욱 디렉팅하는 트리거 조건에 대한 코드의 일부를 포함할 수도 있다. 트리거 조건이 실패하면, 변환의 처리는 제어 구조(500) 내의 시퀀스 또는 행 내의 현재의 행으로부터 두 개 이상의 행만큼 떨어져 있을 수도 있는 상이한 행으로 디렉팅된다. 예를 들어, 제 3 행 내의 코드(522)의 일부는 고유한 트리거 조건의 목록(300) 내의 제 3 트리거 조건을 평가하기 위하여 처리를 디렉팅하고, 트리거 조건이 실패하면, 코드(522)의 일부는 처리를 제어 구조(500)의 제 5 행으로 디렉팅하여 제 4 행을 지나 스킵한다.

[0106] 만일 제어 구조 내의 한 행에 대해 목록화된 트리거 조건들 모두가 통과된다면, 처리는 그 행에 대한 출력으로 디렉팅된다. 도 5 의 제 4 열(530)은 고유한 출력의 목록(350) 내의 출력으로의 참조를 나타낸다. 마지막으로, 제어 구조(500) 내의 한 행에 대한 출력의 실행 이후에, 처리는 제어 구조(500)의 다른 행에서 계속하도록 또는 현재의 작업 아이템에 대한 처리를 종료하도록 디렉팅될 수도 있다. 도 5 의 제 3 열(540)은 제어 구조에 대한 행 인덱스로 참조되는 그 제어 구조(500) 내의 다른 행으로의 참조들(예를 들어, 실행 케이스 번호)을 나타낸다.

[0107] 도 6 은 변환 생성 기법이 사용될 수 있는 예시적인 데이터 처리 시스템(600)을 도시한다. 시스템(600)은 스토리지 디바이스와 같은 데이터의 하나 이상의 소스 또는 온라인 데이터 스트림으로의 접속을 포함할 수도 있는 데이터 소스(602)를 포함하는데, 이들 각각은 데이터를 다양한 스토리지 포맷 중 임의의 것 내에 저장할 수도 있다(예를 들어, 데이터베이스 테이블, 스프레드쉬트 파일, 평텍스트 파일, 또는 메인프레임에 의하여 사용되는 네이티브 포맷). 실행 환경(604)은 변환 생성 모듈(606) 및 실행 모듈(612)을 포함한다. 실행 환경(604)은 적합한 운영 체제, 예컨대 UNIX 운영 체제의 제어 하에 하나 이상의 범용 컴퓨터에 호스팅될 수도 있다. 예를 들어, 실행 환경(604)은 로컬이거나(예를 들어, SMP 컴퓨터와 같은 다중프로세서 시스템), 또는 국지적으로 분산형이거나(예를 들어, 클러스터 또는 MPP로서 커플링되는 다중 프로세서), 또는 원격의, 또는 원격으로 분산형이거나(예를 들어, 근거리 네트워크(LAN) 및/또는 광역 네트워크(WAN)를 통해 커플링되는 다중 프로세서), 또는 이들의 임의의 조합인 다중 중앙 처리 유닛(CPU)을 사용하는 컴퓨터 시스템의 구성을 포함하는 다중-노드 병렬 컴퓨팅 환경을 포함할 수 있다.

[0108] 변환 생성 모듈(606)은 실행 케이스들의 시퀀스를 포함하는 규칙 세트를 수신하고, 상기 실행 케이스들 중 적어도 하나는 하나 이상의 트리거 조건 및 상기 하나 이상의 트리거 조건이 모두 만족되는 경우 생성될 출력의 사양을 포함한다. 또한, 규칙 세트는 하나 이상의 출력을 포함하지만 트리거 조건이 부족한 실행 케이스를 포함하거나 등가적으로 언제나 참으로 평가되는(예를 들어, 예시적 규칙 1 내의 디폴트 실행 케이스) 트리거 조건을 가질 수도 있다. 예를 들어, 규칙 세트는 애플리케이션 전문가 환경(622)을 통하여 실행 환경(604)에 액세스하는 사용자(626)에 의하여 특정될 수도 있다. 몇 가지 구현형태들에서, 애플리케이션 전문가 환경은 사용자(626)에게 규칙 세트를 특정하기 위한 GUI를 제공하는 원격 컴퓨팅 디바이스에서 실행하는 클라이언트 소프트웨어를 포함한다. 예를 들어, 애플리케이션 전문가 환경(622)은 사용자가 도 2a 및 도 2b 와 관련하여 설명되는 바와 같이 스프레드쉬트-기초 GUI 내의 규칙 세트를 특정하도록 할 수도 있다. 몇 가지 구현형태들에서(미도시), 개발 환경(618) 및 애플리케이션 전문가 환경(622)은 결합되고 그러한 데이터흐름 그래프에서 인스턴스화된 변환에 대한 규칙 세트 사양 모두를 편집하는 단일 사용자 또는 사용자들의 그룹에 의하

여 접근될 수도 있다.

[0109] 변환 생성 모듈(606)은 하나 이상의 변환을 수신된 규칙 세트에 기초하여 생성한다. 하나 이상의 변환의 실행 흐름을 제어하는 제어 구조가 생성될 수도 있다. 제어 구조는 규칙 세트 내의 실행 케이스에 대응하는 행을 포함할 수도 있다. 각각의 행은 하나 이상의 트리거 조건의 시퀀스 및 그 실행 케이스에 대한 출력을 특정하는 정보를 포함할 수도 있다. 트리거 조건들 중 몇몇은 트리거 조건이 데이터를 변환하는 처리 도중에 실패할 경우 두 개 이상의 행 만큼 현재의 행 아래의 상이한 행에서 계속하도록 처리를 디렉팅하고, 따라서 몇몇 실행 케이스의 평가를 스킵하여 그 변환에 대한 요구된 처리 시간을 감소시킬 수도 있다.

[0110] 제어 구조는 생성된 변환(들)을 인코딩하는 임의의 다른 데이터와 함께 저장되거나 송신될 수도 있다. 예를 들어, 제어 구조를 포함하는 생성된 변환(들)은 데이터 스토리지 시스템(616)에 저장될 수도 있다. 몇 가지 구현형태들(미도시)에서, 변환 생성 모듈(606)은 애플리케이션 전문가 환경(622)의 일부로서 구현될 수도 있고 그 제어 구조를 포함하는, 생성된 변환(들)을 인코딩하는 데이터는 애플리케이션 전문가 환경(622)을 실행하는 원격 컴퓨팅 디바이스로부터 실행 환경(604)으로 송신될 수도 있다.

[0111] 실행 모듈(612)은 변환 생성 모듈(606)에 의하여 생성된 하나 이상의 변환을 사용하여 입력 데이터 기록을 처리하고 송신 또는 저장하기 위한 출력 데이터 기록을 생성한다. 그래프 기초 계산 내의 하나 이상의 변환이 생성되면, 변환(들)을 포함하는 계산은 실행 모듈(612)에 의하여 입력 데이터에 적용될 수도 있다. 실행 모듈(612)은 데이터 소스(602)로부터 데이터를 독출하고 실행 환경(604)에 액세스가능한 데이터 스토리지 시스템(616) 내에 저장될 수도 있는 출력 데이터 기록(614)을 생성한다. 예를 들어, 데이터 스토리지 시스템(616)은 데이터베이스 서버 및/또는 버전 제어 애플리케이션을 실행하는 서버를 포함할 수도 있다.

