(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2014-0143850
(43) 공개일자 2014년12월17일

(51) 국제특허분류(Int. Cl.)
G06F 17/30 (2006.01) *G06F 17/40* (2006.01)
(21) 출원번호 10-2014-7033288(분할)
(22) 출원일자(국제) 2009년06월30일
심사청구일자 없음
(62) 원출원 특허 10-2011-7002340
원출원일자(국제) 2009년06월30일
심사청구일자 2013년08월23일
(85) 번역문제출일자 2014년11월26일
(86) 국제출원번호 PCT/US2009/049299
(87) 국제공개번호 WO 2010/002919
국제공개일자 2010년01월07일
(30) 우선권주장
61/077,022 2008년06월30일 미국(US)

(71) 출원인
아브 이니티오 테크놀로지 엘엘시
미국 02421 매사추세츠주 렉싱턴 스프링 스트리트 201
(72) 발명자
스튜더 스캇
미국 01833 매사추세츠주 조지타운 필즈베리 레인 27
구드 조엘
미국 02474 매사추세츠주 알링턴 리 테라스 27
필리모어 데이비드
미국 01720 매사추세츠주 액턴 잭슨 드라이브 41
(74) 대리인
유미특허법인

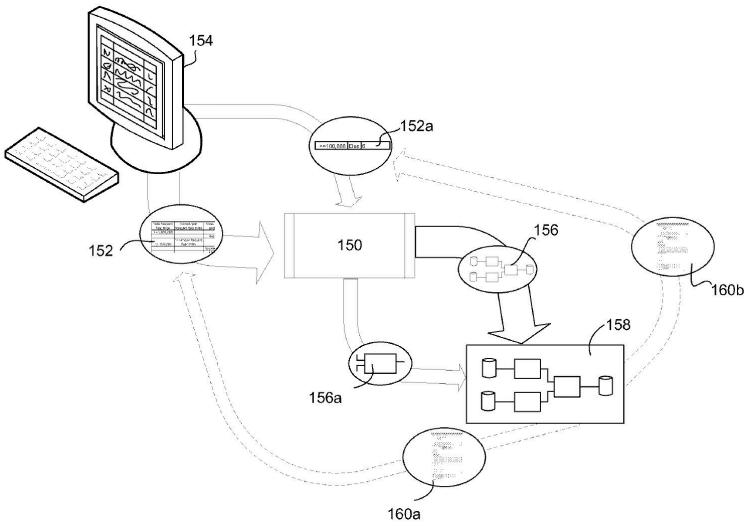
전체 청구항 수 : 총 99 항

(54) 발명의 명칭 **그래프 기반 계산에서의 데이터 로깅**

(57) 요 약

데이터 흐름을 나타내는 링킹 요소에 의해 연결된 데이터 처리 구성요소를 가지는 그래프 기반 계산에 대한 하나 이상의 규칙 명세를 수신한다. 상기 규칙 명세는 입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례와 각각 연관된 규칙들을 정의한다. 상기 수신된 규칙 명세에 기초하여 상기 그래프 기반 계산 내의 하나 이상의 데이터 처리 구성요소에 대한 변환식을 생성하며, 상기 생성된 변환식과 연관된 로그의 특성을 설정하기 위한 인터페이스의 제공을 포함한다. 상기 생성된 변환을 사용하여 하나 이상의 데이터 흐름을 변환하며, 실행시에 상기 그래프 기반 계산 내의 상기 데이터 처리 구성요소에 대한 실행의 트레이싱, 상기 설정된 로그 특성에 따라 상기 트레이싱된 실행에 기초하여 로그 정보의 생성, 및 상기 생성된 로그 정보의 저장 또는 출력을 포함한다.

대 표 도



특허청구의 범위

청구항 1

입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하는 하나 이상의 규칙 명세(rule specification)를 수신하는 단계;

상기 규칙 명세와 연관되어 있는 로그(log)의 특성들을 설정하기 위한 인터페이스를 제공하는 단계로서, 상기 특성들은 로그 정보를 생성하기 위한 조건을 식별하는, 인터페이스를 제공하는 단계;

수신된 상기 규칙 명세에 기초하여 변환식(transform)을 생성하는 단계; 및

출력 레코드를 생성하기 위해 생성된 상기 변환식을 사용하여 입력 데이터를 변환하는 단계

를 포함하고,

상기 입력 데이터를 변환하는 단계는,

하나 이상의 조건이 발생했는지 여부를 결정하는 단계,

상기 변환식을 실행시에 상기 변환식의 실행을 트레이싱(tracing)하는 단계,

상기 하나 이상의 조건이 발생했다는 결정에 응답하여, 트레이싱된 상기 실행에 기초하여 출력 레코드에 포함되지 않는 정보를 포함하는 로그 정보를 생성하는 단계로서, 로그 정보를 생성하기 위한 조건을 식별하는 상기 특성들이 변경되는 경우 상기 출력 레코드는 변경되지 않는, 로그 정보를 생성하는 단계, 및

생성된 로그 정보를 출력하는 단계

를 포함하는, 방법.

청구항 2

제1항에 있어서,

상기 로그의 특성을 설정하는 것은 로그 정보가 생성되어야 하는 하나 이상의 이벤트를 선택하는 것을 포함하는, 방법.

청구항 3

제2항에 있어서,

상기 하나 이상의 이벤트는 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관되어 있는, 방법.

청구항 4

제3항에 있어서,

상기 하나 이상의 이벤트는 변환될 레코드 내의 에러를 식별하는 단계를 포함하는, 방법.

청구항 5

제3항에 있어서,

상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계를 포함하는, 방법.

청구항 6

제5항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계는 상기 주어진 레코드 내의 값들에 기초한 논리식을 충족시키는 단계를 포함하는, 방법.

청구항 7

제5항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계는 상기 레코드 내의 값들을 상기 규칙 사례와 연관된 값들과 비교하는 단계를 포함하는, 방법.

청구항 8

제1항에 있어서,

상기 로그 정보를 생성하는 단계는,

수신된 상기 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항들을 각각 포함하는 하나 이상의 범례(legend) 로그 메시지를 생성하는 단계, 및

다수의 트레이싱 로그 메시지를 생성하는 단계

를 포함하고,

각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술하는, 방법.

청구항 9

제8항에 있어서,

상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술하는, 방법.

청구항 10

제8항에 있어서,

상기 하나 이상의 범례 로그 메시지를 생성하는 단계는 그래프 기반 계산의 실행마다 하나의 범례 로그 메시지를 생성하는 단계를 포함하는, 방법.

청구항 11

제1항에 있어서,

생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 수신된 상기 규칙 명세와 연관된 제1 규칙에 따라 상기 입력 데이터 내의 입력 레코드를 변환하는, 방법.

청구항 12

제11항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보를 생성하는 단계는 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지를 생성하는 단계를 포함하는, 방법.

청구항 13

제11항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보를 생성하는 단계는 생성된 상기 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지를 생성하는 단계를 포함하는, 방법.

청구항 14

제11항에 있어서,

상기 생성된 로그 정보를 출력하는 단계는 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메

시지들을 출력하는 단계를 포함하는, 방법.

청구항 15

제14항에 있어서,

상기 생성된 로그 정보를 출력하는 단계는 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들을 수신하고, 상기 로그 메시지들로부터 적어도 부분적으로 얻은 로그 정보를 저장하는 단계를 더 포함하는, 방법.

청구항 16

제15항에 있어서,

상기 로그 메시지들을 필터링하고, 상기 로그 메시지들의 서브세트로부터 얻은 로그 정보를 저장하는 단계를 더 포함하는 방법.

청구항 17

제14항에 있어서,

상기 생성된 로그 정보를 출력하는 단계는,

명시된 규칙이 충족된 트리거된 규칙 사례를 나타내는 데이터 처리 구성요소의 로그 포트로부터 로그 메시지를 수신하는 단계,

상기 로그 메시지들을 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙들 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트를 결정하는 단계, 및

상기 입력 레코드의 축소된 세트를 저장하는 단계를 더 포함하는, 방법.

청구항 18

메모리; 및

프로세서로서,

입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하는 하나 이상의 규칙 명세(rule specification)의 수신;

상기 규칙 명세와 연관되어 있는 로그(log)의 특성들로서 로그 정보를 생성하기 위한 조건을 식별하는 특성들을 설정하기 위한 인터페이스의 제공;

수신된 상기 규칙 명세에 기초하여 변환식(transform)의 생성; 및

출력 레코드를 생성하기 위해 생성된 상기 변환식을 사용하여 입력 데이터의 변환

을 수행하도록 구성되는 프로세서

를 포함하고,

상기 입력 데이터의 변환은,

하나 이상의 조건이 발생했는지 여부를 결정,

상기 변환식을 실행시에 상기 변환식의 실행을 트레이싱(tracing),

상기 하나 이상의 조건이 발생했다는 결정에 응답하여, 트레이싱된 상기 실행에 기초하여 출력 레코드에 포함되지 않는 정보를 포함하는 로그 정보의 생성으로서, 로그 정보를 생성하기 위한 조건을 식별하는 상기 특성들이 변경되는 경우 상기 출력 레코드는 변경되지 않는, 로그 정보의 생성, 및

생성된 로그 정보의 출력을 포함하는,

컴퓨터 시스템.

청구항 19

제18항에 있어서,

상기 로그의 특성을 설정하는 것은 로그 정보가 생성되어야 하는 하나 이상의 이벤트를 선택하는 것을 포함하는, 컴퓨터 시스템.

청구항 20

제19항에 있어서,

상기 하나 이상의 이벤트는 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관되어 있는, 컴퓨터 시스템.

청구항 21

제20항에 있어서,

상기 하나 이상의 이벤트는 변환될 레코드 내의 에러의 식별을 포함하는, 컴퓨터 시스템.

청구항 22

제20항에 있어서,

상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건의 충족을 포함하는, 컴퓨터 시스템.

청구항 23

제22항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 주어진 레코드 내의 값들에 기초한 논리식의 충족을 포함하는, 컴퓨터 시스템.

청구항 24

제22항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 레코드 내의 값들과 상기 규칙 사례와 연관된 값들의 비교를 포함하는, 컴퓨터 시스템.

청구항 25

제18항에 있어서,

상기 로그 정보의 생성은,

수신된 상기 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항들을 각각 포함하는 하나 이상의 범례(legend) 로그 메시지의 생성, 및

다수의 트레이싱 로그 메시지의 생성

을 포함하고,

각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술하는, 컴퓨터 시스템.

청구항 26

제25항에 있어서,

상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술하는, 컴퓨터 시스템.

청구항 27

제25항에 있어서,

상기 하나 이상의 범례 로그 메시지의 생성은 그래프 기반 계산의 실행마다 하나의 범례 로그 메시지의 생성을 포함하는, 컴퓨터 시스템.

청구항 28

제18항에 있어서,

생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 수신된 상기 규칙 명세와 연관된 제1 규칙에 따라 상기 입력 데이터 내의 입력 레코드를 변환하는, 컴퓨터 시스템.

청구항 29

제28항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터 시스템.

청구항 30

제28항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 생성된 상기 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터 시스템.

청구항 31

제28항에 있어서,

상기 생성된 로그 정보의 출력은 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 출력을 포함하는, 컴퓨터 시스템.

청구항 32

제31항에 있어서,

상기 생성된 로그 정보의 출력은 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 수신, 및 상기 로그 메시지들로부터 적어도 부분적으로 얻은 로그 정보의 저장을 더 포함하는, 컴퓨터 시스템.

청구항 33

제32항에 있어서,

상기 프로세서는 상기 로그 메시지들의 필터링, 및 상기 로그 메시지들의 서브세트로부터 얻은 로그 정보의 저장을 수행하도록 더 구성되는, 컴퓨터 시스템.

청구항 34

제31항에 있어서,

상기 생성된 로그 정보의 출력은,

명시된 규칙이 충족된 트리거된 규칙 사례를 나타내는 데이터 처리 구성요소의 로그 포트로부터 로그 메시지의 수신,

상기 로그 메시지들을 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙들 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트의 결정, 및

상기 입력 레코드의 축소된 세트의 저장을 더 포함하는, 컴퓨터 시스템.

청구항 35

컴퓨터 프로그램을 저장하는 컴퓨터로 판독 가능한 매체로서,

상기 컴퓨터 프로그램은 컴퓨터로 하여금,

입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하는 하나 이상의 규칙 명세(rule specification)의 수신;

상기 규칙 명세와 연관되어 있는 로그(log)의 특성들로서 로그 정보를 생성하기 위한 조건을 식별하는 특성들을 설정하기 위한 인터페이스의 제공;

수신된 상기 규칙 명세에 기초하여 변환식(transform)의 생성; 및

출력 레코드를 생성하기 위해 생성된 상기 변환식을 사용하여 입력 데이터의 변환

을 하도록 하는 명령어들을 포함하며,

상기 입력 데이터의 변환은,

하나 이상의 조건이 발생했는지 여부를 결정,

상기 변환식을 실행시에 상기 변환식의 실행을 트레이싱(tracing),

상기 하나 이상의 조건이 발생했다는 결정에 응답하여, 트레이싱된 상기 실행에 기초하여 출력 레코드에 포함되지 않는 정보를 포함하는 로그 정보의 생성으로서, 로그 정보를 생성하기 위한 조건을 식별하는 상기 특성들이 변경되는 경우 상기 출력 레코드는 변경되지 않는, 로그 정보의 생성, 및

생성된 로그 정보의 출력을 포함하는,

컴퓨터로 판독 가능한 매체.

청구항 36

제35항에 있어서,

상기 로그의 특성을 설정하는 것은 로그 정보가 생성되어야 하는 하나 이상의 이벤트를 선택하는 것을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 37

제36항에 있어서,

상기 하나 이상의 이벤트는 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관되어 있는, 컴퓨터로 판독 가능한 매체.

청구항 38

제37항에 있어서,

상기 하나 이상의 이벤트는 변환될 레코드 내의 에러의 식별을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 39

제37항에 있어서,

상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건의 충족을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 40

제39항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 주어진 레코드 내의 값들에 기초한 논리식의 충족을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 41

제39항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 레코드 내의 값들과 상기 규칙 사례와 연관된 값들의 비교를 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 42

제35항에 있어서,

상기 로그 정보의 생성은,

수신된 상기 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항들을 각각 포함하는 하나 이상의 범례(legend) 로그 메시지의 생성, 및

다수의 트레이싱 로그 메시지의 생성

을 포함하고,

각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술하는, 컴퓨터로 판독 가능한 매체.

청구항 43

제42항에 있어서,

상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술하는, 컴퓨터로 판독 가능한 매체.

청구항 44

제42항에 있어서,

상기 하나 이상의 범례 로그 메시지의 생성은 그래프 기반 계산의 실행마다 하나의 범례 로그 메시지의 생성을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 45

제35항에 있어서,

생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 수신된 상기 규칙 명세와 연관된 제1 규칙에 따라 상기 입력 데이터 내의 입력 레코드를 변환하는, 컴퓨터로 판독 가능한 매체.