[0112] 몇 가지 구현형태들에서, 실행 모듈은 또한 실행 시간 및/또는 입력 데이터 기록의 처리 도중에 평가되는 트리거 조건에 대한 결과를 로깅한다. 예를 들어 이러한 로그는 변환 생성 모듈(606)에 의하여 사용되어 변환을 제어 구조 내의 트리거 조건의 순위를 변경함으로써 업데이트할 수도 있다.

[0113] 데이터 소스(602)를 제공하는 스토리지 디바이스는, 예를 들어 실행 환경(604)을 실행하는 컴퓨터에 연결된 스토리지 매체(예를 들어, 하드 드라이브(608))에 저장되는 실행 환경(104)에 국부적으로 속할 수도 있고, 또는 원격 접속을 거쳐 실행 환경(604)을 실행하는 컴퓨터와 통신하는 원격 시스템(예를 들어, 메인프레임(610))에 호스팅되는 실행 환경(604)에 대해 원격일 수도 있다.

[0114] 데이터 스토리지 시스템(616)도 역시 개발 환경(618)에 액세스가능한데, 여기에서 개발자(620)는 변환에 대응하는 컴포넌트를 포함할 수도 있는 그래프-기초 계산을 생성하고 관리할 수 있다. 이러한 변환들의 동작은 그래프-기초 계산이 적용될 애플리케이션의 전문화된 지식을 가질 수도 있는 사용자(예를 들어, 사용자(626))에 의하여 구성가능할 수도 있다. 몇 가지 구현형태들에서, 개발 환경(618)은 애플리케이션을 꼭지점들 사이의 디렉팅된 링크(작업 엘리먼트의 흐름을 나타냄)에 의하여 연결된 꼭지점들(컴포넌트 또는 데이터셋을 나타냄)을 포함하는 데이터흐름 그래프로서 개발하기 위한 시스템이다. 예를 들어, 이러한 환경은 발명의 명칭이 "Managing Parameters for Graph-Based Applications"인 미국 공개 특허 번호 제 2007/0011668 호에 더욱 상세하게 설명되며, 이것은 원용에 의해 본 명세서에 포함된다. 이러한 그래프-기초 계산을 실행하기 위한 시스템은 미국 특허 제 5,566,072 호인 "XECUTING COMPUTATIONS EXPRESSED AS GRAPHS"에서 설명되는데, 이것은 원용에 의해 본 명세서에 포함된다. 본 시스템에 따라서 만들어진 데이터흐름 그래프는 정보를 프로세스들 사이에서 이동시키기 위하여, 그리고 그 프로세스에 대한 실행 순서를 정의하기 위하여 그래프 컴포넌트에 의하여 표현되는 개개의 프로세스 내에 집어넣고 그리고 그 밖으로 끄집어내기 위한 방법을 제공한다. 이러한 시스템은 인터프로세스 통신 방법(예를 들어, 그래프의 링크에 따르는 통신 경로는 TCP/IP 또는 UNIX 도메인 소켓을 사용할 수 있거나 공유 메모리를 사용하여 데이터를 프로세스들 사이에서 전달할 수 있음) 알고리즘을 포함한다.

[0115] 실행 모듈(612)은 상이한 형태의 데이터베이스 시스템을 포함하는 다양한 타입의 시스템으로부터 데이터를 수신할 수 있다. 데이터는 널(null) 값을 포함하는 것도 가능하며 개별적인 필드("속성" 또는 "열"이라고도 불림)에 대한 값을 가지는 기록으로서 조직화될 수도 있다. 우선 데이터를 데이터 소스로부터 독출하면, 실행 모듈(612)은 통상적으로 그 데이터 소스 내의 기록들에 대한 몇몇 초기 포맷 정보와 함께 시작한다. 몇몇 상황에서, 데이터 소스의 기록 구조는 처음에 공지되지 않을 수도 있고 대신에 데이터 소스의 분석 이후에 결정될 수도 있다. 기록에 대한 초기 정보는 별개의 값을 나타내는 비트수, 기록 내의 필드의 순서, 및 그 비트에 의하여 표현되는 값의 타입(예를 들어, 스트링, 부호/무부호 정수)을 포함할 수 있다.

[0116] 도 7 은 예시적인 변환 생성 및 실행 프로세스(700)에 대한 흐름도를 도시한다. 예를 들어 프로세스(700)는 도

6 의 실행 환경(604)에 의하여 수행될 수도 있다.

[0117] 프로세스(700)는 규칙 세트가 수신될 때에 시작될 수도 있다(702). 규칙 세트는 실행 케이스들의 시퀀스를 포함할 수도 있다. 규칙 세트 내의 실행 케이스는 하나 이상의 트리거 조건 및 그 실행 케이스에 대한 하나 이상의 트리거 조건이 모두 만족될 경우 생성될 출력의 사양을 포함할 수도 있다. 몇 가지 구현형태들에서, 규칙 세트는 규칙 세트를 수신하는 처리 디바이스에 국지적으로 접속된 하드웨어(예를 들어, 컴퓨터 모니터 및 키보드 및/또는 마우스)를 포함하는 사용자 인터페이스(예를 들어, 텍스트 파일 편집기, 스프레드쉬트-기초 GUI, 또는 몇몇 다른 타입의 GUI)를 통하여 수신된다. 예를 들어, 규칙 세트는 도 6 의 실행 환경(604)의 사용자 인터페이스를 통하여 수신될 수도 있다. 몇 가지 구현형태들에서, 규칙 세트는 서버에 의하여 원격 처리 디바이스로부터 네트워크 인터페이스를 거쳐 수신된다. 예를 들어, 규칙 세트는 애플리케이션 전문가 환경(622)을 실행하는 원격 처리 디바이스로부터 실행 환경(604)의 네트워크 인터페이스를 통해 수신될 수도 있다.

[0118] 제어 구조를 포함하는 변환이 수신된 규칙 세트에 기초하여 생성된다(704). 제어 구조는 변환이 입력 데이터에 인가될 때 변환의 실행 흐름을 제어하기 위하여 사용될 수도 있다. 제어 구조는 생성된 변환의 다른 부분(예를 들어, 고유한 트리거 조건의 목록 및/또는 고유한 출력의 목록)을 참조할 수도 있다. 제어 구조는 다양한 포맷으로 인코딩될 수도 있다. 제어 구조 포맷들의 예는 다른 것들 중에서 실행 환경 내의 하나 이상의 프로세서에 대하여 컴파일된 실행가능한 파일, 런-타임에서 컴퓨터 언어 해석기 또는 컴파일러에 의하여 컴파일될 수도 있는 텍스트를 포함하는 텍스트 파일, 해석되거나 컴파일될 수도 있는 코드의 일부를 포함하는 텍스트 기록의 듀얼 인덱스된(2 차원의) 어레이, 도 4a 및 도 4b 의 비순환성 디렉팅된 그래프, 및 도 5 에 예시된 이중으로 링크된 목록을 포함한다.

[0119] 생성된 제어 구조는 규칙 세트 내의 하나 이상의 실행 케이스에 대응하는 행들을 포함할 수도 있다. 제어 구조 내의 행은 하나 이상의 트리거 조건들의 시퀀스 및 실행 케이스에 대한 출력을 특정하는 정보를 포함할 수도 있다. 제어 구조의 행은 하나 이상의 트리거 조건에 대응하는 실행 제어 흐름 코드 및 그 규칙 세트 내의 실행 케이스에 대한 하나 이상의 출력의 논리적 그루핑일 수도 있다. 몇 가지 구현형태들에서, 제어 구조의 행은 처리 디바이스가 트리거 조건을 검사하도록 하거나 출력을 생성하도록 변환의 실행을 디렉팅하는 코드의 일부의 시퀀스(예를 들어, 링크된 목록으로서 저장됨)를 포함한다. 트리거 조건에 대한 코드의 일부는 또한 변환의 실행을 현재의 트리거 조건의 평가의 결과에 기초하여 상이한 트리거 조건 또는 출력에 대응하는 코드의 다른 부분으로 디렉팅할 수도 있다. 제어 구조 내의 트리거 조건은, 데이터를 변환하려는 처리 도중에 실패되면, 행들의 시퀀스 내의 현재의 행으로부터 두 개 이상의 행만큼 떨어져 있는 상이한 행에서 계속하도록 처리를 디렉팅한다. 이러한 방식으로, 트리거 조건의 실행 및/또는 몇몇 행에 대한 출력은 스킵되어 작업 유닛(예를 들어, 입력 데이터 기록)에 대한 처리 시간을 감소시킬 수도 있다.

[0120] 몇 가지 구현형태들에서, 한 행에 대한 트리거 조건들의 시퀀스는 상기 시퀀스 내의 트리거 조건이 데이터의 처리 도중에 실패하면 처리가 디렉팅될 상이한 행의 행 번호에 기초하여 정렬될 수도 있다. 예를 들어, 어떤 행에 대한 트리거 조건들의 시퀀스는 한 행에 대한 트리거 조건들의 시퀀스 내의 더 이른 고장 조건의 경우에 제어 구조를 통한 큰 점프를 야기할 트리거 조건들을 적용시키고 반면에 그 행에 대한 트리거 조건들의 시퀀스 내의 더 늦는 고장 조건의 경우에 제어 구조를 통한 더 작은 점프를 야기할 트리거 조건들을 적용시킴으로써 변환 생성 프로세스 도중에 정렬될 수도 있다.

[0121] 몇 가지 구현형태들에서, 제어 구조의 행은 실행 케이스들의 시퀀스 내의 대응하는 실행 케이스 직전의 실행 케이스에서 역시 발생하는 대응하는 실행 케이스에 대한 트리거 조건을 생략하는 방식으로 생성된다(704). 이러한 생략은 변환을 컴파일 및/또는 실행하기 위한 메모리 요구 사항을 감소시킬 수도 있다.

[0122] 몇 가지 구현형태들에서, 규칙 세트에 대한 고유한 트리거 조건들의 목록 도 역시 그 변환의 일부로서 생성된다(704). 그 규칙 세트에 대한 고유한 트리거 조건의 목록(300)은 규칙 1 을 포함하고 규칙 2 는 생성될 수도 있는 고유한 트리거 조건들의 목록의 일 예이다. 예를 들어, 행 내의 트리거 조건의 상기 시퀀스가 코드의 부분의 시퀀스로서 저장되는 경우, 코드의 부분 중 하나는 처리를 상기 규칙 세트로부터의 고유한 트리거 조건의 목록 내에 인코딩된 트리거 조건으로 디렉팅할 수도 있다.

[0123] 몇 가지 구현형태들에서, 규칙 세트에 대한 고유한 출력들의 목록 도 역시 그 변환의 일부로서 생성된다(704). 그 규칙 세트에 대한 고유한 출력의 목록(350)은 규칙 1 을 포함하고 규칙 2 는 생성될 수도 있는 고유한 출력들의 목록의 일 예이다. 예를 들어, 행 내의 출력이 코드의 부분으로서 저장되는 경우, 코드의 부분은 처리를 상기 규칙 세트로부터의 고유한 출력의 목록 내에 출력으로 디렉팅할 수도 있다.

[0124] 몇 가지 구현형태들에서, 상기 제어 구조의 행은 상기 행에 대한 트리거 조건의 전부가 만족되고 그 행에 대한 출력(들)이 생성된 바 있는 경우에 다음에 처리될 상기 제어 구조 내의 상이한 행으로 처리를 디렉팅하는 코드의 일부를 포함할 수도 있다.

[0125] 예를 들어, 제어 구조를 포함하는 변환은 도 6 의 실행 환경(604)에서 실행 중인 변환 생성 모듈(606)에 의하여 생성될 수도 있다(704).

[0126] 제어 구조를 포함하는 생성된 변환은 저장될 수도 있고 그리고/또는 송신될 수도 있다(706). 몇 가지 구현형태들에서, 변환은 메모리 디바이스(예를 들어, 랜덤 액세스 메모리)에 저장되고 입력 데이터로 적용될 수도 있는 실행 모듈로 전달될 수도 있다. 예를 들어, 변환은 실행 환경(604)의 일부인 휘발성 메모리 디바이스 내의 변환 생성 모듈(606)에 의하여 저장될 수도 있는데(706), 이것은 도 6 의 실행 모듈(612)에 의하여 액세스될 수도 있다. 몇 가지 구현형태들에서, 변환은 비-휘발성 메모리(예를 들어, 데이터베이스 서버 또는 버전 제어 애플리케이션을 실행하는 서버)를 포함하는 데이터 스토리지 디바이스 내에 저장될 수도 있다. 예를 들어, 변환은 데이터 스토리지 시스템(616) 내에 저장될 수도 있다(706). 몇 가지 구현형태들에서, 변환은 원격 디바이스로 송신될 수도 있다(예를 들어, 전자 통신 네트워크를 통하여). 예를 들어, 변환은 애플리케이션 전문가 환경에서 실행 중인 변환 생성 모듈로부터 입력 데이터로 적용되기 위하여 원격 실행 환경으로 송신될 수도 있다(706).

[0127] 변환이 생성되고 변환을 실행할 처리 시스템에게 이용가능해지면, 변환은 입력 데이터에 적용될 수도 있다. 예를 들어 변환은 변환을 구현하는 하나 이상의 컴포넌트(예를 들어, 데이터흐름 그래프(100)의 컴포넌트(130 및 140))를 포함하는, 데이터흐름 그래프를 실행하는 처리 시스템에 의하여 액세스될 수도 있다. 입력 데이터는 하나 이상의 데이터 소스(예를 들어, 데이터 소스(602))로부터 수신될 수도 있다(708). 몇 가지 구현형태들에서, 입력 데이터는 전-처리되어(예를 들어, 결합 프로세스를 구현하는 컴포넌트(120)에 의하여) 그 변환을 구현하는 하나 이상의 컴포넌트로 전달될 수도 있는 데이터흐름에 대한 작업 유닛을 생성한다. 각각의 작업 유닛이 수신된 입력 데이터에 기초하여 준비되기 때문에, 이것은 변환으로 전달될 수도 있다. 몇 가지 구현형태들에서, 작업 유닛들의 그룹이 묶음 내의 변환으로 전달된다. 예를 들어, 입력 데이터는 도 6 의 실행 환경(604)의 네트워크 인터페이스를 통하여 수신될 수도 있다(708).

[0128] 변환은 실행되어(710) 수신된 입력 데이터를 처리한다. 몇 가지 구현형태들에서, 변환은 새로운 입력 데이터를 처리할 때에 런-타임에서 해석되고/되거나 컴파일된다. 예를 들어, 변환은 도 8 과 관련하여 설명된 프로세스(800)를 사용하여 실행되고(710) 입력 데이터로 적용될 수도 있다. 변환을 입력 데이터에 적용시키는 것은 트리거 조건을 제어 구조를 사용하여 결정된 시퀀스 내의 입력 데이터에 대하여 검사하는 것을 포함할 수도 있다. 예를 들어, 변환은 도 6 의 실행 환경(604)에서 실행 중인 실행 모듈(606)에 의하여 실행될 수도 있다(710).