청구항 46

제45항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 47

제45항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 생성된 상기 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 48

제45항에 있어서,

상기 생성된 로그 정보의 출력은 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 출력을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 49

제48항에 있어서,

상기 생성된 로그 정보의 출력은 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 수신, 및 상기 로그 메시지들로부터 적어도 부분적으로 얻은 로그 정보의 저장을 더 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 50

제49항에 있어서,

상기 컴퓨터 프로그램은 컴퓨터로 하여금, 상기 로그 메시지들의 필터링, 및 상기 로그 메시지들의 서브세트로부터 얻은 로그 정보의 저장을 하도록 하는 명령어들을 더 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 51

제48항에 있어서,

상기 생성된 로그 정보의 출력은,

명시된 규칙이 충족된 트리거된 규칙 사례를 나타내는 데이터 처리 구성요소의 로그 포트로부터 로그 메시지의 수신,

상기 로그 메시지들을 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙들 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트의 결정, 및

상기 입력 레코드의 축소된 세트의 저장을 더 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 52

입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하는 하나 이상의 규칙 명세(rule specification)를 수신하는 단계;

상기 규칙 명세와 연관되어 있는 로그(log)의 특성들을 설정하기 위한 인터페이스를 제공하는 단계로서, 상기 특성들은 로그 정보를 생성하기 위한 조건을 식별하는, 인터페이스를 제공하는 단계;

수신된 상기 규칙 명세에 기초하여 변환식(transform)을 생성하는 단계; 및

생성된 상기 변환식을 사용하여 입력 데이터를 변환하는 단계

를 포함하고,

상기 입력 데이터를 변환하는 단계는,

하나 이상의 조건이 발생했는지 여부를 결정하는 단계,

상기 변환식을 실행시에 상기 변환식의 실행을 트레이싱(tracing)하는 단계,

상기 하나 이상의 조건이 발생했다는 결정에 응답하여, 트레이싱된 상기 실행에 기초하여 하나 이상의 범례(legend) 로그 메시지를 생성하는 단계로서, 각각의 범례 로그 메시지는 수신된 상기 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항들을 포함하는, 범례 로그 메시지를 생성하는 단계,

다수의 트레이싱 로그 메시지를 생성하는 단계로서, 각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술하는, 다수의 트레이싱 로그 메시지를 생성하는 단계, 및

생성된 범례 로그 메시지 및 트레이싱 로그 메시지를 출력하는 단계

를 포함하는, 방법.

청구항 53

제52항에 있어서,

상기 로그의 특성을 설정하는 것은 로그 정보가 생성되어야 하는 하나 이상의 이벤트를 선택하는 것을 포함하는, 방법.

청구항 54

제53항에 있어서,

상기 하나 이상의 이벤트는 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관되어 있는, 방법.

청구항 55

제54항에 있어서,

상기 하나 이상의 이벤트는 변환될 레코드 내의 에러를 식별하는 단계를 포함하는, 방법.

청구항 56

제54항에 있어서,

상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계를 포함하는, 방법.

청구항 57

제56항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계는 상기 주어진 레코드 내의 값들에 기초한 논리식을 충족시키는 단계를 포함하는, 방법.

청구항 58

제56항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계는 상기 레코드 내의 값들을 상기 규칙 사례와 연관된 값들과 비교하는 단계를 포함하는, 방법.

청구항 59

제52항에 있어서,

상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술하는, 방법.

청구항 60

제52항에 있어서,

상기 하나 이상의 범례 로그 메시지를 생성하는 단계는 그래프 기반 계산의 실행마다 하나의 범례 로그 메시지를 생성하는 단계를 포함하는, 방법.

청구항 61

제52항에 있어서,

생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 수신된 상기 규칙 명세와 연관된 제1 규칙에 따라 상기 입력 데이터 내의 입력 레코드를 변환하는, 방법.

청구항 62

제61항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보를 생성하는 단계는 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지를 생성하는 단계를 포함하는, 방법.

청구항 63

제61항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보를 생성하는 단계는 생성된 상기 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지를 생성하는 단계를 포함하는, 방법.

청구항 64

제61항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지를 출력하는 단계는 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들을 출력하는 단계를 포함하는, 방법.

청구항 65

제64항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지를 출력하는 단계는 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들을 수신하고, 상기 로그 메시지들로부터 적어도 부분적으로 얻은 로그 정보를 저장하는 단계를 더 포함하는, 방법.

청구항 66

제65항에 있어서,

상기 로그 메시지들을 필터링하고, 상기 로그 메시지들의 서브세트로부터 얻은 로그 정보를 저장하는 단계를 더 포함하는, 방법.

청구항 67

제64항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지를 출력하는 단계는,

명시된 규칙이 충족된 트리거된 규칙 사례를 나타내는 데이터 처리 구성요소의 로그 포트로부터 로그 메시지를 수신하는 단계,

상기 로그 메시지들을 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙들 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트를 결정하는 단계, 및

상기 입력 레코드의 축소된 세트를 저장하는 단계를 더 포함하는, 방법.

청구항 68

메모리; 및

프로세서로서,

입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하는 하나 이상의 규칙 명세(rule specification)의 수신;

상기 규칙 명세와 연관되어 있는 로그(log)의 특성들로서 로그 정보를 생성하기 위한 조건을 식별하는 특성들을 설정하기 위한 인터페이스의 제공;

수신된 상기 규칙 명세에 기초하여 변환식(transform)의 생성; 및

생성된 상기 변환식을 사용하여 입력 데이터의 변환

을 수행하도록 구성되는 프로세서

를 포함하고,

상기 입력 데이터의 변환은,

하나 이상의 조건이 발생했는지 여부를 결정,

상기 변환식을 실행시에 상기 변환식의 실행을 트레이싱(tracing),

상기 하나 이상의 조건이 발생했다는 결정에 응답하여, 트레이싱된 상기 실행에 기초하여, 수신된 상기 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항들을 각각 포함하는 하나 이상의 범례(legend) 로그 메시지의 생성

다수의 트레이싱 로그 메시지의 생성으로서, 각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술하는, 다수의 트레이싱 로그 메시지의 생성, 및

생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력을 포함하는,

컴퓨터 시스템.

청구항 69

제68항에 있어서,

상기 로그의 특성을 설정하는 것은 로그 정보가 생성되어야 하는 하나 이상의 이벤트를 선택하는 것을 포함하는, 컴퓨터 시스템.

청구항 70

제69항에 있어서,

상기 하나 이상의 이벤트는 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관되어 있는, 컴퓨터 시스템.

청구항 71

제70항에 있어서,

상기 하나 이상의 이벤트는 변환될 레코드 내의 에러의 식별을 포함하는, 컴퓨터 시스템.

청구항 72

제70항에 있어서,

상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건의 충족을 포함하는, 컴퓨터 시스템.

청구항 73

제72항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 주어진 레코드 내의 값들에 기초한 논리식의 충족을 포함하는, 컴퓨터 시스템.

청구항 74

제72항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 레코드 내의 값들과 상기 규칙 사례와 연관된 값들의 비교를 포함하는, 컴퓨터 시스템.

청구항 75

제68항에 있어서,

상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술하는, 컴퓨터 시스템.

청구항 76

제68항에 있어서,

상기 하나 이상의 범례 로그 메시지의 생성은 그래프 기반 계산의 실행마다 하나의 범례 로그 메시지의 생성을 포함하는, 컴퓨터 시스템.

청구항 77

제68항에 있어서,

생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 수신된 상기 규칙 명세와 연관된 제1 규칙에 따라 상기 입력 데이터 내의 입력 레코드를 변환하는, 컴퓨터 시스템.

청구항 78

제77항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터 시스템.

청구항 79

제77항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 생성된 상기 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터 시스템.

청구항 80

제77항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력은 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 출력을 포함하는, 컴퓨터 시스템.

청구항 81

제80항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력은 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 수신, 및 상기 로그 메시지들로부터 적어도 부분적으로 얻은 로그 정보의 저장을 더 포함하는, 컴퓨터 시스템.

청구항 82

제81항에 있어서,

상기 프로세서는 상기 로그 메시지들의 필터링, 및 상기 로그 메시지들의 서브세트로부터 얻은 로그 정보의 저장을 수행하도록 더 구성되는, 컴퓨터 시스템.

청구항 83

제80항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력은,

명시된 규칙이 충족된 트리거된 규칙 사례를 나타내는 데이터 처리 구성요소의 로그 포트로부터 로그 메시지의 수신,

상기 로그 메시지들을 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙들 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트의 결정, 및

상기 입력 레코드의 축소된 세트의 저장을 더 포함하는, 컴퓨터 시스템.

청구항 84

컴퓨터 프로그램을 저장하는 컴퓨터로 판독 가능한 매체로서,

상기 컴퓨터 프로그램은 컴퓨터로 하여금,

입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하는 하나 이상의 규칙 명세(rule specification)의 수신;

상기 규칙 명세와 연관되어 있는 로그(log)의 특성들로서 로그 정보를 생성하기 위한 조건을 식별하는 특성들을 설정하기 위한 인터페이스의 제공;

수신된 상기 규칙 명세에 기초하여 변환식(transform)의 생성; 및

생성된 상기 변환식을 사용하여 입력 데이터의 변환

을 하도록 하는 명령어들을 포함하며,

상기 입력 데이터의 변환은,

하나 이상의 조건이 발생했는지 여부를 결정,

상기 변환식을 실행시에 상기 변환식의 실행을 트레이싱(tracing),

상기 하나 이상의 조건이 발생했다는 결정에 응답하여, 트레이싱된 상기 실행에 기초하여, 수신된 상기 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항들을 각각 포함하는 하나 이상의 범례(legend) 로그 메시지의 생성

다수의 트레이싱 로그 메시지의 생성으로서, 각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술하는, 다수의 트레이싱 로그 메시지의 생성, 및

생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력을 포함하는,

컴퓨터로 판독 가능한 매체.

청구항 85

제84항에 있어서,

상기 로그의 특성을 설정하는 것은 로그 정보가 생성되어야 하는 하나 이상의 이벤트를 선택하는 것을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 86

제85항에 있어서,

상기 하나 이상의 이벤트는 수신된 상기 규칙 명세와 연관된 규칙에 따라 상기 입력 데이터 내의 레코드를 변환하는 것과 연관되어 있는, 컴퓨터로 판독 가능한 매체.

청구항 87

제86항에 있어서,

상기 하나 이상의 이벤트는 변환될 레코드 내의 에러의 식별을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 88

제86항에 있어서,

상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건의 충족을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 89

제88항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 주어진 레코드 내의 값들에 기초한 논리식의 충족을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 90

제88항에 있어서,

상기 주어진 레코드에 대한 규칙 사례의 조건의 충족은 상기 레코드 내의 값들과 상기 규칙 사례와 연관된 값들의 비교를 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 91

제84항에 있어서,

상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술하는, 컴퓨터로 판독 가능한 매체.

청구항 92

제84항에 있어서,

상기 하나 이상의 범례 로그 메시지의 생성은 그래프 기반 계산의 실행마다 하나의 범례 로그 메시지의 생성을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 93

제84항에 있어서,

생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 수신된 상기 규칙 명세와 연관된 제1 규칙에 따라 상기 입력 데이터 내의 입력 레코드를 변환하는, 컴퓨터로 판독 가능한 매체.

청구항 94

제93항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 95

제93항에 있어서,

트레이싱된 상기 실행에 기초하여 로그 정보의 생성은 생성된 상기 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지의 생성을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 96

제93항에 있어서,

상기 생성된 로그 정보의 출력은 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 출력을 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 97

제96항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력은 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지들의 수신, 및 상기 로그 메시지들로부터 적어도 부분적으로 얻은 로그 정보의 저장을 더 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 98

제97항에 있어서,

상기 컴퓨터 프로그램은 컴퓨터로 하여금, 상기 로그 메시지들의 필터링, 및 상기 로그 메시지들의 서브세트로

부터 얻은 로그 정보의 저장을 하도록 하는 명령어들을 더 포함하는, 컴퓨터로 판독 가능한 매체.

청구항 99

제96항에 있어서,

상기 생성된 범례 로그 메시지 및 트레이싱 로그 메시지의 출력은,

명시된 규칙이 충족된 트리거된 규칙 사례를 나타내는 데이터 처리 구성요소의 로그 포트로부터 로그 메시지의 수신,

상기 로그 메시지들을 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙들 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트의 결정, 및

상기 입력 레코드의 축소된 세트의 저장을 더 포함하는, 컴퓨터로 판독 가능한 매체.

명 세 서

기 술 분 야

[0001] 관련출원의 상호 참조

[0002] 본 출원은 2008년 6월 20일에 출원된 미국 특허출원 제61/077,022호에 대해 우선권을 주장하며, 이는 인용에 의해 본 출원에 포함된다.

[0003] 본 발명은 일반적으로 그래프 기반 계산(graph-based computation)에서의 데이터 로깅(data logging)에 관한 것이다.

배 경 기 술

[0004] 복잡한 계산은 보통 그래프의 꼭지점(vertex)들과 연관되는 계산의 구성요소(componet)들을 가지는, 유향 그래프(directed graph)에 의해 데이터 흐름(data flow)으로, 그리고 그래프의 링크[유향변(arc), 변(edge)]들에 대응하는 구성요소들 사이의 데이터 흐름들로서 표현될 수 있다. 명칭 "EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS"의 미국특허 제5,966,072호에는 그러한 그래프 기반 계산을 구현하는 시스템이 기술되어 있다. 어떤 경우에는, 꼭지점과 연관된 계산을 "비지니스 규칙(business rule)"이라고 하는 사람이 판독 가능한 형태로 기술한다.

발명의 내용

해결하려는 과제

[0005] 그래프 기반 계산(graph-based computation)에서의 데이터 로깅(data logging)을 제공한다.

과제의 해결 수단

[0006] 일 측면에서, 일반적으로, 방법은 데이터 흐름을 나타내는 링킹 요소(linking element)에 의해 연결된 데이터 처리 구성요소를 가지는 그래프 기반 계산에 대한 하나 이상의 규칙 명세(rule specification)를 수신하는 단계; 상기 수신된 규칙 명세에 기초하여 상기 그래프 기반 계산 내의 하나 이상의 데이터 처리 구성요소에 대한 변환식(transform)을 생성하는 단계; 및 상기 생성된 변환식을 사용하여 하나 이상의 데이터 흐름을 변환하는 단계를 포함하고; 상기 규칙 명세는 입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례(rule case)와 각각 연관되어 있는 규칙들을 정의하며; 상기 변환식을 생성하는 단계는, 상기 생성된 변환식과 연관된 로그(log)의 특성을 설정하기 위한 인터페이스를 제공하는 단계를 포함하고; 상기 데이터 흐름을 변환하는 단계는, 실행시에 상기 그래프 기반 계산 내의 상기 데이터 처리 구성요소에 대한 실행을 트레이싱(tracing)하는 단계, 상기 설정된 로그 특성에 따라 상기 트레이싱된 실행에 기초하여 로그 정보를 생성하는 단계, 및 상기 생성된 로그 정보를 저장 또는 출력하는 단계를 포함한다.

[0007] 여러 측면들은 다음 특징 중 하나 이상을 포함할 수 있다.

[0008] 상기 로그의 특성을 설정하는 단계는 로그 정보가 생성되어야 하는 하나 이상의 이벤트(event)를 선택하는 단계를 포함한다.

[0009] 상기 하나 이상의 이벤트는 상기 수신된 규칙 명세와 연관된 규칙에 따라 상기 데이터 흐름 내의 레코드를 변환하는 것과 연관되어 있다.

[0010] 상기 하나 이상의 이벤트는 변환될 레코드 내의 에러를 식별하는 단계를 포함한다.

[0011] 상기 하나 이상의 이벤트는 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계를 포함한다.