[0129] 변환의 실행은 이용가능한 입력 데이터가 더 이상 없을 때까지(712) 계속할 수도 있다. 변환의 입력 데이터로의 적용의 결과를 반영하는 데이터가 저장될 수도 있다(714)(예를 들어, 데이터 싱크에 기록되는 출력 데이터 기록으로서). 저장된 결과 데이터는 제어 구조에 의하여 특정되는 출력(들)에 기초하여 생성된 바 있을 수도 있다. 예를 들어, 결과들은 도 6 의 데이터 스토리지 시스템(614) 내의 실행 모듈(612)에 의하여 저장될 수도 있다. 몇 가지 구현형태들에서(미도시), 제어 구조에 의하여 특정된 출력(들)에 기초하는 결과 데이터는 전자적 통신 네트워크를 거쳐(예를 들어, 실행 환경(604)의 네트워크 인터페이스를 통하여) 송신된다(예를 들어 애플리케이션 전문가 환경(622)으로).

[0130] 도 8 은 규칙 세트에 기초한 변환을 실행하기 위한 예시적인 프로세스(800)의 흐름도이다. 예를 들어, 프로세스(800)는 도 6 의 실행 환경(604)에서 실행 중인 실행 모듈(612)에 의하여 수행될 수도 있다.

[0131] 프로세스(800)는 입력 데이터로 인가될 변환에 대한 제어 구조를 취출(802)하는 것을 포함한다. 몇 가지 구현형태들에서, 제어 구조는 그 변환을 구현하는 데이터흐름 그래프 내의 컴포넌트가 데이터흐름 내의 작업 유닛으로 전달될 때에 취출된다(802). 몇 가지 구현형태들에서, 제어 구조는 메모리 디바이스(예를 들어, 랜덤 액세스 메모리)로부터 취출된다(802). 몇 가지 구현형태들에서, 제어 구조는 비-휘발성 메모리(예를 들어, 데이터베이스 서버 또는 버전 제어 애플리케이션을 실행하는 서버)로부터 취출된다(802). 몇 가지 구현형태들에서, 제어 구조는 런-타임에서 해석기 및/또는 컴파일러로 전달되어 실행을 위한 제어 구조를 준비한다.

[0132] 제 1 트리거 조건은 작업 유닛에 대한 입력 데이터에 대하여 검사된다(810). 예를 들어, 트리거 조건을 인코딩하는 DML 표현은 해석되고 실행되어 작업 유닛과 연관된 기록(들) 내의 임의의 참조 입력 데이터 필드에 액세스하고 트리거 조건의 로직을 적용함으로써 액세스된 데이터를 테스트할 수도 있다. 이러한 평가의 결과는 통과

또는 실패(참 또는 거짓)일 수도 있다.

[0133] 트리거 조건의 평가의 결과는 로깅될 수도 있다(예를 들어, 테스트, 디버깅, 또는 최적화를 위하여). 몇 가지 구현형태들에서, 트리거 조건에 대한 실행 시간(예를 들어, 마이크로초 또는 프로세서 사이클로 측정됨)은 로깅될 수도 있다. 트리거 조건이 입력 데이터로 인가될 때 그들에 대한 실패율 또는 실행 시간에 관련되는 데이터가 장래 기록들에 대한 평균 처리 시간을 감소시키기 위한 노력에서 제어 구조를 동적으로 업데이트하기 위하여 사용될 수도 있다.

[0134] 만일 트리거 조건이 작업 유닛에 대한 입력 데이터에 의하여 만족되지 않으면(815)(예를 들어, 결과가 실패이거나 거짓임), 변환의 실행은 부분적으로 트리거 조건과 연관된 제어 흐름 코드(예를 들어, 트리거 조건을 참조하거나 이것을 포함하는 코드의 부분)에 기초하여 제어 구조의 상이한 행으로 디렉팅될 수도 있다. 제어 구조에 따라서, 실행은 제어 구조의 상이한 행으로 점프할 수도 있다(820). 예를 들어, 몇몇 고장 조건은 실행이 제어 구조에 대한 행들의 시퀀스 내의 현재의 행으로부터 두 개 이상의 행 만큼 떨어진 상이한 행으로 점프(820)하도록 할 수도 있다. 이러한 방식으로, 트리거 조건의 실행 및/또는 몇몇 행에 대한 출력은 스킵되어 작업 유닛에 대한 처리 시간을 감소시킬 수도 있다. 그러면 새 행에 대한 다음 트리거 조건이 검사될 수도 있다(810). 몇 가지 구현형태들에서(미도시), 변환의 실행은 트리거 조건이 없는 행으로(예를 들어, 디폴트 실행 케이스에 대응) 또는 직접적으로 제어 구조의 종결로 점프할 수도 있다.

[0135] 작업 유닛에 대한 입력 데이터에 의하여 트리거 조건이 만족되면(815)(예를 들어, 결과가 통과 또는 참이면), 변환의 실행은 제어 구조의 현재의 행 내의 다음 엘리먼트로 디렉팅될 수도 있다. 만일 그 행에 대한 트리거 조건들의 시퀀스에 더 많은 트리거 조건이 있으면(825), 그 행 내의 다음 트리거 조건이 검사된다(810). 그렇지 않으면, 그 행에 대한 하나 이상의 출력이 생성될 수도 있다(830).

[0136] 예를 들어, 출력을 인코딩하는 DML 표현은 해석되고 실행되어 작업 유닛과 연관된 기록(들) 내의 임의의 참조 입력 데이터 필드에 액세스하고/하거나 출력의 로직을 적용함으로써 하나 이상의 출력 기록을 생성할 수도 있다(810). 결과적인 출력 기록(들)은 완전히 새로울 수도 있고, 또는 현존 기록이 추가적 필드 또는 다른 데이터를 포함하도록 업데이트되거나 확장될 수도 있다.

[0137] 현재의 행에 대한 출력(들)이 생성된(830) 이후에, 변환의 실행은 추가적 실행 케이스(들)에 대응하는 상이한 행으로 디렉팅될 수도 있다. 몇 가지 구현형태들에서, 행은 변환의 실행을 제어 구조 내의 상이한 행으로 디렉팅하는 포인터를 포함한다. 처리될 더 많은 실행 케이스가 존재한다면(835), 제어 구조는 변환의 실행이 제어 구조 내의 상이한 행으로 점프(840) 하도록 할 수도 있다. 예를 들어, 다중-작동 규칙에 대하여, 추가적 실행 케이스에 대응하는 추가적 행이 처리될 필요가 있을 수도 있다. 만일 변환이 다중 규칙을 가지는 규칙 세트에 대응한다면, 제어 구조는 변환의 실행이 상이한 규칙에 대응하는 제어 구조 내의 상이한 행으로 점프(840) 하도록 할 수도 있다. 그러면 새 행에 대한 다음 트리거 조건이 검사될 수도 있다(810).

[0138] 더 실행 케이스가 더 이상 없고, 따라서 처리될 필요가 있는 행도 더 이상 없으면(835), 제어 구조는 처리된 입력 데이터에 대한 로그 정보에 기초하여 동적으로 업데이트될 수도 있다. 예를 들어, 어느 행에 대한 트리거 조건들의 시퀀스 내의 트리거 조건은 트리거 조건에 대한 실행 시간의 평균 실패율에 대한 새 로그 정보에 부분적으로 기초하여 정렬될 수도 있다(850).

[0139] 몇 가지 구현형태들에서, 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실행 시간의 측정은 입력 데이터로써 트리거 조건을 실행시키는데 걸리는 시간에 기초하여 갱신될 수도 있다. 어느 행에 대한 트리거 조건들의 시퀀스 내의 트리거 조건은 트리거 조건에 대한 실행 시간의 업데이트된 측정에 부분적으로 기초하여 정렬될 수도 있다(850). 몇 가지 구현형태들에서, 고유한 트리거 조건의 목록 내의 트리거 조건에 대한 실패율은 트리거 조건이 입력 데이터 내의 하나 이상의 기록(들)에 의하여 만족되는지 여부에 기초하여 업데이트될 수도 있다. 어느 행에 대한 트리거 조건들의 시퀀스 내의 트리거 조건은 트리거 조건에 대한 실패율의 업데이트된 측정에 부분적으로 기초하여 정렬될 수도 있다(850).

[0140] 제어 구조의 업데이트된 버전은 장래 입력 데이터로의 적용을 위하여 저장될 수도 있다(852). 몇 가지 구현형태들에서, 업데이트된 제어 구조는 메모리 디바이스(예를 들어, 랜덤 액세스 메모리)에 저장될 수도 있다(852). 몇 가지 구현형태들에서, 업데이트된 제어 구조는 비-휘발성 메모리(예를 들어, 데이터베이스 서버 또는 버전 제어 애플리케이션을 실행하는 서버)를 포함하는 데이터 스토리지 디바이스 내에 저장될 수도 있다(852).

[0141] 위에서 설명된 변환 생성 접근법은 컴퓨터에서 실행되기 위한 소프트웨어를 사용하여 구현될 수 있다. 예를 들면, 소프트웨어는, 각각 적어도 하나의 프로세서, 적어도 하나의 데이터 스토리지 시스템(휘발성 및 비-휘발성

메모리 및/또는 스토리지 엘리먼트를 포함), 적어도 하나의 입력 디바이스 또는 포트, 및 적어도 하나의 출력 디바이스 또는 포트를 포함하는 하나 이상의 프로그래밍된 또는 프로그래밍가능한 컴퓨터 시스템(분산형, 클라이언트/서버, 또는 그리드와 같은 다양한 아키텍처일 수 있음)에서 실행하는 하나 이상의 컴퓨터 프로그램 내의 프로시저를 형성한다. 소프트웨어는, 예를 들어 데이터흐름 그래프의 디자인 및 구성에 관련된 다른 서비스를 제공하는 더 큰 프로그램의 하나 이상의 모듈을 형성할 수도 있다. 그래프의 노드 및 엘리먼트는 컴퓨터 판독가능 매체에 저장된 데이터 구조 또는 데이터 저장소 내에 저장된 데이터 모델에 따르는 다른 조직화된 데이터로서 구현될 수 있다.

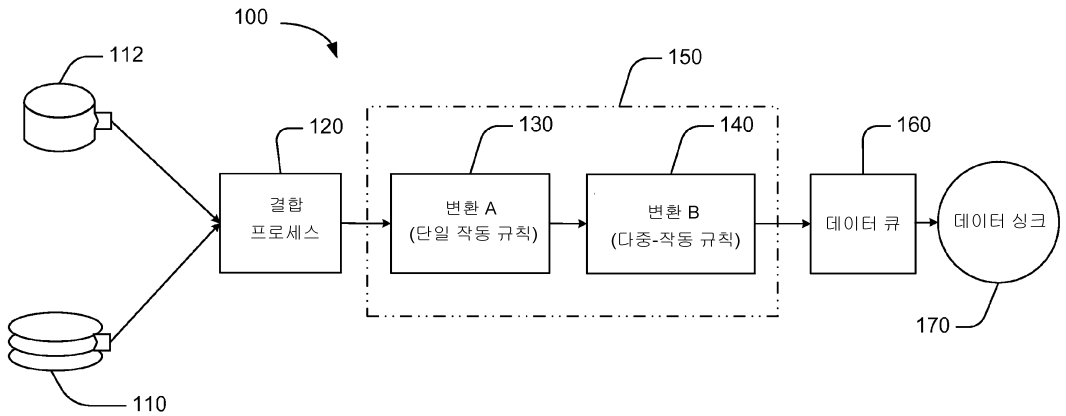
[0142] 소프트웨어는 스토리지 매체, 예컨대 번용 또는 주문형 프로그래밍가능한 컴퓨터에 의하여 독출가능한 CD-ROM과 같은 저장 매체에 제공되거나 네트워크의 통신 매체를 거쳐 실행되는 컴퓨터의 스토리지 매체로 배달(전파되는 신호로 인코딩됨)될 수도 있다. 기능들 모두는 특수 목적 컴퓨터에서, 또는 특수-목적 하드웨어, 예컨대 코프로세서를 사용하여 수행될 수도 있다. 소프트웨어는 소프트웨어에 의하여 특정된 계산의 상이한 부분들이 상이한 컴퓨터에 의하여 수행되는 방식인 분산 방식으로 구현될 수도 있다. 각각의 이러한 컴퓨터 프로그램은 바람직하게는 스토리지 미디어 또는 디바이스가 본 명세서에서 설명되는 프로시저를 수행하기 위하여 컴퓨터 시스템에 의해 판독될 때 컴퓨터를 구성하고 작동시키기 위하여, 범용 또는 특수 목적 프로그래밍가능한 컴퓨터에 의하여 판독가능한 유형의 비일시적 스토리지 매체 또는 디바이스(예를 들어, 매체의 고상 메모리(solid state memory) 또는 광학적 매체)에 저장되거나 다운로드된다. 진보적인 시스템은 또한 컴퓨터 프로그램으로써 구성된 컴퓨터-판독가능 저장 매체로서 구현될 수 있다고 간주될 수도 있는데, 여기에서 이와 같이 구성된 스토리지 매체는 컴퓨터 시스템이 본 명세서에서 설명된 기능들을 수행하기 위하여 특정하고 선정의된 방식으로 작동하도록 한다.

[0143] 본 발명의 다수 개의 실시예들이 설명되었다. 그럼에도 불구하고, 다양한 변형이 본 발명의 사상 및 범위에서 벗어나지 않으면서 이루어질 수도 있다는 것이 이해될 것이다. 예를 들어, 위에서 설명된 단계들 중 일부는 순서 독립적일 수도 있고, 따라서 설명된 것과 상이한 순서로 수행될 수 있다.