[0012] 상기 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계는 상기 주어진 레코드 내의 값들에 기초한 논리식(logical expression)을 충족시키는 단계를 포함한다.

[0013] 상기 주어진 레코드에 대한 규칙 사례의 조건을 충족시키는 단계는 상기 레코드 내의 값들을 상기 규칙 사례와 연관된 값들과 비교하는 단계를 포함한다.

[0014] 상기 로그 정보를 생성하는 단계는, 상기 수신된 규칙 명세에 의해 정의된 규칙들을 포함하는 규칙 세트의 세부사항(detail)들을 각각 포함하는 하나 이상의 범례(legend) 로그 메시지를 생성하는 단계, 및 다수의 트레이싱 로그 메시지를 생성하는 단계를 포함하고, 각 트레이싱 로그 메시지는 범례 레코드와 연관되어 있고, 상기 수신된 규칙 명세와 연관된 규칙에 따라 상기 데이터 흐름 내의 레코드를 변환하는 것과 연관된 하나 이상의 이벤트를 기술한다.

[0015] 상기 이벤트를 기술하는 트레이싱 로그 메시지는 상기 범례 레코드 내의 인덱스를 사용하여 데이터 처리 구성요소의 하나 이상의 입력 또는 출력을 기술한다.

[0016] 상기 하나 이상의 범례 메시지를 생성하는 단계는 그래프 기반 계산의 실행마다 하나의 범례 메시지를 생성하는 단계를 포함한다.

[0017] 상기 생성된 변환식을 사용하는 상기 데이터 처리 구성요소는 상기 수신된 규칙 명세와 연관된 제1 규칙에 따라 상기 데이터 흐름 내의 입력 레코드를 변환한다.

[0018] 상기 트레이싱된 실행에 기초하여 로그 정보를 생성하는 단계는 명시된 기준이 충족되는 상기 제1 규칙의 규칙 사례 각각에 대한 로그 메시지를 생성하는 단계를 포함한다.

[0019] 상기 트레이싱된 실행에 기초하여 로그 정보를 생성하는 단계는 상기 생성된 변환식을 사용하여 상기 데이터 처리 구성요소에 의해 생성된, 변환된 레코드 내의 값 각각에 대한 로그 메시지를 생성하는 단계를 포함한다.

[0020] 상기 생성된 로그 정보를 저장 또는 출력하는 단계는 하나 이상의 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지를 출력하는 단계를 포함한다.

[0021] 상기 생성된 로그 정보를 저장 또는 출력하는 단계는 또한 상기 데이터 처리 구성요소의 로그 포트로부터 로그 메시지의 데이터 흐름을 수신하고, 적어도 일부를 상기 로그 메시지로부터 얻은 로그 정보를 저장하는 단계를 더 포함한다.

[0022] 상기 방법은 또한 상기 로그 메시지의 수신된 데이터 흐름을 필터링하고, 상기 로그 메시지의 서브세트로부터 얻은 로그 정보를 저장하는 단계를 포함한다.

[0023] 상기 생성된 로그 정보를 저장 또는 출력하는 단계는 또한, 명시된 규칙이 충족되어 트리거된(triggered) 규칙 사례를 나타내는 데이터 처리 구성요소 내의 로그 포트로부터 로그 메시지의 데이터 흐름을 수신하는 단계, 상기 로그 메시지를 검사하여, 모든 입력 레코드에 의해 트리거되는 상기 규칙 명세에 의해 정의된 규칙 각각의, 규칙 사례 각각에 대한 하나 이상의 로그 메시지를 제공하는 입력 레코드의 축소된 세트를 결정하는 단계, 및 상기 입력 레코드의 축소된 세트를 저장하는 단계를 더 포함한다.

[0024] 다른 측면에서, 일반적으로, 컴퓨터 시스템은, 데이터 흐름을 나타내는 링킹 요소에 의해 연결된 데이터 처리 구성요소를 가지는 그래프 기반 계산에 대한 하나 이상의 규칙 명세를 저장하고, 상기 규칙 명세는 입력 데이터에 의존하는 하나 이상의 출력값을 결정하기 위한 기준을 명시하는 하나 이상의 규칙 사례와 각각 연관된 규칙을 정의하는, 스토리지 시스템; 생성되는 변환식과 연관된 로그의 특성을 설정하기 위한 인터페이스의 제공을 포함하여, 상기 수신된 규칙 명세에 기초하여 상기 그래프 기반 계산 내의 하나 이상의 데이터 처리 구성요소에 대한 변환식을 생성하도록 구성된 생성기; 및 실행시에 상기 그래프 기반 계산 내의 상기 데이터 처리 구성요소에 대한 실행의 트레이싱, 상기 설정된 로그 특성에 따라 상기 트레이싱된 실행에 기초하여 로그 정보의 생성, 및 상기 생성된 로그 정보의 저장 또는 출력을 포함하여, 상기 생성된 변환식을 사용하여 하나 이상의 데이터 흐름을 변환하도록 구성된 그래프 기반 계산 시스템을 포함한다.

[0025] 다른 측면에서, 일반적으로, 컴퓨터 프로그램은 컴퓨터로 판독 가능한 매체에 저장되며, 상기 컴퓨터 프로그램은 상기 컴퓨터로 하여금,

[0026] 데이터 흐름을 나타내는 링킹 요소에 의해 연결된 데이터 처리 구성요소를 가지는 그래프 기반 계산에 대한 하나 이상의 규칙 명세를 수신하도록 하고, 상기 규칙 명세는 입력 데이터에 의존하는 하나 이상의 출력값을 결정하기 위한 기준을 명시하는 하나 이상의 규칙 사례와 각각 연관된 규칙을 정의하며; 생성되는 변환식과 연관된 로그의 특성을 설정하기 위한 인터페이스의 제공을 포함하여, 상기 수신된 규칙 명세에 기초하여 상기 그래프 기반 계산 내의 하나 이상의 데이터 처리 구성요소에 대한 변환식을 생성하도록 하고; 실행시에 상기 그래프 기반 계산 내의 상기 데이터 처리 구성요소에 대한 실행의 트레이싱, 상기 설정된 로그 특성에 따라 상기 트레이싱된 실행에 기초하여 로그 정보의 생성, 및 상기 생성된 로그 정보를 저장 또는 출력을 포함하여, 상기 생성된 변환식을 사용하여 하나 이상의 데이터 흐름을 변환하도록 하는, 명령어들을 포함한다.

[0027] 다른 측면에서, 일반적으로, 컴퓨터 시스템은, 데이터 흐름을 나타내는 링킹 요소에 의해 연결된 데이터 처리 구성요소를 가지는 그래프 기반 계산에 대한 하나 이상의 규칙 명세를 수신하고, 상기 규칙 명세는 입력 데이터에 의존하는 하나 이상의 출력 값을 결정하기 위한 기준들을 명시하는 하나 이상의 규칙 사례와 각각 연관된 규칙들을 정의하는, 수단; 생성되는 변환식과 연관된 로그의 특성을 설정하기 위한 인터페이스의 제공을 포함하여, 상기 수신된 규칙 명세에 기초하여 상기 그래프 기반 계산 내의 하나 이상의 데이터 처리 구성요소에 대한 변환식을 생성하는 수단; 및 실행시에 상기 그래프 기반 계산 내의 상기 데이터 처리 구성요소에 대한 실행의 트레이싱, 상기 설정된 로그 특성에 따라 상기 트레이싱된 실행에 기초하여 로그 정보의 생성, 및 상기 생성된 로그 정보의 저장 또는 출력을 포함하여, 상기 생성된 변환식을 사용하여 하나 이상의 데이터 흐름을 변환하는 수단을 포함한다.

[0028] 본 발명의 하나 이상의 실시예에 대한 세부사항은 첨부도면 및 이하의 구체적인 내용에 설명한다. 본 발명의 다른 특징들, 목적들, 및 이점들은 설명과 도면, 및 청구항들로부터 명백해질 것이다.

발명의 효과

[0029] 그래프 기반 계산(graph-based computation)에서의 데이터 로깅(data logging)이 개선된다.

도면의 간단한 설명

[0030] 도 1a는 그래프 기반 계산 환경에서 입력 데이터를 출력 데이터로 변환하는 것을 나타낸다.

도 1b는 그래프 기반 계산에서의 그래프의 일 예를 나타낸다.

도 1c는 로깅 정보를 가지는 변환식의 생성을 나타낸 블록도이다.

도 2a는 스프레드시트 기반 규칙 엔트리의 예를 나타낸다.

도 2b는 개별 규칙의 일 예를 나타낸다.

도 3은 그래프 기반 계산 환경에서의 데이터 로깅의 몇몇 동작에 대한 흐름도이다.

도 4는 그래프 기반 계산 환경에서의 로깅 설정의 그래픽 사용자 인터페이스를 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0031] 데이터 로깅 및 감사(auditing) 메커니즘의 예시적인 맥락은 그래프 기반 계산 시스템에서 그래프 기반 계산에 연관된 메타데이터(metadata)를 저장하는 그래프 기반 계산 패러다임(paradigm)을 포함한다. 이와 관련하여, 각 컴퓨터 프로그램은 소위 데이터 흐름 그래프 또는 간단히 그래프라고도 하는 계산 그래프를 사용하여 구현된다. 그래프는 구성요소들 사이에 데이터의 흐름을 나타내는 유향 변(directed edge)으로 연결된 데이터 처리 구성요소를 나타내는 하나 이상의 노드 또는 꼭지점을 포함한다. 병렬 처리 환경에서는 복수의 그래프가 실행(run)될 수 있다. 시스템은 그래프의 발전에 있어 변화를 추적하고, 통계적 및 종속성 분석(statistical and dependency analysis)을 수행하며, 그래프의 발전과 관련되는 메타데이터(metadata)를 관리한다. 그래프 관련 메타데이터의 저장은, 데이터가 그래프에서 처리될 때 데이터가 어떻게 변하고 있는지, 그리고 그러한 변화가 다른 그래프에 미치는 영향에 대한 시각 자료(visual sense)를 사용자에게 제공하기 위한 데이터 영향 분석을 가능하게 한다. 또, 시스템은 코드 변화가 있을 때 저장될 그래프의 복수의 버전을 허용하는 설정/변경 관리를 제공하여, 최근 코드와 데이터가 사용 가능하도록 보장한다.

[0032] 메타데이터의 서브세트인 비지니스 규칙은 시스템 내에 저장된다. 비지니스 규칙의 여러 측면은, 예를 들면 인용에 의해 본 명세서에 포함되는 미국특허출원 제11/733,434호에 기술되어 있다. 각 비지니스 규칙은 개별 객체(object)로 저장될 수 있다. 비지니스 규칙은 데이터를 하나의 포맷에서 다른 포맷으로 변환, 데이터에 관한 결정, 또는 입력 데이터의 세트에 기초한 새로운 데이터의 생성을 위한 기준의 세트로서 표현될 수 있다. 예를 들면, 도 1a에서, 항공편 예약 시스템에서의 레코드(102)는 승객의 성명(104), 올해 비행한 마일수(106), 티켓의 등급(108), 및 좌석의 열을 나타내는 필드들에 대한 값을 포함한다. 비지니스 규칙은 그러한 승객을 탑승 그룹 1에 넣어야 한다는 것을 나타낸다. 비지니스 규칙은 일반적으로 사람이 이해하기 용이하다, 즉 "1등석 승객은 그룹 1에 속한다"라는 것이지만, 데이터 처리를 위해 사용될 수 있기 전에 컴퓨터가 이해할 수 있는 어떤 것으로 번역될 필요가 있다.

[0033] 그래프 기반 계산 환경에서 비지니스 규칙을 구현하기 위해서는, 하나 이상의 데이터 소스, 예컨대 입력 데이터세트(100)로부터 레코드(102)와 같은 입력 레코드를 수신하여, 출력 데이터세트(120)에 승객의 성명(104)과 그가 속하는 그룹(118)을 나타내는 출력 레코드, 예컨대 레코드(114)를 제공하는, 변환식(112)을 생성한다. 이 예에서, 데이터세트는 하나의 예시적인 레코드를 나타내는 것으로 예시되어 있지만, 일반적으로 데이터세트는 임의의 개수의 레코드를 포함할 수 있다. 입력 및 출력 데이터세트는 데이터 스트림으로서, 예컨대 그래프 내로 또는 밖으로의 데이터세트 흐름을 구성하는 데이터로서 처리될 수 있다.

[0034] 변환식은 데이터 흐름을 나타내는 링킹 요소에 의해 연결된 데이터 처리 구성요소를 가지는 그래프 기반 계산에서 구현될 수 있다. 예를 들면, 도 1b의 단순 계산 그래프(130)는 입력으로 두 개의 데이터세트(132, 134)(예를 들면, 상용 고객(frequent flier) 데이터 및 항공편 예약 데이터), 각 세트 내의 데이터를 별개의 포맷 구성요소(136, 138)에서 포맷하고, 이들을 결합 구성요소(140)에서 결합하여 출력 데이터세트(142)를 생성한다. 변환식은 그래프(130)에서와 같이 자체가 그래프 기반 계산일 수 있고, 또는 그래프(130)를 구성하는 개별 구성요소(136, 138, 140)과 같은 그래프의 구성요소 안에 구현될 수 있다.

[0035] 숙달되지 않은(non-technical) 사용자를 위해 변환식의 생성을 단순화하려면, 그러한 사용자에게 그들이 변환식에게 수행하기를 원하는 것을 컴퓨터 시스템에게 알려주는, 규칙 세트라고 불리는 비지니스 규칙의 세트를 그들에게 익숙한 포맷으로 입력하기 위한 도구가 제공된다. 규칙 세트는 단일의 변환식을 생성하는 규칙들로 이루어진 세트이다. 규칙은 입력에 의존하는 규칙의 출력에 대해 상이한 값을 결정하는 하나 이상의 규칙 사례로 구성될 수 있다. 규칙은 또한 다른 규칙들을 포함할 수 있다. 규칙 세트 내의 다른 규칙들은 추가 또는 다른 출력에 대한 값을 생성할 수 있다. 규칙 세트는 "포함되는(included)" 규칙 세트이라고 불리는 다른 규칙 세트들을 포함할 수 있다.

[0036] 도 1c에는 로깅 정보를 가지는 변환식 생성 시스템의 일반적인 모델이 나타나 있다. 비지니스 규칙 환경(business rules environment, BRE)은 편집기(154)로부터 규칙 세트(152)을 입력으로서 수신하고 변환식(156)을 생성하는 생성기(150)를 포함한다. 변환식 생성 옵션 중 하나로서, 로깅은 그래픽 사용자 인터페이스 내의 정보 및 각종 로깅 이벤트의 커스터마이징에 의해 나중에 활성화될 수 있다. 로그는 조직의 시스템 및 네트워크 내에서 발생하는 이벤트들의 레코드이다. 로그들은 엔트리들로 구성되고, 각 엔트리는 시스템 또는 네트워크 내에서 발생된 특정한 이벤트와 관련된 정보를 포함한다. 로그는 문제 해결(troubleshooting problem)을 위해, 또한 시스템 및 네트워크 성능을 최적화, 사용자의 행위(action)를 기록, 및 비정상적인 활동의 조사에 유용한 데이터 제공과 같은, 많은 기능을 제공하기 위해 사용될 수 있다. 로그는 많은 상이한 타입의 이벤트와 관련된 정보를 포함할 수 있다. 생성된 변환식(156)은 시스템의 아키텍처 및 변환식과 비지니스 규칙의 목적에 따라 그래프에서 사용될 구성요소로서 또는 전체 그래프 자체로서 그래프 기반 계산 시스템(158)에 제공될 수 있다. 생성기(150)는 예를 들면 규칙 세트(152)를 수신하여 변환식(156)을 출력하는, 컴파일러, 주문 제작된(custom-built) 프로그램, 또는 표준 도구를 사용하여 구성된 다른 그래프 기반 계산일 수 있다.

[0037] 생성기(150)는 또한 규칙 세트(152)가 편집되는 경우 변환식(156)을 갱신할 수 있다. 규칙 세트(152)가 편집되는 경우, 편집기(154)는 전체 규칙 세트를 편집자에게 제공할 수 있고, 또는 새롭거나 수정된 규칙 또는 규칙 사례(152a)만을 제공할 수 있다. 생성기(150)는 원래의 변환식(156)을 대체하는 완전히 새로운 변환식을 생성할 수 있고, 또는 변환식을 사용하는 시스템의 능력 및 필요에 따라 변환식을 포함하는 구성요소(156a)을 제공할 수 있다.

[0038] 그래프 기반 계산(15) 중에 로깅 동작을 수행하기 위한 개별, 전용의 실행 엔진은 필요하지 않다. 로깅은 그래프 구성요소가 실행될 때 그래프 구성요소에 의해 호출된 함수의 사용을 발생시키도록 구성될 수 있다. 상이한 로깅 설정(logging configuration)에 대해서는, 규칙 실행에 대한 상이한 감사 보고서가 제공될 수 있다. 예를

들면, 도 1c에서 점선의 화살표(실선의 화살표로 표시된 실제 데이터 흐름이 아님)로 도시된 바와 같이, 로그(160a)는 규칙 세트(152) 내의 특정한 입력 레코드로 돌아가 트레이싱할 수 있으며; 로그(160b)는 이전 시각에 활성화된(fired) 특정한 규칙 사례(152a)를 반영할 수 있다.

[0039] 도 2a를 참조하면, 어떤 예에서는, 스프레드시트 포맷으로 규칙을 입력할 수 있다. 스프레드시트(200)의 트리거 칼럼(trigger column)(202, 204, 206, 208)은 이용 가능한 데이터 값에 해당하고, 로우(210a-h)는 규칙 사례, 즉 이용 가능한 데이터 값에 관련된 기준 세트에 해당한다. 기준을 가지는 트리거 칼럼 각각에 대해, 해당 레코드의 데이터 값이 트리거 기준을 충족하는 경우, 규칙 사례(210n)가 주어진 레코드(예컨대, 도 1a의 102)에 적용된다. 규칙 사례(210n)가 적용되는 경우, 하나 이상의 출력 칼럼(212)에 기초하여 출력이 생성된다. 모든 트리거 기준이 충족된 규칙 사례는 "트리거된 것(triged)"이라고 지칭될 수 있다. 각 출력 칼럼(212)은 잠재적인 출력 변수에 해당하고, 적용 가능한 로우(210n)의 대응하는 셀(cell)의 값이, 있다면, 그 변수에 대한 출력을 결정한다. 셀은 변수에 할당된 값을 포함할 수 있다. 또는 셀은 이하에 설명하는 바와 같이 출력값을 생성하기 위해 평가되어야 하는 표현식(expression)을 포함할 수 있다. 도 2a에는 출력 칼럼이 한 개만 도시되어 있지만, 출력 칼럼은 하나 이상 있을 수 있다.

[0040] 변수에 해당하는 칼럼, 한번 계산될 뿐이고 나중에 변수처럼 취급되는 표현식을 포함하는 칼럼, 및 표현식만을 포함하는 칼럼을 비롯한, 몇몇 상이한 타입의 트리거 칼럼이 존재할 수 있다. 다른 칼럼 타입들에는 데이터만을 포함하는 칼럼들 및 이 데이터만을 포함하는 칼럼들에 기초하여 로우마다 평가하기 위한 표현식을 명시하는 칼럼들을 포함한다. 표현식만을 포함하는 칼럼들은 변수에 해당되거나 변수로 취급되는 칼럼들보다 단순하다.

[0041] 도 2a의 예에서, 첫 번째 로우(210a)는 단 하나의 칼럼(202)에서 기준을 가지며, 이는 여행객의 마일리지(frequent flier miles)의 총계가 1,000,000보다 많은 경우이면, 규칙 사례는 기타 칼럼들의 값이 무엇이든 관계 없이 적용된다는 것을 나타낸다. 그 경우에, 해당 사용자에 대한 "탑승 그룹" 출력 변수는 그룹 1로 설정된다. 마찬가지로, 두 번째 규칙 사례(210b)는 일등석의 임의의 승객이 그룹 1에 속한다는 것을 나타낸다. 어떤 예에서, 규칙들은 차례로 평가되어, 1,000,000이 넘는 마일리지 가지는 여행객과 일등석 티켓을 가지는 여행객은 그룹 1에 속하게 될 것이지만, 첫 번째 규칙 사례(210a)만 트리거될 것이다. 일단 규칙 사례가 트리거되면, 그 규칙 내의 다른 규칙 사례들을 평가할 필요는 없다.

[0042] 다음의 규칙 사례(210c)는 두 개의 입력값(202, 204)에 기초하며, 총 마일리지 및 금년 마일리지 모두에 대해 정의된 기준을 충족하는 경우, 해당 승객은 그룹 2에 속한다. 네 번째 규칙 사례(210d)에서, 또한 모든 비지니스석 고객도 그룹 2에 속한다. 나머지 규칙 사례(210e-h)는 다른 규칙 사례, 즉 "그 외(else)" 및 "동일(same)"과 관련된 기준을 포함한다. "그 외"는 그보다 위에 있는 로우에서 해당 칼럼 내의 어떠한 기준도 충족되지 않았음을 나타내며 좌측과 동일한 기준(규칙 210b 및 210c)을 가지고, 한편 "동일"은 그보다 위의 규칙 사례가 해당 칼럼에 대해 적용되었다면 그 규칙 사례가 적용되는 것을 나타낸다. 따라서, 다섯 번째 규칙 사례(210e)는 처음의 두 칼럼(202 또는 204)의 어떠한 기준에도 부합하지 않고(규칙 사례(210a 또는 210c)에 의해 처리되었을 것이기 때문임), "좌석 등급(class of seat)" 칼럼이 "일등석" 또는 "비지니스석"이 아니고(칼럼 206에서 "그 외" 키워드), "좌석열" 값(208)이 10 이하인 모든 레코드에 적용된다. 나머지 규칙 사례(210f-h)는 또한 칼럼 202 또는 204 내의 어떤 상위(higher) 규칙 사례에도 부합하지 않고, "좌석 등급" 칼럼이 "일등석" 또는 "비지니스석"이 아니며, 적절한 "좌석열" 값을 가지는 레코드에 적용된다.

[0043] 도 2a의 예에서 규칙 사례(210f-h)는 또한 도 2b에 도시된 바와 같이, 각자의 스프레드시트 내에서 개개의 샘플 규칙으로 나타낼 수 있다. 규칙(220a-d)은 도 2a의 로우(210a-d)에 각각 대응하고, 규칙(220e)는 로우(210e-h) 모두에 대응하는 네 개의 규칙 사례를 가진다. 사용자는 도 2a에 나타낸 표 전체를 생성하기 보다, 이들 개개의 규칙을 각각 생성할 수 있다. 각 규칙 사례는 모든 트리거 칼럼에 대한 값 및 모든 출력 칼럼에 대한 값을 포한다(이 값은 공백, 즉 사실상 "아무것(any)"으로 설정될 수 있음). 복수의 규칙이 동일한 출력을 생성하는 경우, 그 규칙들은 순서가 매겨지고, 규칙들은 하나의 규칙 내의 규칙 사례가 입력에 대해 트리거가 되어 출력을 생성할 때까지 순서대로 고려된다. 규칙 내에 트리거가 되는 규칙 사례가 없는 경우, 동일한 출력을 생성하는 다음 규칙이 처리된다. 규칙 내의 어떠한 사례도 출력의 트리거가 되지 않는 경우, 디폴트 값이 사용된다.

[0044] 어떤 예에서, 편집기 인터페이스(150)는 표현식을 포함하는 셀을 그래픽적으로 식별할 수 있다. 이는 사용자가 저절로 참 또는 거짓이 평가될 표현식과, 칼럼의 변수와 비교된 값을 반환(return)하는 표현식 사이의 차를 이해하도록 도울 것이다. 사용자가 타이핑하고 있을 때, 사용자는 예를 들면 처음에 별표(asterisk, *)를 타이핑하여 특정한 셀이 표현식 셀이 될 것임을 나타낼 수 있다.

[0045] 출력 변수에 대응하는 칼럼에 대해, 셀은 다음 중 하나를 포함할 수 있다:

[0046] ● 값. 이 값이 출력 변수에 할당될 것이다.

[0047] ● 표현식. 이 표현식의 값이 출력 변수에 할당된다. 이 표현식이 NULL(널)로 평가하면 필드는 NULL 값을 얻고, 아니면 출력 필드는 널을 허용하지 않는다(not-nullable). 이 경우, 에러가 생성된다.

[0048] ● 키워드 "null(널)". 출력 필드가 널을 허용하면, 필드는 NULL을 할당받을 것이다. 그렇지 않으면, 에러가 생성된다.

[0049] ● 빈 스트링(empty string). 출력 필드가 디폴트 값을 가지면, 그 디폴트 값이 할당된다. 그렇지 않으면, 셀은 키워드 "null(널)"을 포함하는 것처럼 취급된다.

[0050] ● 키워드 "same(동일)". 출력 필드가 그 위의 셀에서 계산된 동일한 값을 할당받는다.

[0051] 가능한 경우, 에러는 검출된 때, 즉 널을 허용하지 않는 필드의 출력 칼럼에 "null(널)"을 넣은 때 보고될 수 있다. 그러나, 어떤 에러는 테스트시 또는 실행시까지 보고될 수 없다.

[0052] 테이블의 로우로서 생성되든 개개의 규칙으로서 생성되든, 각 규칙은 속성(attribute)들로 이루어진 어떤 특정한 세트를 가진다. 규칙 세트는 포함하는 규칙에 대해 이 속성들을 결정할 수 있다. 이 속성에는 명칭, 규칙 타입, 설명 및 주석 필드, 출력 변수의 리스트, 입력 변수의 리스트, 인수(argument)의 리스트, 트리거 칼럼의 리스트, 변경 이력, 테스트 데이트세트, 및 에러 처리 거동(error handling behavior)을 포함할 수 있다. 명칭은 자명한 것(self-explanatory)으로, 규칙 세트 내의 규칙을 열거하기 위해 사용된다. 어떤 예에서, 규칙 타입은 규칙 세트의 속성이다. 출력 변수의 리스트는 규칙에 의해 생성된 변수 또는 할당된 값의 세트이다. 출력 변수의 리스트는 규칙 세트로부터 상속될 수 있고, 하나 이상의 출력이 있을 수 있다. 입력 변수의 리스트는, 규칙이 칼럼의 맨 위의 레코드 및 표현식 내부에 사용된 레코드(예컨대, 도 2a의 규칙(210c)에 사용된 "작년 마일리지(last year frequent flyer miles)" 값이 표현식에 사용되지만 그 자신의 칼럼을 가지지 않는다)를 포함하는 레코드를 평가할 필요가 있음을 나타내는 모든 변수를 식별할 수 있게 해준다.

[0053] 규칙들은 단발식(single-fired) 또는 다발식(multi-fired)일 수 있다. 예를 들면, 복수의 규칙 사례가 하나 이상의 출력에 대한 복수의 값을 생성하는 데 사용될 수 있다. 다수의 규칙 사례를 트리거할 수 있는 규칙을 다발 규칙(multi-fire rule)이라고 한다. 다발 규칙은 오로지 그 규칙에 의해 계산된 출력의 타입에 기초하여서만 식별된다. 규칙에 의해 계산된 출력(들)이 리스트(각 레코드 내의 복수의 값을 가질 수 있는 출력)이면, 그 규칙은 다발 규칙이다. 다발 규칙에서, 일단 규칙 사례가 트리거되면, 대응하는 출력 값은 출력에 대한 값의 리스트에 첨부된다. 그러나, 단발 규칙(single-fire rule)과 달리, 다발 규칙에 있어서는 규칙 사례가 트리거된 후라도 평가는 계속된다. 각각의 그 다음 규칙 사례도 평가되며, 트리거시키는 모든 규칙 사례는 다른 값이 출력(들)에 대한 값의 리스트에 첨부되게 할 것이다.

[0054] 어떤 예에서, 규칙들은, 논리곱 연산되는(ANDed) 로우들 및 논리합 연산되는(ORed) 칼럼들 내의 규칙 사례들을 사용하여, 전술한 것에 대해 대화 방식으로 평가될 수 있다. 즉, 규칙은 모든 로우가 트리거(로우들을 논리곱함)되지만 단 하나의 셀만 참이어야 하는 경우에만, 셀은 각각의 로우를 트리거하기 위한 출력을 생성한다.

[0055] 인자의 리스트는 함수 규칙에만 존재한다. 인자의 리스트는 규칙에 대한 입력인 파리미터의 명칭 및 타입을 식별할 수 있게 해주고, 규칙 세트의 특성일 수 있다. 트리거 칼럼의 리스트는 어떤 칼럼들이 규칙의 응용예를 트리거시킬 수 있는지를 식별할 수 있게 해준다. 도 2a 및 도 2b의 예에 도시된 입력 변수에 국한되지 않고, 트리거 칼럼들은 파리미터, 룩업 변수, 이전 규칙으로부터의 출력 변수, 포함된 규칙 세트들의 출력 변수, 그 규칙 세트의 파라미터, 또는 표현식에 해당할 수 있다. 트리거 칼럼은 또한 함수 규칙으로부터의 입력 변수, 즉 인자를 포함할 수도 있다.

[0056] 에러 처리 방식(error handling)은 규칙 세트로부터 생성된 변환식이 규칙을 평가할 때 발생하는 에러를 취급하는 방법을 결정한다. 트리거 표현식에서의 에러 처리를 위해, 옵션들은 에러를 허용할 수 있고, 이 경우 변환식은 에러가 발생된 레코드를 거부하거나, 에러를 무시하는데, 이는 트리거 표현식을 거짓인 것으로 가정하고 다음의 규칙으로 이동하는 것과 같다. 출력 표현식의 경우, 에러를 허용하고 그 레코드를 거부함으로써, 에러를 무시하고 그 출력을 NULL로 설정함으로써, 또는 규칙 내의 로우를 무시하고 다음 로우로 이동함으로써, 에러를 처리할 수 있다.

[0057] 위에서 언급한 바와 같이, 변환식은 규칙 세트로부터 생성된다. 규칙 세트는 다음의 속성을 가질 수 있다:

[0058] 명칭, 설명, 및 주석 - 이것들은 규칙 세트를 식별할 수 있게 해준다. 백엔드 구현 방식(back-end implementation)에 따라, 규칙 세트는 시스템 내에서 자신의 위치에 대한 식별정보(identification)를 포함할 수 있다. 어떤 예에서, 규칙 세트의 위치는 프로젝트 내의 경로이다. 어떤 예에서, 규칙 세트는 관계형 데이터베이스(relational database)로 구조화(organized)되고 명칭에 의해 배치될 수 있다. 변경 이력은 변경 명칭, 날짜 및 체크인 주석(check-in comment)를 포함할 수 있다.