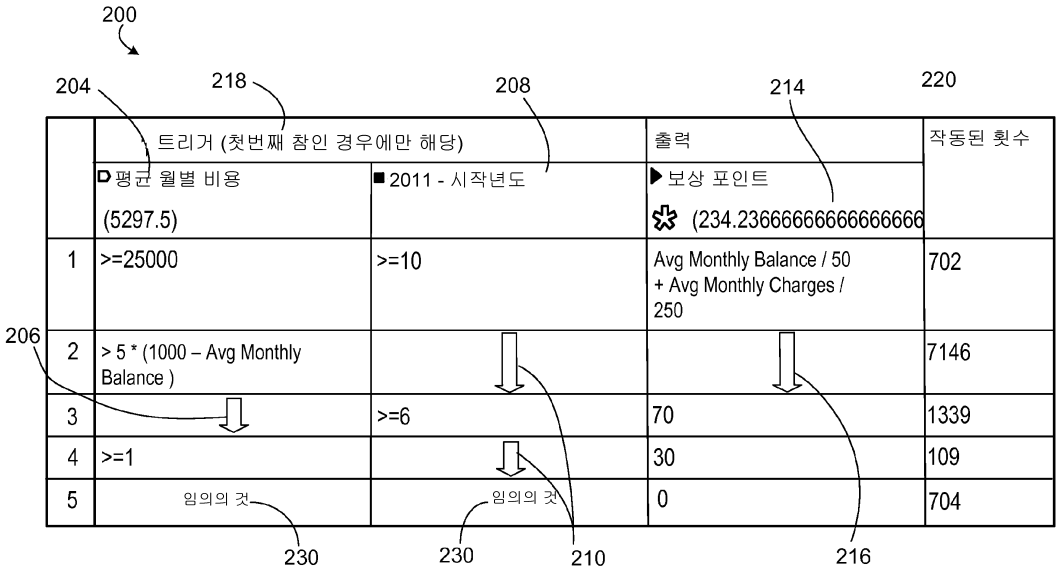
[0144] 앞선 설명이 예시하기 위하여 의도되고 첨부된 특허청구범위에 의하여 정의되는 본 발명의 범위를 한정하기 위한 것이 아님이 이해되어야 한다. 예를 들어, 위에서 설명된 다수 개의 기능 단계들은 전체 처리에 실질적으로 영향을 주지 않으면서 상이한 순서로 수행될 수도 있다. 도 2a 및 도 2b 의 예에서 설명되고 본 명세서 전체에 걸쳐 참조되는 신용 계정에 관련된 특정 비즈니스 규칙의 세부사항이 GUI(200) 및 GUI(250) 및 이들이 사용자 인터페이스를 제공하는 변환 생성 시스템의 성능을 예시하기 위한 것일 뿐이라는 것이 강조된다. 제공된 특정 비즈니스 규칙의 세부사항은 필수적 특징이 아니고 청구항들의 범위를 한정하는 것을 해석되어서는 안 된다. 다른 실시예들은 다음의 청구항들의 범위 내에 있다.

도면

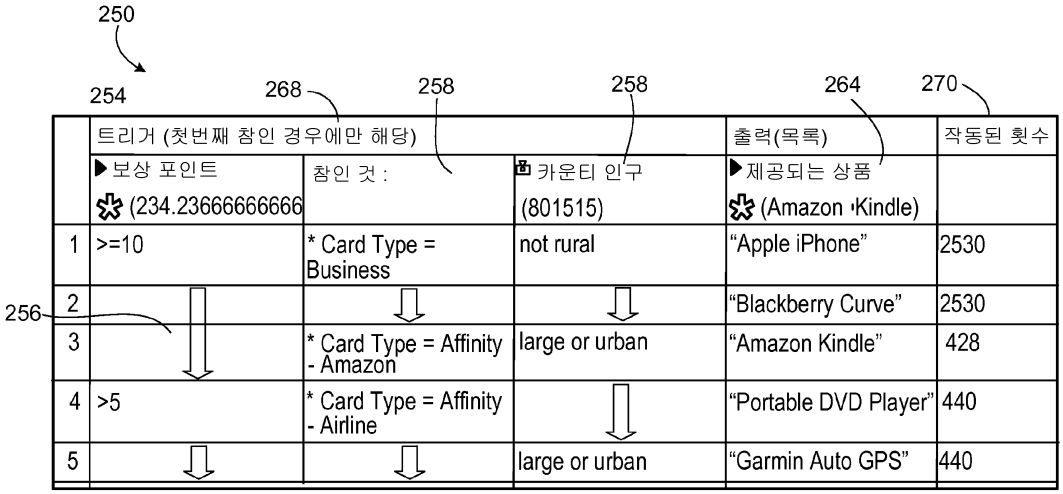
도면1



도면2a


200
204
218
208
214
220
206
트리거 (첫번째 참인 경우에만 해당)
출력
작동된 횟수
평균 월별 비용
(5297.5)
2011 - 시작년도
보상 포인트
(234.23666666666666666
1
>=25000
>=10
Avg Monthly Balance / 50
+ Avg Monthly Charges /
250
702
2
> 5 * (1000 – Avg Monthly
Balance)
7146
3
>=6
70
1339
4
>=1
30
109
5
임의의 것
임의의 것
0
704
230
230
210
216

도면2b


250
254
268
258
258
264
270
256
트리거 (첫번째 참인 경우에만 해당)
출력(목록)
작동된 횟수
보상 포인트
(234.23666666666
참인 것 :
카운티 인구
(801515)
제공되는 상품
(Amazon Kindle)
1
>=10
* Card Type =
Business
not rural
“Apple iPhone”
2530
2
“Blackberry Curve”
2530
3
* Card Type = Affinity
- Amazon
large or urban
“Amazon Kindle”
428
4
>5
* Card Type = Affinity
- Airline
“Portable DVD Player”
440
5
large or urban
“Garmin Auto GPS”
440

도면3a

300

310

320

	DML 표현	사용 포인터 (rule_id, row_id, col_id)
1	in0.avg_purchases >=25000	(1,1,1)
2	ColumnHeader24* >= 10	(1,1,2) (1,2,2)
3	in0.avg_purchases > 5 * (1000 - in0.avg_balance)	(1,2,1) (1,2,1)
4	ColumnHeader24 >= 6	(1,3,2) (1,4,2)
5	in0.avg_purchases >= 1	(1,4,1)
6	Award_Points >= 10	(2,1,1) (2,2,1) (2,3,1)
7	in0.card_type == "biz"	(2,1,2) (2,2,2)
8	bzt_3(in0).CENSUS2000POP) <=10000	(2,1,3) (2,2,3)
9	in0.card_type == "amzn"	(2,3,2)
10	sql_or(sql_and((bzt_3(in0).CENSUS2000POP>=60001),(bzt_3(in0).CENSUS2000POP<=100000)),(bzt_3(in0).CENSUS2000POP>=100001)))	(2,3,3) (2,4,3) (2,5,3)
11	Award_Points >5	(2,4,1) (2,5,1)
12	in0.card_type == "air"	(2,4,2) (2,5,2)

도면3b

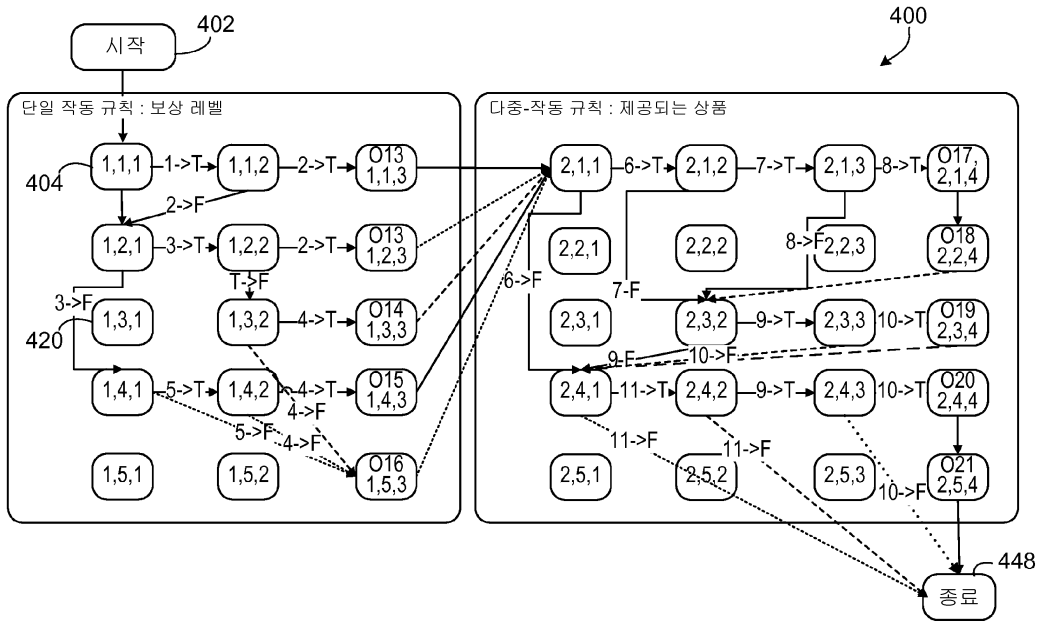
350

360

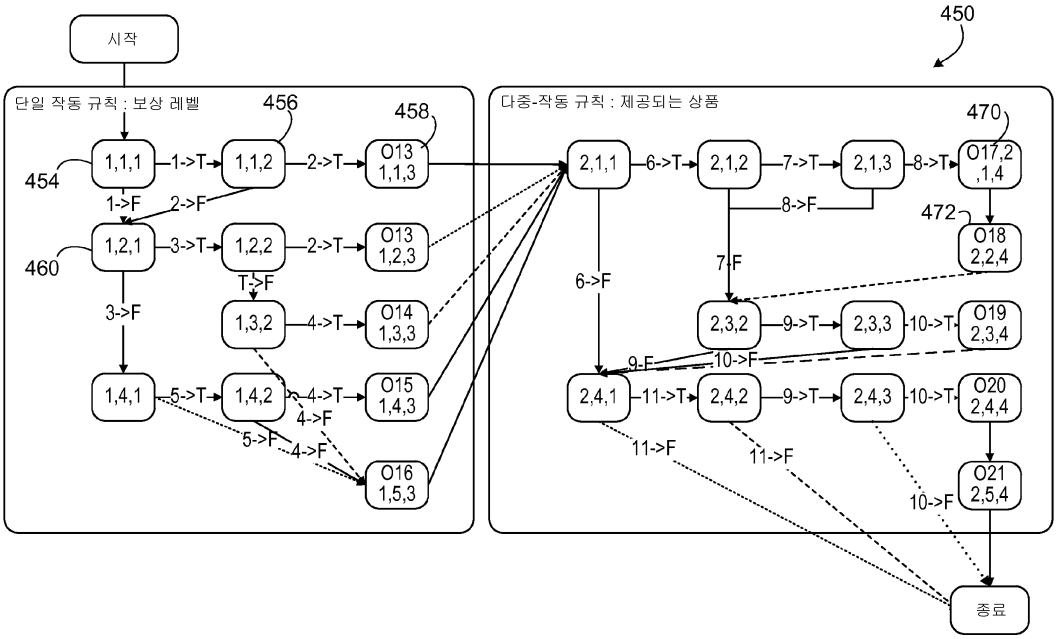
370

	DML 표현	사용 포인터 **(rule_id, row_id, col_id)**
13	Award_Points=in0.avg_balance / 50 + in0.avg_purchases / 250	(1,1,3),(1,2,3)
14	Award_Points=70	(1,3,3)
15	Award_Points=30	(1,4,3)
16	Award_Points=0	(1,5,3)
17	Products_Offered=vector_append(Products_Offered,"Apple iPhone");	(2,1,4)
18	Products_Offered=vector_append(Products_Offered,"Blackberry Curve");	(2,2,4)
19	Products_Offered=vector_append(Products_Offered,"Amazon Kindle");	(2,3,4)
20	Products_Offered=vector_append(Products_Offered,"Portable DVD Player");	(2,4,4)
21	Products_Offered=vector_append(Products_Offered,"Garmin Auto GPS");	(2,5,4)

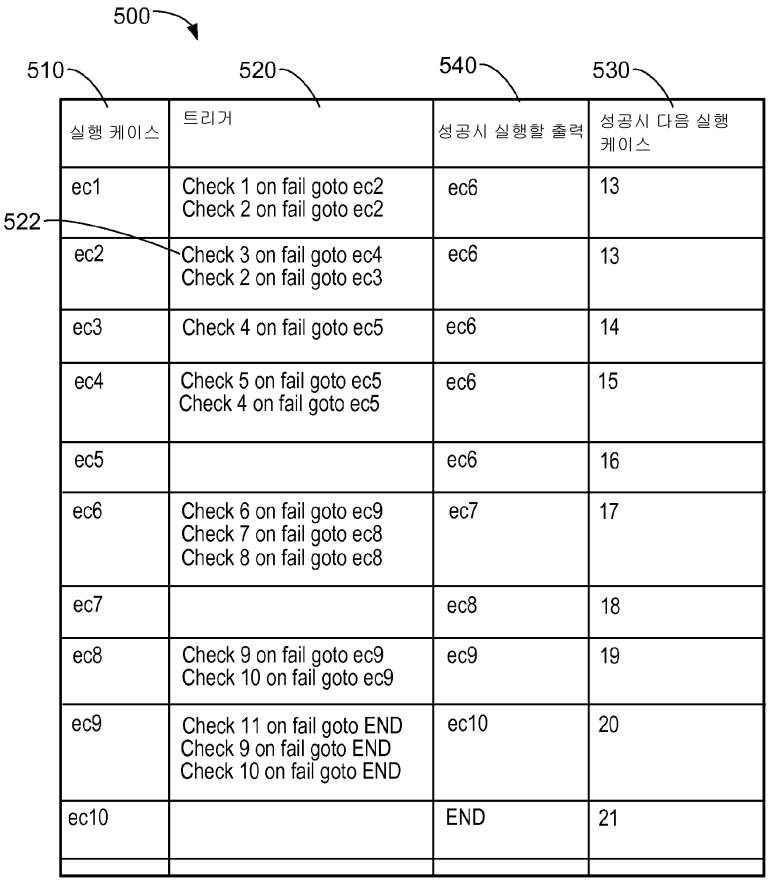
도면4a


시작
402
400
단일 작동 규칙 : 보상 레벨
다중-작동 규칙 : 제공되는 상품
404
420
1,1,1
1,1,2
1,2,1
1,2,2
1,3,1
1,3,2
1,4,1
1,4,2
1,5,1
1,5,2
O13 1,1,3
O13 1,2,3
O14 1,3,3
O15 1,4,3
O16 1,5,3
1-->T
2-->T
2-->F
3-->T
T-->F
3-->F
4-->T
5-->T
4-->F
5-->F
2,1,1
2,1,2
2,1,3
2,2,1
2,2,2
2,2,3
2,3,1
2,3,2
2,3,3
2,4,1
2,4,2
2,4,3
2,5,1
2,5,2
2,5,3
O17 2,1,4
O18 2,2,4
O19 2,3,4
O20 2,4,4
O21 2,5,4
6-->T
7-->T
8-->T
6-->F
7-F
8-->F
9-->T
10-->T
9-F
10-->F
11-->T
11-->F
10-F
448
종료