[0059] 변환식 타입 - 이것은 규칙 세트로부터 어떤 타입의 변환식이 생성되는지를 결정한다. 가능한 값은 이하에 설명하는 바와 같이, 리포맷(reformat), 조인(join), 롤업(rollup), 및 필터(filter)이다.

[0060] 입력 데이터세트 - 이것들은 편집을 위해 명명된 상수(named constant) 및 필드의 리스트를 제공한다. 어떤 예에서, 변환식이 생성될 때, 변환식은 디폴트에 의해 입력 데이터세트 중 하나에 대한 레코드 포맷을 가정할 것이다. 규칙 세트가 상이한 환경들에 대한 변환식을 생성하는 것을 허용하는 ,복수의 입력 데이터세트가 있을 수 있다. 이것이 또한 논리-물리 매핑(logical to physical mapping)의 복수의 세트, 상이한 물리 명칭의 세트들을 허용한다. 어떤 예에서, 하나 이상의 데이터 세트를 가지는 입력 매핑 테이블이 있다. 어떤 예에서, join(조인) 구성요소는 복수의 매핑 테이블을 가질 수 있고, 각각은 복수의 데이터세트를 가질 수 있다.

[0061] 출력 데이터세트 - 이것들은 출력 필드 명칭의 리스트를 제공한다. 디폴트로, 변환식이 생성될 때, 변환식은 출력 데이터세트 중 하나에 대한 레코드 포맷을 가정할 것이다. 출력 데이터세트는 입력 데이터세트와 동일할 수 있다. 포함된 규칙 세트는 출력 데이터세트를 갖지 않을 것이다. 어떤 예에서, 입력 데이터세트들이 있는 것처럼, 복수의 출력 데이터세트가 있어, 규칙 세트가 상이한 환경에 대한 변환식들 생성할 수 있도록 해준다.

[0062] 포함된 규칙 세트의 리스트 - 하나의 규칙 세트는 다른 규칙 세트에 의해 계산된 출력 필드(출력 레코드 포맷의 필드가 아닌, 분명하게 리스트에 실린 출력 필드)을 사용할 수 있다. 포함된 규칙 세트 내의 출력 변수는, 포함하는 규칙 세트 내에서 알아볼 수 있는(visible) 포함된 규칙 세트로부터 출력 변수의 세트를 규정하는 포함된 규칙 세트 매핑 테이블에 기초하여, 포함하는 규칙 세트 내의 변수로서 사용될 수 있다.

[0063] 포함된 변환식 파일의 리스트 - 규칙 세트를 처리할 때 사용될 변환식을 명시하는 하나 이상의 파일이 선택적으로 포함될 수 있다.

[0064] 변수 및 상수의 리스트를 작성하는 일련의 매핑 테이블 - 이 테이블들은 입력 데이터세트와 출력 데이터세트를 밀접하게 관련되게 한다(intertwined with). 이 테이블들은 편집기에 알려져 있는 변수의 리스트를 만들고 비지니스 명칭(business name)과 기술 명칭(technical name) 간의 매핑을 문서화한다(document). 각 변수는 비지니스 명칭, 기술 명칭(표현식을 사용하여 계산될 수 있음), 그리고 베이스 타입(base type)(스트링, 숫자, 날짜 또는 일시(日時, datetime))을 가질 수 있다. 각 변수와 연관되어 있는 것은 비지니스 명칭과 기술 명칭 간의 매핑을 문서화하는 선택적인 상수의 리스트이다. 변수 테이블에 대해서는 이하에 더욱 상세하게 설명한다.

[0065] 외부 테스트 데이터 파일에 대한 참조 - 전술한 임베디드(embedded) 테스트 데이터세트와 마찬가지로, 테스트 파일은 규칙을 테스트하기 위해 사용된다.

[0066] 거부 불가 플래그(no-reject flag) - 이 플래그가 설정되면, 규칙 세트에 의해 생성된 변환식은 레코드를 거절(에러 발생시킴)하지 않을 것이다. 이것은 에러를 발생시키는(throw) 규칙이 결코 트리거되지 않은 것처럼 무시되도록 하기 위해 사용될 수 있다.

[0067] 디플로이먼트 테이블(deployment table) - 이것은 어떤 규칙들이 각 빌드(build) 내에 포함되어야 하는지를 (간접적으로) 나타내는 하나 이상의 디플로이먼트를 열거한다. 디플로이먼트 테이블은 나중에 더욱 자세하게 설명한다.

[0068] 선택적인 키(optional key) - 이것은 사용자가 조인 타입(join-type) 및 롤업 타입(rollup-type) 규칙 세트를 나타내는 특수한 입력 필드에 대한 비지니스 명칭을 명시할 수 있도록 해준다. 어떤 예에서, 이 키는 실제로는 키의 타입을 가지는, 입력 변수의 테이블에서 엔트리로서 구현된다.

[0069] 선택적인 룩업 파일의 리스트 - 이것은 비지니스 명칭, 키 정보 및 입력 변수 및 정수의 완전한 테이블, 룩업 파일마다 하나의 테이블을 제공한다. 룩업 파일 지원에 대해서는 이하에 더욱 자세하게 설명한다.

[0070] 파라미터 테이블 - 이것은 값이 환경으로부터 또는 실행 시에 파라미터 세트에서 온 변수를 열거한다.

[0071] 규칙 세트는 몇몇 상이한 테이블과 연관되어 있다:

[0072] 1. 입력 변수 및 정수의 테이블. 변환식 타입(transform-type) 규칙 세트의 경우, 이 테이블은 규칙 내에서 참조될 입력 레코드 포맷으로 필드들을 포함한다. 레코드 포맷의 모든 필드가 열거될 필요는 없지만, 보통은 열거된다.

[0073] 2. 모든 포함된 규칙 세트에 대한 입력 변수 및 정수의 테이블. 포함된 규칙 세트들을 사용할 때, 각각의 포함된 규칙 세트는 각자의 입력 변수 및 정수의 테이블을 가지고 있다. 변환식이 만들어질 때, 포함된 규칙 세트들에 의해 사용된 입력 변수는 포함을 실행하는 규칙 세트의 맥락에서 실제 입력에 매핑된다. 그러므로, 이 리스트는 포함하는 규칙 세트로 승격된다(promoted). 복수의 포함된 규칙 세트가 포함되어 있으면, 각각의 입력 변수 테이블이 승격된다(포함된 규칙 세트 자체가 규칙 세트를 포함하면, 제2 단계(second-level) 변수는 승격되지 않는다). 포함된 규칙 세트로부터 승격된 입력 변수 및 정수는 포함하는 규칙 세트에 사용 불가능하다. 이 테이블이 포함되어서, 포함된 규칙 세트에의 입력과 포함하는 규칙 세트에의 입력 사이에 매핑이 이루어질 수 있다. 더욱 자세한 것은 이하를 참조하기 바란다.

[0074] 3. 모든 포함된 규칙 세트의 출력 변수 및 정수의 테이블. 규칙 세트가 포함되었을 때, 그러한 포함된 규칙 세트의 출력은 포함하는 규칙 세트에의 입력이 된다. 이 테이블은 그러한 모든 변수를 열거한다. 이 테이블은 처음에 모든 포함된 규칙 세트 내의 출력 변수 및 정수의 테이블로부터 직접 이주된(populated) 것이지만, 비지니스 명칭은 명칭의 충돌을 피하기 위해 변경될 수 있다. 이 테이블의 경우, 기술 명칭은 포함된 규칙 세트 내부의 비지니스 명칭이다.

[0075] 4. 출력 변수 및 정수의 테이블. 변환식 타입(transform-type) 규칙 세트의 경우, 이 테이블은 규칙 세트에 의해 계산될 출력 레코드 포맷으로 필드들을 포함한다. 계산되지 않는 출력 변수가 포함될 수도 있고 규칙 세트에 의해 무시될 것이다(생성된 변환식은 입력을 출력에 복사하기 위해 와일드카드(wildcard) 규칙을 가진다. 또, 출력은 포함된 디폴트 값들을 가질 수 있다).

[0076] 출력 변수는 또한 뒤의 규칙에서 참조될 수 있는 하나의 규칙으로부터 생성된 출력의 값을 의미하는 매개 변수(intermediate variable)로서 사용될 수도 있다. 때로 출력은 이렇게 사용되고 변환식으로부터의 출력 레코드 내에 직접 포함되지 않는다.

[0077] 5. 파라미터의 테이블. 규칙은 파라미터에 대한 참조를 포함할 수 있다. 파라미터는 그래프의 파라미터 세트의 맥락에서 실행시에 분해된다. 다른 변수들과 유사하게, 규칙 내에서 파라미터는 비지니스 명칭, 기술 명칭(예컨대, $RUNDATE) 및 타입을 가진다.

[0078] 6. 각각의 룩업 파일에 대한 변수 매핑 테이블. 이것들은 입력 테이블과 유사하지만, 룩업 파일의 레코드 포맷으로 필드에 매핑한다.

[0079] 비공유(non-shared) 규칙 세트(변환식을 생성하도록 구성된 것)은 보통 입력 및 출력 데이터세트 양쪽에 관련된다. 입력 데이터세트는 입력 변수들의 소스이다. 출력 데이터세트는 출력 변수들의 소스이다. 때로 규칙 세트는 복수의 입력 데이터세트 및/또는 복수의 출력 데이터세트를 가질 것이다. 그 경우에, 각각의 입력 데이터세트 및 출력 데이터세트는 변환식의 가능한 입력 또는 출력이다. 한 세트의 입력 변수만 있을 수 있지만(join 연산 제외), 상이한 데이터세트의 기술 명칭과 비지니스 명칭 사이에는 상이한 매핑이 있을 수 있다. 어떤 경우에, 입력 변수는 규칙 세트에 의해 사용될 수 있고 하나의 데이터세트 내에 존재하지만 두 번째 입력 데이터세트 내에는 존재하지 않는다. 그 경우에, 표현식은 두 번째 입력 데이터세트 내의 누락 변수의 기술 명칭으로서 지정된다. 규칙 세트가 입력 변수를 사용하지 않는 경우, 모든 입력 데이터세트에 기술 명칭을 제공할 필요는 없다.

[0080] 포함된 규칙 세트는 약간 다르게 취급된다. 포함된 규칙 세트는 연관된 입력 및 출력 데이터세트를 갖지 않을 수 있다. 대신에, 포함된 규칙 세트는 입력 변수들 및 출력 변수들을 가진다. 포함된 규칙 세트를 포함하는 규칙 세트는 입력과 출력의 매핑을 책임진다.

[0081] 변수

[0082] 변수는 다음의 특징을 가질 수 있고, 표 형태로 사용자에게 표시될 수 있다.

[0083] 1. 비지니스 명칭(논리 명칭). 비지니스 명칭은 규칙에 사용된 명칭이다. 어떤 예에서, 두 개의 입력 변수가 동일한 명칭을 가질 수 없고, 두 개의 출력 변수가 동일한 명칭을 가질 수 없으며, 포함된 규칙 세트들로부터의 두 개의 출력이 동일한 명칭을 가질 수 없고, 또한 동일한 룩업 파일 내의 두 개의 룩업 변수가 동일한 변수를 가질 수 없도록 제한이 가해진다. 입력 변수는 출력 변수와 동일한 변수를 가질 수 있다. 이러한

경우에, 사용자 인터페이스는 맥락에 기초하여 또는 출력 변수 명칭의 앞에 "out."과 같은 접두사를 사용함으로써 명확하게 할 수 있다. 상이한 룩업 파일 내의 룩업 변수들은 동일한 명칭을 가질 수 있다. 그러므로, 룩업 파일 자신의 명칭과 같은 접두사의 사용은 룩업 변수들을 명확하게 할 것이다.

[0084] 2. 단순한 타입. 어떤 예에서, 네 가지 기본 타입, 즉 스트링, 숫자, 날짜 및 일시가 지원될 수 있다. 이들은 스트링(int), 십진수(20), 날짜("YYYY-MM-DD") 및 일시("YYYY-MM-DD HH24:MI:SS.nnnnnn") 타입에 대응한다. 변환식에 사용된 기본 타입과 실제 타입 간의 전환은, 예를 들면 생성된 변환식 구성요소에 의해 비지니스 규칙의 편집과는 별개로 취급될 것이다.

[0085] 3. 디폴트 값. 디폴트 값은 출력 변수에만 필요하다. 이것은 (1) 해당 출력에 대한 규칙의 출력 칼럼에 빈 셀(empty cell)이 있는 경우, 또는 (2) 해당 출력에 대한 값을 계산하도록 트리거하는 규칙이 없는 경우에 사용되는 값이다. 디폴트 값은 출력 변수가 널을 허용하는 한, NULL일 수 있다(그리고 빈 셀은 NULL로 해석된다).

[0086] 디폴트 값들은 규칙 표현식 테이블 내의 출력 칼럼들에 사용되어 있는 표현식과 다를바 없는 표현식들이다. 이것이 디폴트 값이 입력 변수 또는 출력 정수를 나타낼 수 있거나 표현식들을 포함한다는 것을 의미한다. 디폴트 값은 또한 순환성(circularity)이 도입되지 않는 한, 다른 출력들을 나타낼 수도 있다.

[0087] 4. 기술 명칭(물리 명칭) 또는 표현식. 이것은 변수를 명시하는 표현식이다. 입력 및 포함된 변수에 대한 필드 명칭 대신에 표현식을 사용하는 것도 가능하다(어떤 예에서는, 출력 변수에 대해 표현식을 사용하는 것은 허용되지 않는다). 벡터의 경우에, 표현식은 정규화되어야 한다(fully qualified).

[0088] 포함된 규칙 세트로부터의 승격된 변수와 입력 및 출력 변수를 처리할 때, 변수와 연관된 기술 명칭은 공유된 규칙 세트 내부에 사용된 바로 그 비지니스 명칭이다. 내부적으로만 사용된 출력 변수(하나의 규칙에서 계산되고 다음의 규칙에서 사용된 매개 변수)을 처리할 때, 기술 명칭은 공백(blank)일 수 있다.

[0089] 5. 선택적인 설명 및 주석

[0090] 주석

[0091] 변수들로 이루어진 변수 테이블은 변수는 물론 정수에 대한 매핑을 포함한다. 상수는 C++에서의 열거형(enum)에 대응한다. 시스템은 처음에 유효 값 및 무효 값에서 비롯되는 상수 값, 및 처음에 유효 범위 및 무효 범위에서 비롯되는 상수 범위를 지원할 수 있다. 또, 시스템은 별개의 값 및/또는 범위의 세트를 나타내는 상수를 생성할 수 있다.

[0092] 상수들은 변수들과 연관되어 있다. 이것이 의미하는 바는, 상수의 비지니스 명칭은 규칙 세트 전체에서 유일할 필요가 없다는 것이다. 편집기는 보통 상수 나타나는 규칙 내의 어느 칼럼에 기초하여 임의의 상수에 대한 전후사정을 알게 될 것이지만, 사용자는 표현식 내의 다른 변수에 속하는 상수를 선택하는 것이 가능하다. 그 경우에, 상수는 변수 명칭(예컨대, "Airline class.business")으로 정규화(qualified)될 것이다.

[0093] 출력 변수를 계산할 때, 단일 값 상수만이 사용된다(출력 필드에 범위를 할당하는 것은 이치에 맞지 않다).

[0094] 상수는 아래의 특성을 가지고, 표 형태로 사용자에게 표시될 것이다(변수 및 상수는 다른 테이블 내부에 테이블을 임베딩하는 것과 유사하게, 섞일 수 있다).

[0095] 1. 변수 명칭. 상수들이 하나의 변수에 적용된다. 변수 명칭은 실제 연관된 변수 자체의 일부이다.

[0096] 2. 비지니스 명칭. 비지니스 명칭은 규칙에 사용된 명칭이다. 명칭은 값 식별자일 필요는 없으며, 구체적으로는 내부 공백(internal space) 및 구둣점이 허용된다. 어떤 경우에, 상수에 대한 비지니스 명칭은 적용되는 변수 내에서만 유일하다.

[0097] 3. 상수 타입. 값, 범위, 또는 세트 중 하나이다. 앞서 설명한 바와 같이, 범위 및 세트는 할당 시에 아니라(출력), 비교 시에(입력)에 사용될 때 합법적이다.

[0098] 4. 값의 경우: 실제 값. 본 예에서, 스트링은 따옴표 안에 있고, 숫자는 그렇지 않다. 날짜 및 일시는 디폴트 형태에서는 따옴표 안에 있다(예컨대, "YYYY-MM-DD"). 표현식이 변수 타입으로 자동 전환될 수 있는 단순한 타입을 반환(회신)하는 한, 표현식의 사용도 허용된다.

[0099] 상수가, 포함된 규칙 세트에 대한 입력 또는 출력의 테이블의 일부인 경우, 값은 없다. 대신에, 그 값은 대응하는 입력 또는 출력 변수에 대한 연관된 상수의 비지니스 명칭이다.

[0100] 5. 범위의 경우: 최대값 및 최소값. 위에 기재된 실제 값과 마찬가지로, 둘 다 상수 또는 표현식이다. 범위는 규칙의 비교에 약기(shorthand)로 사용된다. 동등 비교(equality comparison)만이 범위에 대해 허용되고, 시스템은 범위를 " 변수 >= 최소값 및 변수 <= 최대값"으로 번역할 것이다. 최소값이 지정되지 않는 경우, 그 부분의 비교는 건너뛸 것이다. 최대값에 대해서도 마찬가지이다. 범위는 최대값과 최소값을 분리하는 콤마와 함께 저장된다.

[0101] 6. 세트의 경우: 콤마로 구분된(comma separated) 값의 리스트. 이 리스트의 각 요소는 위에 기재된 실제 값과 마찬가지로, 상수 또는 표현식이다. 동등 비교만이 세트에 대해 허용되고, 시스템은 세트를 "[값의 벡터 리스트]의 변수 요소(variable member)" 형태의 표현식으로 번역할 것이다.

[0102] 7. 선택적인 설명 및 주석

[0103] 공유된 규칙 세트들로부터 확장된 변수들을 처리할 때, 상수도 확장된다. 공유된 규칙 세트에 대한 입력 및 출력 변수를 나타내는 테이블에는, 그러한 변수와 연관된 상수도 나타나 있다. 그러한 상수에 대한 디폴트 매핑은 확장된 정보의 일부이지만, 사용자는 상수 값을 오버라이드(override)할 수 있다.

[0104] 시스템은 상수들의 불일치로 인해 변수의 사용에 있어 충돌 가능성이 있는 경우를 검출할 수 있다. 구체적으로는, (1) 임의의 변수의 값이 다른 변수에 복사된 경우, (2) 두 개의 변수가 정의된 상수를 가지는 경우, 및 (3) 상수의 세트가 명칭 및 값 모두 일치하지 않는 경우이면, 사용자가 하나의 변수에 대한 값을 다른 것에 대한 값들로 바꾸어야 하는 오류가 발생될 것이다. 소스 변수(source variable)는 입력 변수, 룩업 변수, 포함된 규칙 세트로부터의 출력, 및 입력으로 사용된 출력 변수를 포함한다. 타켓 변수는 출력 변수 및 포함된 규칙 세트에의 입력을 포함한다. 할당은 규칙 표현식 또는 변수 테이블에서 일어난다.

[0105] 변수의 정렬(ordering)

[0106] 순환 논리(circular logic)를 피하기 위해, 시스템은 변수 및 규칙의 엄격한 정렬을 실시한다. 전역 정렬의 예는 다음과 같다:

[0107] 입력 변수 및 파라미터.

[0108] 첫 번째 포함된 규칙 세트의 입력 매핑들.

[0109] 첫 번째 포함된 규칙 세트의 출력 값들.

[0110] ...

[0111] N번째 포함된 규칙 세트의 입력 매핑들.

[0112] N번째 포함된 규칙 세트의 출력 값들.

[0113] 첫 번째 룩업 파일의 디폴트 키 값.

[0114] 첫 번째 룩업 파일의 출력 필드들.

[0115] ...

[0116] N번째 룩업 파일의 디폴트 키 값.

[0117] N번째 룩업 파일의 출력 필드들.

[0118] 모든 출력 변수의 디폴트 값.

[0119] 각 항목의 계산에는 이전 단계들에서 계산된 값들을 사용한다. 이것이 예를 들면, 첫 번째 포함된 규칙은 그 매핑 테이블 내의 입력 변수 및 파라미터를 참조할 수 있다는 것을 의미한다. 그러나 두 번째 포함된 규칙은 자신의 입력들을 첫 번째 포함된 규칙으로부터 계산된 출력들에 매핑할 수 있다. 마찬가지로, 출력 변수 각각에 대한 디폴트 값은 모든 규칙 이전에 계산되어서, 모든 포함된 규칙으로부터의 입력 변수의 값, 파라미터, 룩업 파일 또는 출력의 기초가 된다. 실제로 규칙의 출력을 계산할 때가 되면, 규칙들은 차례로 평가되어서 더 뒤의 규칙들은 더 앞의 규칙들로부터 계산된 값들을 사용할 수 있다.

[0120] 변수에의 데이터세트 링킹

[0121] 어떤 예에서, 입력 변수의 테이블은 입력 데이터세트 레코드 포맷으로부터 직접 생겨나고, 비지니스 명칭은 입력 데이터세트에 대한 메타데이터로부터 생겨난다. 그러나, 어떤 예에서는, 규칙 세트의 내부에 이 매핑에 대

한 복사본을 가지는 것에 대한 이점이 있다. 첫째, 규칙 세트 내에 변수 매핑 테이블의 복사본을 가지는 것은 생성 환경의 맥락 밖에서 규칙 세트를 편집하는 것을 가능하게 한다. 규칙 세트 및 연관된 규칙은 샌드박스(sandbox) 내로 연재되고(serialized) 샌드박스 프로젝트(sandbox project)의 일부로서 편집될 수 있다. 둘째, 입력 변수 매핑 테이블의 복사본을 가지는 것은, 사용자가 충돌을 해결하거나 아니면 기존의 메타데이터를 오버라이드하는 것을 가능하게 한다. 예를 들면, 입력 데이터세트 내의 두 개의 필드가 동일한 비지니스 명칭에 매핑되는 경우, 그러한 비지니스 명칭들 중 하나는 입력 변수의 테이블에서 변경될 수 있다.

[0122] 규칙 세트가 먼저 생성될 때, 입력 변수 테이블은 비어있다. 사용자가 입력 데이터세트를 식별하면 바로, 입력 변수 테이블이 입력 데이터세트의 메타데이터로부터 자동으로 옮겨진다(동일한 논리가 출력 변수 및 출력 데이터세트에 적용되지만, 간단히 본 명세서의 나머지 부분에서는 입력 데이터세트에만 집중할 것이다).

[0123] 간단히 본 설명에서는 단수형 용어 "입력 데이터세트"가 사용된다. 입력 변수에 링크될 수 있는 영(zero)개 이상의 데이터세트가 있고, 출력 데이터세트에 링크될 수 있는 영개 이상의 입력 데이터세트의 개별 세트가 있다. 구체적으로는, 입력 변수 테이블은 비지니스 명칭을 위한 하나의 컬럼, 타입 등을 위한 하나의 컬럼, 및 기술 명칭들을 위한 다수의 컬럼을, 입력 데이터세트마다 하나씩 가진다. 일단 단일의 입력 데이터세트가 지정되는 경우, 두 번째 것은 유사한 기법을 사용하여 추가될 수 있다. 그러나, 두 번째 또는 다음의 데이터세트의 경우에는, 특히 시스템이 두 번째 및 다음의 데이터세트 내의 각 필드의 어떤 변수가 매핑하는지를 알 수 없기 때문에, 기술 명칭과 비지니스 명칭 사이의 매핑은 덜 완전할 수 있다. 그러한 예에서, 사용자는 임의의 누락 정보를 수동으로 바로잡을 수 있다.

[0124] 처음에 입력 데이터세트로부터 입력 테이블을 생성할 때, 입력 데이터세트 내의 각 필드는 하나의 입력 변수가 생성되게 할 것이다. 입력 변수에 대한 기술 명칭은 필드의 명칭이 될 것이다. 타입은 필드 타입에 기초하여 할당될 것이다. 빈칸(void)는 스트링처럼 취급될 것이고, 실수(real)는 숫자처럼 취급될 것이다. 서브레코드는 입력 변수와 대응관계를 갖지 않을 것이다. 하지만, 서브레코드 내부의 필드들은 입력 변수와 대응관계를 가질 것이다. 유니온(union)은 유니온의 브랜치(branch) 각각에 대해 입력 변수들을 생성할 것이다. 요소가 벡터인 경우, 대응하는 입력 변수의 기술 명칭은 그 벡터의 첫 번째 요소로 가정될 것이다("in.vect[0]"). 사용자는 이것을 오버라이드할 수 있다. 예를 들면, 다중 출력(multi-output) 변환식의 경우, 사용자는 기술 명칭을 in.vect[index]로 변경할 수 있다. 또는, 사용자는 벡터의 길이가 고정되어 있는 경우, 벡터의 다른 요소들에 대응하는 추가적인 입력 변수들을 생성할 수 있다. 유니온 및 벡터는 출력 데이터세트에서 지원되지 않을 수 있다(그것들에 대해서는 출력 변수들이 생성되지 않을 것이다). 어떤 예에서, 다중 출력 구성요소의 변화(variation)는 다중 출력 레코드 대신 출력 벡터를 출력할 수 있다.

[0125] 어떤 예에서, 비지니스 명칭은 메타데이터로부터 계산된다. 필드에 대한 비지니스 명칭을 결정하는 논리의 일예는 다음과 같다:

[0126] 필드(물리 요소, Physical Element)가 display_name을 가지면, 필드의 display_name이 비지니스 명칭으로 사용된다.

[0127] 그렇지 않고, 필드가 논리 요소(Logical Element)를 가지고 그 논리 요소가 display_name을 가지는 경우, 논리 요소의 display_name이 비지니스 명칭으로 사용된다.

[0128] 그렇지 않고, 필드가 논리 요소를 가지는 경우, 논리 요소의 명칭이 비지니스 명칭으로 사용된다.

[0129] 그렇지 않으면, 비지니스 명칭은 기술 명칭으로부터 계산된다.

[0130] 충돌(명칭 중복)이 있으면, 하나의 비지니스 명칭만이 할당될 수 있다. 나머지 필드들은 어떤 비지니스 명칭이 할당되지 않을 것이다.

[0131] 어떤 예에서는, 규칙 세트들과 데이터세트 메타데이터 사이에 동적 링크가 존재하지 않는다. 사용자가 메타데이터 데이터를 변경하는 경우(예를 들면, 논리 요소를 재명명), 그 변경을 시스템에 의해 자동으로 알게 되지는 않는다. 어떤 예에서, 데이터 간의 양방향 관계(two-way relationship)는 그러한 변경이 검출될 수 있도록 하기 위해 사용될 수 있다.

[0132] 사용자가 규칙 세트에 두 번째 데이터세트를 추가하는 경우, 시스템은 앞서 열거한 바와 같이 동일한 물리-논리(physical to logical) 매핑을 사용하여 비지니스 명칭 각각에 대한 필드들 내를 채우려고 할 것이다. 변수가 매핑될 수 없는 경우, 그 변수에 대한 기술 용어는 추가된 데이터세트에 대해 공백으로 남을 것이고, 사용자는 수동으로 필드 명칭 또는 표현식을 채워야 할 것이다. 이용 가능한 필드는 사용자 인터페이스에서 풀다운으로

열거될 것이다.

[0133] 입력 변수 테이블이 데이터세트 메타데이터로부터 생성되는 것과 동시에, 정수들이 또한 데이터세트 메타데이터로부터 입력 변수 테이블에 추가될 수 있다. 각 논리 요소(Logical Element) 또는 물리 요소(Physical Element)와 연관된 유효 규격(Validation Spec)과 연관된 모든 유효 및 무효 값과 모든 유효 및 무효 범위들에 대한 상수를 생성할 것이다.

[0134] 상수에 대한 비지니스 명칭을 결정하는 논리의 일 예는 다음과 같다:

[0135] 유효 값(유효 범위, 등)이 display_name을 가지는 경우, 그 display_name이 비지니스 명칭으로 사용될 것이다.

[0136] 그렇지 않고, 유효 값(유효 범위, 등)이 설명부(description)를 가지는 경우, 그 설명부가 비지니스 명칭으로 사용될 것이다.

[0137] 그렇지 않으면, 상수는 비지니스 명칭 없이 변수 테이블에 포함될 것이다.

[0138] 데이터세트들로부터 시작하는 변수들을 반드시 생성할 필요는 없다. 입력 변수의 리스트를 생성하는 두 번째 방법은 기본 시스템(underlying system) 내의 논리 엔티티(Logical Entity)를 식별하는 것이다. 논리 엔티티가 선택되면, 시스템은 논리 엔티티 내의 논리 요소 각각에 대해 하나의 변수를 가지는 변수들의 테이블을 생성할 것이다. 변수의 비지니스 명칭은 논리 요소(Local Element)의 display_name이 될 것이다. 논리 요소가 유효 규격(Validation Spec)를 가지는 경우, 상수는 또한 이전의 문서 규칙들을 사용하여 생성될 것이다.

[0139] 끝으로, 입력 및 출력 변수는, 변수 테이블에의 추가 또는 규칙의 편집 중 생성에 의해, 수동으로 추가될 수 있다. 예를 들면, 사용자가 칼럼을 규칙에 추가할 때, 그 칼럼에 어떤 입력 변수를 사용할 것인지를 선택한다. 그러나 사용자는 또한 "새로운..."를 선택하여 실시간(on the fly) 입력 변수를 생성할 수도 있다. 그러면 시스템은 사용자에게 데이터타입 및 선택적인 주석에 대해 재촉할 것이다. 기술 명칭은 그 뒤까지 채워질 필요가 없다.

[0140] 시스템은 규칙을 편집할 수 있도록 하기 위해 변수의 리스트를 필요로 한다. 그러나, 비지니스 명칭과 기술 명칭 사이의 매핑은 그 뒤까지 완전해지지 않아도 된다. 매핑은, 사용자가 외부 테스트 파일에 대해 규칙 세트 전체를 테스트할 준비가 되어 있거나 실제로 규칙 세트로부터 변환식을 생성할 준비가 되어 있을 때에만 필요할 수 있다.

[0141] 포함되는 규칙 세트

[0142] 어떤 예에서, 규칙 세트들은 공유될 수 있다. 구체적으로는, 포함되는 규칙 세트는 다른 규칙 세트 내부에 포함되도록 설계되므로 그 논리는 포함하는 규칙 세트의 생성된 변환식의 일부분이 된다.