도면4b

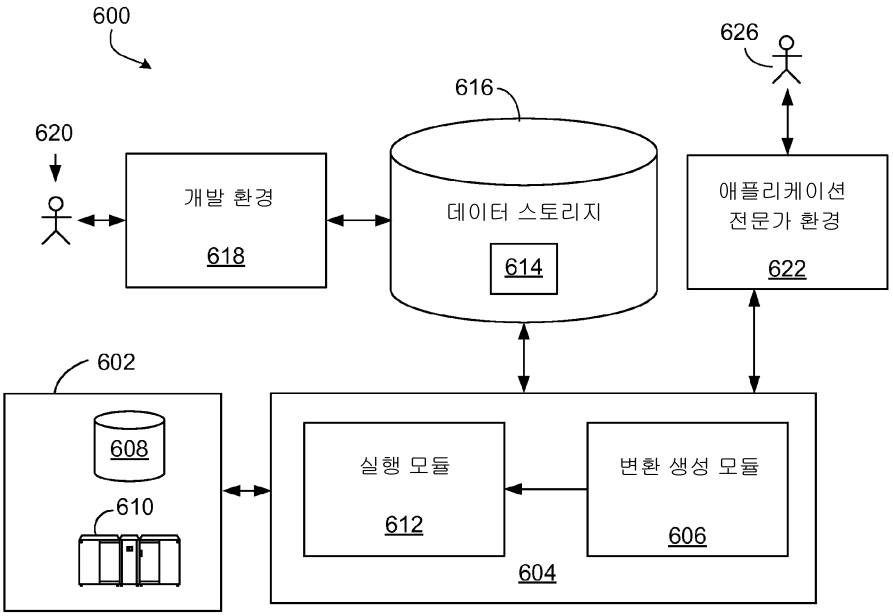

시작
450
단일 작동 규칙 : 보상 레벨
다중-작동 규칙 : 제공되는 상품
456
458
454
460
470
472
1,1,1
1,1,2
1,2,1
1,2,2
1,3,2
1,4,1
1,4,2
O13 1,1,3
O13 1,2,3
O14 1,3,3
O15 1,4,3
O16 1,5,3
1-->T
2-->T
1-->F
2-->F
3-->T
T-->F
3-->F
4-->T
5-->T
4-->F
5-->F
2,1,1
2,1,2
2,1,3
2,3,2
2,3,3
2,4,1
2,4,2
2,4,3
O17,2 ,1,4
O18 2,2,4
O19 2,3,4
O20 2,4,4
O21 2,5,4
6-->T
7-->T
8-->T
8-->F
6-->F
7-F
9-->T
10-->T
9-F
10-->F
11-->T
11-->F
10-F
종료

도면5

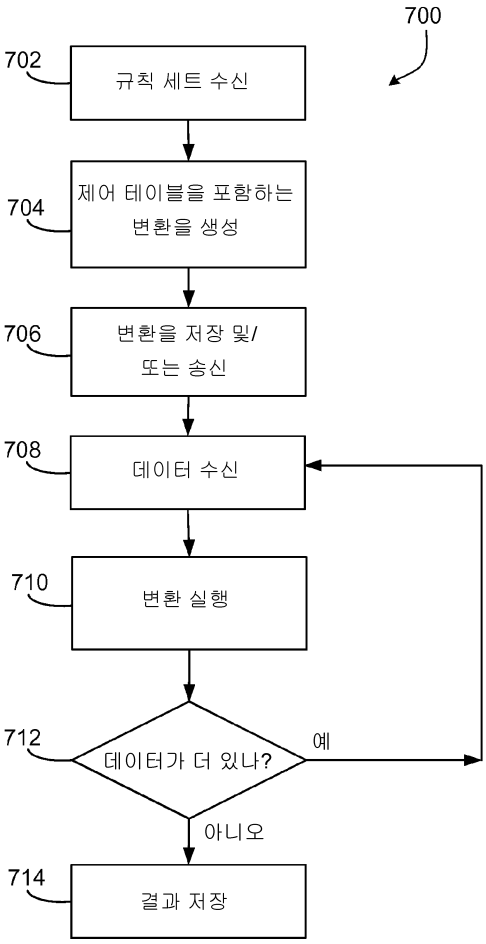


실행 케이스	트리거	성공시 실행할 출력	성공시 다음 실행 케이스
ec1	Check 1 on fail goto ec2 Check 2 on fail goto ec2	ec6	13
ec2	Check 3 on fail goto ec4 Check 2 on fail goto ec3	ec6	13
ec3	Check 4 on fail goto ec5	ec6	14
ec4	Check 5 on fail goto ec5 Check 4 on fail goto ec5	ec6	15
ec5		ec6	16
ec6	Check 6 on fail goto ec9 Check 7 on fail goto ec8 Check 8 on fail goto ec8	ec7	17
ec7		ec8	18
ec8	Check 9 on fail goto ec9 Check 10 on fail goto ec9	ec9	19
ec9	Check 11 on fail goto END Check 9 on fail goto END Check 10 on fail goto END	ec10	20
ec10		END	21

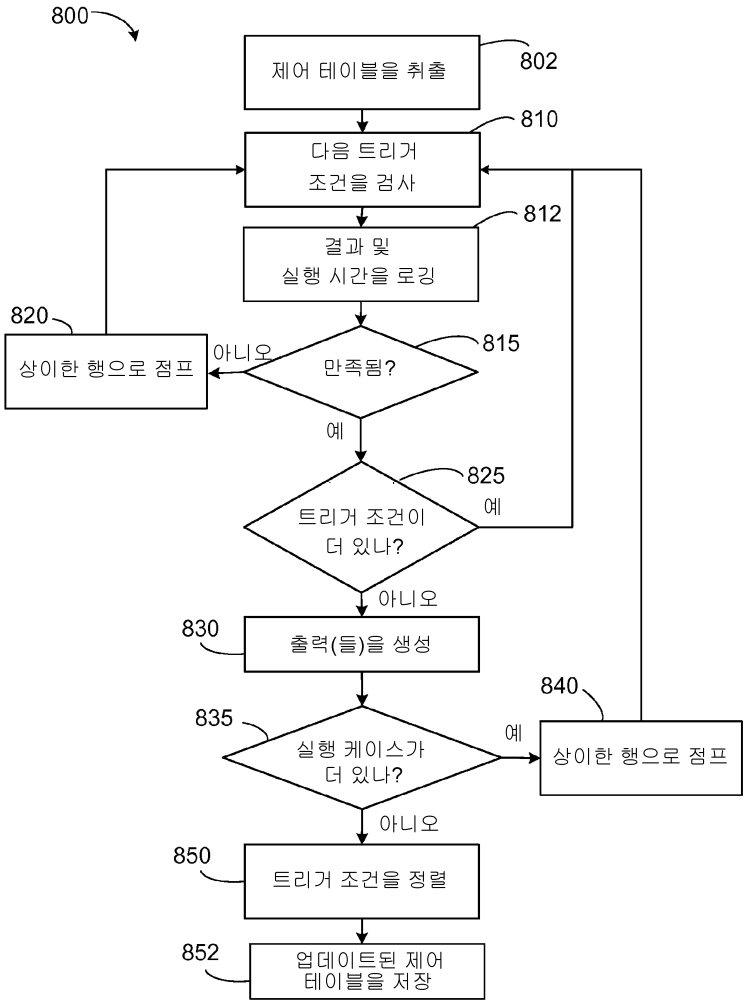
도면6



도면7


700
702
규칙 세트 수신
704
제어 테이블을 포함하는 변환을 생성
706
변환을 저장 및/또는 송신
708
데이터 수신
710
변환 실행
712
데이터가 더 있나?
예
아니오
714
결과 저장

도면8


800
제어 테이블을 취출
802
다음 트리거
조건을 검사
810
결과 및
실행 시간을 로깅
812
만족됨?
815
아니오
상이한 행으로 점프
820
예
트리거 조건이
더 있나?
825
예
아니오
출력(들)을 생성
830
실행 케이스가
더 있나?
835
예
상이한 행으로 점프
840
아니오
트리거 조건을 정렬
850
업데이트된 제어
테이블을 저장
852