[0143] 포함되는 규칙 세트들은 보통 공유되도록 특별히 고안되지만, 포함되는 규칙 세트는 또한 단독으로 사용되어 변환식을 생성할 수도 있다. 예를 들면, 사용자는 필터 타입(filter-typ) 변환식에 대해 불 방식의 출력(Boolean output)을 계산하는 규칙세트를 생성할 수 있다. 그러나, 동시에, 그 규칙 세트는 다른 변환식 내에 포함될 수 있고 불 방식의 출력은(포함하는 규칙 세트에서 이용 가능한, 공유된 규칙 세트의 출력 변수)는 보다 복잡한 출력을 계산하는 데 사용될 수 있다.

[0144] 포함되는 규칙 세트들은 다른 타입의 규칙 세트들과 비슷하다. 포함되는 규칙 세트들은 입력 변수들 및 출력 변수들을 가진다. 또 포함되는 규칙 세트들은, 자신도 다른 포함되는 규칙 세트들을 포함할 수 있다. 그러나 포함된 규칙 세트 내의 입력 및 출력 변수의 취급은 변환식 타입(transform-type) 규칙 세트와 다르다. 변환식 타입 규칙 세트에서는, 입력 및 출력 변수를 기술 명칭에 매핑하여 변환식을 생성할 수 있다. 그러나 포함된 규칙 세트에서는 입력 및 출력 변수를 기술 명칭에 매핑할 필요가 없다(규칙 세트가 공유되고 또한 변환식의 생성에도 사용되면, 입력 및 출력 변수는 변환식을 생성하는 디플로이먼트의 기술 명칭에 매핑될 것이다.

[0145] 사용자가 다른 규칙 세트에 포함되는 규칙 세트를 포함할 때, 포함하는 규칙 세트는 포함되는 규칙 세트의 입력 및 출력을 매핑하기 위한 변수 매핑 테이블을 가져야 한다. 포함하는 규칙 세트의 맥락에서, 공유된 규칙 세트의 입력 변수 및 출력 변수만이 가시적이다. 공유된 규칙 세트 내에 포함된 모든 규칙 세트의 모든 변수는 포함하는 규칙 세트에 대해 드러나지 않는다.

[0146] 포함하는 규칙 세트의 맥락에서, 공유된 규칙 세트의 입력 변수들은 포함하는 규칙 세트의 변수 또는 그러한 변수를 사용하는 표현식에 미핑되어야 한다. 공유된 규칙 세트의 비지니스 명칭은 변수 매핑 테이블 내에 열거될

것이지만, 그러한 명칭은 포함하는 규칙 세트 내의 규칙에 사용될 수 없을 것이다. 대신에, 포함하는 규칙 세트는 공유된 규칙 세트의 각각의 입력 변수(비지니스 명칭에 의하는 것)와 포함하는 규칙 세트 내의 표현식을 일치(match)시킬 필요만 있을 수 있다.

[0147] 포함되는 규칙 세트는 입력 변수, 파라미터 및 룩업 이전에 평가되는 것으로 생각되므로 포함되는 규칙 세트의 출력은 룩업에 대한 키로 사용될 수 있다. 어떤 예에서, 평가의 순서는 더욱 유통성이 있고, 포함되는 규칙 세트의 평가에 대한 룩업의 정렬(ordering)은 의존성 분석에 기초하여 자동으로 결정될 수 있다. 포함되는 규칙 세트의 임의의 출력 변수들이 계산되기 이전에 평가되기 때문에, 포함하는 규칙 세트 내의 출력 변수들은 포함되는 규칙 세트 내의 입력들에 매핑될 수 없다. 포함되는 규칙 세트의 입력에 대한 매핑이 단순한 입력 변수를 사용하여 수행될 수 없는 경우, 표현식이 대신 사용될 수 있다.

[0148] 포함되는 규칙 세트의 입력 변수가 널을 허용한다면, 포함된 규칙 세트 입력 변수에 대한 매핑은 NULL일 수 있다. 이 매핑은 또한 공백으로 둘 수도 있다. 매핑을 공백으로 두면, 포함하는 규칙 세트의 출력의 계산에 입력 변수가 필요한 경우에만, 변환식 생성시에 에러가 보고될 것이다. 어떤 예에서, 모든 것이 널을 허용하는 것으로 가정되는데, 이는 사용자 인터페이스를 단순화 한다.

[0149] 포함하는 규칙 세트의 맥락에서, 공유된 규칙 세트의 출력 변수들은 또한 포함하는 규칙 세트 내의 비지니스 명칭에 매핑되어야 한다. 이 매핑 테이블은 위의 것과 정반대이다. 공유된 규칙 세트의 입력 변수들을 매핑할 때, 테이블은 공유된 규칙 세트의 입력 변수의 비지니스 명칭을 포함하는 규칙 세트 내의 기존의 변수에 매핑한다. 그러나 공유된 규칙 세트의 출력 변수들을 매핑할 때, 포함하는 규칙 세트는 공유된 규칙 세트들의 출력들에 대한 비지니스 명칭을 명시하는 테이블을 가진다 - 포함하는 규칙 세트 내의 명칭들을 공유된 규칙 세트 내의 대응하는 명칭들에 매핑시킴.

[0150] 출력 변수 매핑은 잠재적인 명명 충돌(naming conflict)을 해결하여야 한다. 디폴트 매핑은 포함하는 규칙 세트와 공유된 규칙 세트 양쪽에서의 동일한 비지니스 명칭들의 사용을 단순화하는 것이다. 그러나 공유된 규칙 세트 내의 출력 변수들의 명칭들은 포함하는 규칙 세트 내에 이미 정의된 변수들의 비지니스 명칭들과 충돌할 수 있으므로, 규칙 세트 내에서의 매핑된 명명(명칭 부여)는 변경될 수 있다.

[0151] 공유된 규칙 세트로부터의 모든 출력이 매핑되어야 하는 것은 아니다. 출력이 매핑되지 않은 채로 남겨지는 경우, 그 출력은 포함하는 규칙 세트에서 사용될 수 없고, 공유된 규칙 세트로부터의 대응하는 논리는 무시될 것이다. 한편, 공유된 규칙 세트로부터의 모든 입력은, 규칙 세트의 설계자가 그것들이 필요하지 않을 것으로 확신하는 경우에 관심 대상이 아닌 변수(uninteresting variable)에 매핑될 수 있긴 하지만, 매핑될 수 있다. 어떤 예에서, 시스템은 스스로 어느 입력들이 실제로 매핑되어야 하는지를 결정할 수 있다.

[0152] 어떤 예에서, 매핑 테이블은 비지니스 명칭에 의해 완료되며, 참조에 의해 완료되지 않는다. 공유되는 규칙 세트가 다른 규칙 세트에 포함될 때, 포함하는 규칙 세트는 공유된 규칙 세트로부터 입력 및 출력의 복사본을 얻는다. 이들 명칭은 매핑 정보와 함께 포함하는 규칙 세트 내에 저장된다. 공유된 규칙 세트의 변경은, 어떤 입력 또는 출력의 추가, 삭제 또는 재명명(rename)을 초래할 수 있다.

[0153] 포함하는 규칙 세트와 포함되는 규칙 세트 사이의 참조 무결성(referential integrity) 문제는, 규칙 세트가 시스템으로부터 로딩될 때 포함하는 규칙 세트에 의해 처리될 수 있다. 공유된 규칙 세트에서 사라진 입력 변수는 포함하는 규칙 세트에서 삭제된다. 공유된 규칙 세트에 추가된 입력 변수는 포함하는 규칙 세트의 매핑 테이블에 추가되지만, 여전히 매핑되지 않은 상태이다. 출력 변수가 공유된 규칙 세트에서 삭제되고, 또한 포함하는 규칙 세트에서 사용되지 않는 경우, 매핑 테이블에서 바로 삭제지만, 포함하는 규칙 세트에서 사용되는 경우, 사용자는 그 변수가 더 이상 유효하지 않다는 에러를 입수한다.

[0154] 포함하는 규칙 세트는 실제로 공유된 규칙 세트로부터의 중복 정보(redundant information)를 유지한다. 구체적으로는, 입력 및 출력 변수 매핑 테이블에서, 포함하는 규칙 세트는 공유된 규칙 세트 내의 비지니스 명칭의 리스트만을, 포함하는 규칙 세트 내에 대응하는 명명된 것과 함께 유지하여 할 수 있다. 효율성을 위해, 포함하는 규칙 세트는 또한 유형, 디폴트 값, 설명 및 주석, 공유된 규칙 세트의 모든 복사본을 유지한다. 이 값들은 포함하는 규칙 세트를 편집할 때에만 판독되지만, 효율성을 위해 보고서 및 다른 분석결과의 생성에 포함된다.

[0155] 공유된 규칙 세트 매핑 테이블은 또한 추가적인 주석인, 포함된 규칙 세트 내의 하나의 추가 엔트리를 가진다. 이로써 사용자는 매핑된 값에 다른 주석을 추가할 수 있게 된다.

[0156] 룩업 파일

[0157] 규칙 세트는 선택적으로 하나 이상의 룩업 파일을 가질 수 있다. 규칙 세트 내의 각각의 룩업 파일은 다음의 정보를 포함한다:

[0158] 1. 룩업 파일의 비지니스 명칭.

[0159] 2. 선택적인 설명 및 주석.

[0160] 3. 키를 구성하는 필드들에 대한 비지니스 명칭의 리스트. 이 명칭들은 룩업 파일이 표현식에 추가될 때 사용되어서 사용자는 이와 유사한 것을 본다: lookup(My Lookup File, <customer name key>, <account type key>).

[0161] 4. 키 각각에 대한 디폴트 표현식의 리스트.

[0162] 5. 룩업 파일의 기술 명칭. 어떤 예에서, 이것은 디플로이먼트에서 오버라이딩될 수 있다.

[0163] 6. 하나 이상의 룩업 데이터세트. 각각의 룩업 파일은, 규칙 세트가 입력 데이터세트와 결합되는 것처럼 시스템 내의 데이터세트에 느슨하게 결합된다(loosely tied). 디폴트로, 규칙 세트 내의 룩업 파일 각각과 연관된 하나의 룩업 데이터세트가 존재하지만, 다른 디플로이먼트들에서 사용하기 위해 더 많은 룩업 데이터세트가 존재할 수 있다.

[0164] 7. 입력 변수 및 상수의 테이블. 이 테이블은 각각의 룩업 파일에 대해 하나의 테이블이 존재하는 것을 제외하고는 룩업 세트의 입력 변수 및 상수의 테이블과 비슷하다. 입력 변수가 그러하듯, 룩업 파일에 대한 입력 변수 및 상수의 테이블은 연관된 룩업 데이터세트 각각에 대응하는, 복수의 기술 명칭을 가질 수 있다.

[0165] 하나 이상의 룩업 파일이 존재할 수 있다는 것 외에는, 룩업 파일은 입력 변수와 비슷하게 처리된다. 각각의 룩업 파일은 하나의 페이지 상에서 편집되고, 비지니스 명칭과 기술 명칭 간의 매핑 테이블을 가지며, 복수의 데이터세트와 연관될 수 있다. 룩업 파일은 또한 각각의 필드와 연관된 상수를 가진다. 룩업 파일에 대한 매핑은 입력 변수에 대한 메타데이터를 입력 데이터세트로부터 로딩하는 방식으로 룩업 데이터세트에 대한 메타데이터를 판독함으로써 초기화될 수 있다.

[0166] 사용자가 룩업 필드 변수를 사용하고, 그 키는 룩업에서 발견되지 않는 경우, 그 필드의 값은 널로 가정된다. 규칙 사례가 명확하게 트리거되지 않는 한, 필드가 널인 경우에는, 그 규칙 사례는 거짓으로 평가되어 건너뛰게 될 것이다. 그러한 경우에, 에러는 발생되지 않는다. 사용자가 룩업 파일 변수(룩업 파일 자체이고 필드가 아님)를 사용하면, 함수 look_match가 가정되어 룩업 파일 변수는 참 또는 거짓을 평가한다. 사용자가 룩업 필드 변수를 출력 변수의 디폴트로 사용하는 경우, 룩업의 찾기 실패는 NULL로 바뀐된다.

[0167] 참조 파일에서 하나 이상의 데이터 레코드를 검색하기 위해 키를 사용하는 룩업 동작은, 보통의 처리에 비해 느리다. BRE는 각각의 레코드에 대한 룩업 결과를 캐시에 저장(caching)함으로써 비용이 드는(expensive) 룩업 동작의 수를 제한하도록 설계된 코드를 포함한다. 규칙이 룩업 변수(룩업 동작에 의해 반환될 것인 값들 중 하나)를 참조할 때마다, 변환식 생성 프로세스는 룩업 동작을 서브루틴 호출(subroutine call)로 바꿔 놓는다. 서브루틴은 그 서브루틴이 현재의 레코드에 대해 이미 호출되었는지를 나타내는, 모든 레코드의 개시시에 거짓으로 초기화된, 전역 불 변수(global Boolean)을 포함한다. 처음 룩업 서브루틴이 호출될 때, 불 변수(Boolean)는 거짓일 것이다. 이 예에서는, 이 불 변수를 참으로 설정한다. 그후, 실제 룩업 동작이 수행되고 룩업 호출에 의해 반환된 레코드는 캐시에 변수로 저장된다. 끝으로, 테스팅이 인에이블될 때, 룩업 동작의 결과가 이벤트 로그(event log)에 추가된다.

[0168] 그 동일한 레코드를 처리하는 동안의 모든 다음 룩업 동작은 동일한 서브루틴을 불러올 것이다(invoke). 그러나, 그 다음의 서브루틴 호출에 대해, 불 변수는 참이 될 것이다. 이것이 서브루틴의 거동을 변화시켜, 중복 룩업 동작을 하는 대신에 이전에 판독하여 캐시에 저장된 데이터가 반환될 수 있다(또한 추가 로그 이벤트의 생성을 회피한다).

[0169] 간단히, 캐시에의 저장은 하나의 키 값에 대해서만 이루어진다. 상이한 키 값들(동일한 레코드 내에서) 사용하여 동일한 룩업 파일에 대해 복수의 참조가 이루어지는 경우, 그러한 룩업 결과 중 하나만 캐시에 저장될 것이다. 다른 모든 룩업 서브루틴 호출은 실제의 룩업 동작으로 바뀔 것이다. 그렇긴 하지만, 당업자라면 단순한 변수 대신에 캐시에 저장된 결과들에 대한 해시 테이블(hash table)을 사용함으로써, 이것이 어떻게 상이한 키들을 사용하여 복수 룩업을 지원하기 위해 확장될 수 있는 지를 알아야 한다.

[0170] 파라미터

[0171] 규칙 세트는 파라미터를 나타낼 수 있다. 어떤 예에서, 각각의 규칙 세트는 변수 테이블처럼 파라미터의 비지니스 명칭을 그 기술 명칭에 매핑하는 선택적인 파라미터 테이블을 가진다. 파라미터 테이블 내의 각 엔트리는 다음의 속성을 가진다:

[0172] 1. 비지니스 명칭. 규칙 본체(rule body)에 나타날 때, 이것은 파라미터의 명칭이다. 일반적으로 파라미터는 입력 변수가 사용되는 곳 어디든 사용될 수 있다.

[0173] 2. 기술 명칭. 이것은 개발 환경(development environment)에서의 파라미터의 명칭이다.

[0174] 3. 파라미터의 타입(스트링, 십진수, 날짜 또는 일시). 생성된 변환식에서, 파라미터들은 필요에 따라 다른 타입으로 변환될 수 있다.

[0175] 4. 선택적인 설명 및 주석.

[0176] 파라미터들은, 그 값이 입력 파일 전체에 걸쳐 일정하고, 또한 처리가 시작될 때 그 값이 외부에서 지정되는 것을 제외하고는 변수와 비슷하다.

[0177] 테스팅 규칙 및 로깅

[0178] 변환식을 생성하는 부분은, 변환식이 대응할 것인 규칙의 테스팅을 포함한다. 규칙은 또한 인증된다, 즉 구문(syntax) 및 의미 일관성(semantic consistency)에 대해 검사된다. 인증(validation)과 달리, 테스팅은 규칙의 실행을 포함하고 사용자에 의해, 예를 들면 예상된 출력을 제공함으로써 또는 출력을 예산된 값과 수동으로 비교함으로써, 정확성(correctness)이 결정된다.

[0179] 시스템은 테스팅을 두 가지 레벨로 지원한다. 앞서 설명한 바와 같이, 각각의 규칙은 값 및 예상되는 결과의 임베디드(embedded) 테이블 형태로, 연관된 테스트 데이터세트를 가질 수 있다. 이것은 단위 테스팅(unit testing)이라고 한다. 규칙을 편집할 때, 테스트 데이터의 각 라인에 대한 규칙의 출력을 재평가할 수 있다. 실제 결과와 예상되는 결과 사이의 어떤 불일치 또는 유효한 결과 생성에 대한 실패는 해결(resolution)을 위해 강조된다.

[0180] 어떤 예에서, 외부 입력 테스트 파일은 표준 메커니즘을 사용하는 서버 로세스에 대해 액세스 가능하다. 외부 파일을 사용한 테스팅은 파일 테스팅이라고 불린다. 테스트 파일은 규칙 세트에 대한 입력 데이터세트와 일치하는 레코드 포맷을 가진다. 어떤 예에서는 다른 레코드 포맷을 제공할 수 있다. 선택적으로, 사용자는 예상되는 결과를 포함하는 데이터세트를 식별할 수 있다. 시스템은 테스트 데이터세트에 대해 규칙 세트를 실행하고 어떤 결과가 생성되었고 왜 그런지를 표시한다. 예상된 결과가 포함된 경우, 시스템은 실제 결과를 예상된 결과와 비교하고 상이하였던 모든 레코드를 열거한다. 어떤 예에서는, 사용자가 개개의 값을 증분적으로 재훈련(incrementally retrain)할 수 있도록 인터페이스를 확장할 수 있다.

[0181] 단위 테스팅과 파일 테스팅 간의 몇가지 차이점은 다음을 포함한다:

[0182] 1. 룩업 파일의 경우: 단위 테스팅 모드에서는, 각각의 테스트 사례(test case)에 대해, 각각의 룩업 변수에 대한 값은 테스트의 일부로서 정의된다. 키는 지정되지 않는다; 테스트를 실행할 때, 동일한 값이 각각의 테스트 사례에 대해, 각각의 룩업 변수에 대해 가정된다. 테스트 데이터세트는 복수의 테스트 사례를 포함하고, 각각의 테스트 사례는 각각의 룩업 변수에 대해 상이한 값을 지정할 수 있다. 파일 테스팅 모드에서는, 실제 룩업 파일을 사용한다. 이것은 상이한 키들은 상이한 값들을 반환할 것임을 의미하지만, 또한 특정한 키에 대한 임의의 주어진 룩업 변수에 사용된 값이 테스트 중에 변경될 수 없다는 것을 의미하기도 한다.

[0183] 2. 포함되는 규칙 세트들의 경우: 단위 테스팅 모드에서는, 포함되는 규칙 세트는 실행되지 않으며 완전할 필요도 없다. 대신, 값은 각각의 포함되는 규칙 세트로부터의 출력 각각에 대한 테스트 데이터세트 내에 지정된다. 파일 테스팅 모드에서, 포함되는 규칙 세트는 생성시에 실행되었을 것인 그 방식으로 실행된다. 이것은 포함되는 규칙 세트들에 의해 요구된 모든 룩업 파일 또는 파라미터도 테스트시에 지정되어야 한다는 것을 암시한다.

[0184] 3. 파라미터의 경우: 단위 테스팅 모드에서, 각각의 테스트 사례에 대한 각각의 파라미터에 대해 상이한 값이 설정될 수 있다. 파일 테스팅 모드에서는, 각 파라미터의 값은 전체 테스트 동안에 불변이다.

[0185] 4. 현재 날짜의 경우: 테스팅할 때, 규칙들이 현재의 날짜 또는 시각을 나타내는 경우에, 사용자는

현재의 날짜 및 시각에 대해 가정되어야 하는 값을 지정한다. 단위 테스팅 모드에서는, 날짜 및 시각은 각 테스트 사례에 대해 다를 수 있다. 파일 테스팅 모드에서는, 전체 테스트에 대해 단일의 날짜 및 시각이 설정된다(이 값은 테스트가 실행될 때의 기계의 날짜 및 시각으로 다를 수 있다).

[0186] 5. 레코드 포맷 및 매핑의 경우: 단위 테스팅의 경우 매핑은 지정될 필요가 없다; 테스팅은 전적으로 변수들의 비지니스 명칭들에 기초하여 실행된다. 파일 테스팅의 경우, 모든 변수가 기술 명칭에 매핑되고 입력, 출력, 및 룩업의 레코드 포맷이 지정된다.

[0187] 규칙 세트는 도 3에 도시된 흐름도(300)에 예시된 바와 같은 커스터마이즈된 로깅 특성(customized logging characteristic)을 사용하여 테스트 및 인증된다. 단계 302에서 하나 이상의 규칙 세트 및 대응하는 테스트 데이터를 입력으로 수신한다. 단계 304에서 생성기가 규칙 세트들에 기초하여 변환식을 생성하고, 변환식을 사용하여 테스트 데이터 내의 모든 레코드에 대한 출력 값을 계산한다. 선택적으로, 단계 306에서 사용자는, 생성된 변환식을 포함하는 그래프 기반 계산에 대한 단계 308에서의 트레이싱되는 실행에 의해 생성될 로그의 특성을 설정할 수 있다. 그래프 기반 계산에서, 관련 구성요소의 "로그" 포트가 테스팅 출력에 사용된다. "로그" 포트는 구성요소들을 위한 대역외(out-of-band) 통신 채널이다. 이것은 실제 출력 데이터의 레코드 포맷의 변경을 필요로 하지 않는 변환식으로부터 추가의 출력 메타데이터를 얻는 방법이다. 단계 310에서는 실행하는 동안에 액세스 가능한 스토리지 시스템에 저장되는 로그 파일로 수집되는 로그 메시지로서, 로깅 정보를 로그 포트들로부터 출력한다. 선택적으로, 로그 메시지는 필터링될 수 있고 그 서브세트는 로그 파일에 저장된다. 예를 들면, 생성된 변환식을 포함하는 구성요소는 대부분의 입력 레코드들을 변경되지 않은 출력 레코드들로서 통과시킬 수 있다(예컨대, 주어진 입력 레코드에 대해 어떠한 규칙 사례도 트리거되지 않은 경우). 입력 레코드의 필드들 내의 하나 이상의 값을 변경하는 트리거된 규칙 사례에 대응하는 긴 메시지만들 저장하는 것이 바람직할 수 있다. 그래프 내의 하나 이상의 구성요소가 로그 포트로부터의 레코드를 필터링하여, 그러한 로그 메시지만이 로그 파일에 기록되도록 할 수 있다.

[0188] 테스팅을 위해 생성된 변환식은 정상 실행을 위해 생성될 변환식과는 약간 다를 수 있다. 예를 들면, 정상 실행 중에, 변환식을 사용하는 구성요소는 주어진 출력 포트에서 주어진 타입의 출력 레코드를 생성할 수 있다. 테스트시에, 출력 레코드가 변경되지 않고 있는 동안에, 출력에 관련된 테스팅은 로그 포트에 전송될 수 있다.

[0189] 첫 번째 테스트 사례로부터 시작하여, 테스트 사례의 입력이 변환식에 입력되고, 어떤 규칙이 출력을 생성하였는지에 대한 표시와 함께 출력이 출력 어레이에 기록된다. 이 프로세스는 최종 로우(row)가 평가될 때까지 각각의 로우에 대해 반복된다. 그후 출력 어레이는 전술한 바와 같은 결과 테이블들을 생성하기 위해 사용될 수 있다. 출력 어레이는 규칙 세트가 유효한지를 판정하기 위해 평가될 수 있다. 하나의 테스트로부터 생성된 출력 값들이 이전의 테스트에서 생성된 값들과 비교될 수 있는 경우에, 출력 값들은 테스트 데이터 내에 포함될 수 있다. 출력 어레이의 첫 번째 로우에서 시작하여, 생성된 출력은 테스트 데이터로부터의 예상된 출력 또는 이전의 테스트 결과들과 비교된다. 어떠한 출력도 일치하지 않는 경우, 불일치가 로깅된다. 각 로우마다 이를 반복한다. 어떤 예에서, 평가 단계들은 출력 생성 단계들 내에 통합되고, 각각의 생성된 출력은 생성될 때 대응하는 예상된 출력과 비교된다. 테스트 데이터 결과들 처리 시의 임의의 불일치 또는 다른 에러는 로깅된다. 앞서 언급한 바와 같이, 하나의 규칙 세트에 대한 출력들은 다른 규칙 세트들에 대한 입력이 될 수 있고, 그 경우에 포함된 규칙 세트는 포함하는 규칙 세트의 일부로서 평가된다.

[0190] 사용자는 출력 필드에 의해 또는 입력 필드들을 사용할 수 있는 표현식에 의해 테스트되는 규칙들을 제한할 수 있다. 어떤 예에서, 사용자는 테스팅하는 동안에 규칙을 디스에이블하도록 선택할 수 있다. 어떤 예에서, 사용자는 처리될 전체 테스트 파일을 기다릴 필요가 없으며; 테스트 결과들은 처음의 몇 개의 레코드가 작용하면서 출력에 나아가면 바로 이용 가능하다.

[0191] 테스트 데이터 자체 외에, 다음의 정보 중 어느 것이 파일 테스팅 및 로깅을 위해 단계 308에서 트레이싱될 수 있다:

[0192] 1. 입력 데이터세트의 물리 위치(Physical Location). 이것은 입력 데이터세트 각각에 대한 입력 변수 테이블에서 규칙 세트에 저장된다. 결합 타입 데이터세트들의 경우, 모든 물리 위치가 필요하다. 물리 위치가 필요할 때마다, 데이터베이스 내의 테이블 명칭이 사용될 수 있다.

[0193] 2. 입력 데이터세트의 레코드 포맷. 디폴트로, 이것은 입력 데이터세트의 데이터세트 정의로부터 얻어진다. 입력 변수 내에는 확인된(checked-out) 다른 레코드 포맷을 사용하여 이것을 샌드박스 내로 오버라이드하기 위한 장소가 있다.

[0194] 3. 어떠 디플로이먼트를 사용하는지.

[0195] 4. 모든 룩업 파일에 대한 물리 위치. 이것은 룩업 파일들의 테이블에 저장된다.

[0196] 5. 룩업 파일 각각에 대한 레코드 포맷. 각각의 룩업 파일과 연관된 데이터세트 정의 또는 오버라이드 레코드 포맷 파일로부터 얻는다.

[0197] 6. 파라미터 각각에 대한 값. 이것은 테스트 파라미터 다이얼로그(dialog)에서 설정될 것이다.

[0198] 7. 출력 파일에 대한 물리 위치. 이것은 회귀(비교) 테스팅할 때만 필요하다. 이것은 출력 변수 테이블에 저장된다.

[0199] 8. 출력 파일에 대한 레코드 포맷. 다시 한번, 회귀(regression) 테스팅할 때만 필요하고, 출력 데이터세트 정의 또는 선택적인 오버라이드 레코드 포맷 파일로부터 얻는다.

[0200] 9. 프로젝트 샌드박스의 위치. 테스팅은 호스트 상의 샌드박스 밖에서 수행되어야 한다. 샌드박스는 규칙 세트를 포함하는 프로젝트의 확인된 복사본이어야 한다.

[0201] 10. 규칙이 "지금", "오늘" 또는 이와 유사한 값을 나타낼 때 날짜 및 시각에 사용하는 값.

[0202] 이 예에서, 변환식은 디폴트로 셀 상태를 로깅하지 않지만, 이 함수는 도 4에 도시된 바와 같은 사용자 인터페이스(400)에서 활성화될 수 있다. 즉, 사용자는 여러 특정한 기준의 로깅의 온 및 오프와 같은 로그 특성을 단계 306에서 설정하는 것이 허용된다. 인터페이스(400)는 로그 메시지("로그 레코드"라고 함)가 생성될 때(각각의 입력 레코드에 대해, 에러가 발생한 때 또는 특정한 표현식이 참일 때 포함함)를 지정하기 위한 섹션(410), 및 각각의 로그 메시지에 포함되는 것(예컨대, 규칙 사례 활성화(firing), 입력 값, 출력 값, 룩업 값, 포함되는 규칙 세트들을 형성하는 출력 값들, 파라미터 값들, 및/또는 각 셀들의 평가 결과)을 지정하기 위한 섹션(420)을 포함한다. 입력들, 출력들, 등의 로깅은 실행 속도를 떨어뜨리긴 하지만 약간일 뿐이다. 셀 상태의 로깅은 실행 속도를 상당히 떨어뜨리는데, 아마도 10배 정도 떨어뜨릴 것이다. 테스팅이 수행될 것이고 검색된 결과들은 특정한 구성에 따라 표시될 것이다.

[0203] 테스팅 및/또는 로깅을 하지 않을 때, 생성기는 입력, 출력, 등을 로깅하는 변환식을 여전히 생성할 수 있고, 그 변환식을 생성에(in production) 사용할 수 있다. 강화된(enhanced) 변환식은 동일한 출력들을 생성할 뿐아니라 어떤 규칙들이 실행되었는지를 결정하기 위해 실행 후(post-execution) 분석될 수 있는 일련의 로그 메시지를 생성하기도 한다. 사용자가 단계 310에서 생성된 로그 메시지를 로그 파일로 저장한다면, 테스팅 입력이 사용되지 않았더라도, BRE가 규칙 세트의 실행을 재현한 후 생성에 사용될 수 있다. 이 실행은 재생(playback)이라고 불리며 감사에 유용하다. 로그 파일은 전술한 바와 같은 입력, 출력, 룩업, 트리거된 규칙 사례, 등의 세부사항을 포함한다. 로그 파일은 또한 정확히 어느 버전의 규칙 세트(명칭 및 버전)가 로그 파일을 생성한 변환식을 생성하기 위해 사용되었는지를 기록하는 헤더를 가진다. 생성에 있어 실행할 때, 고객은 로그 포트의 출력 파일(선택적으로 압축함)로 저장하여야 한다. 세션을 재생하려면, 감사를 위해, 고객은 BRE를 개시한 다음 재생 세션을 시작할 것이다. 고객은 세션 로그를 포함하는 파일을 식별한다. 그후 BRE는 헤더를 읽어, 표시된 규칙 세트 및 버전을 연 다음, 로그 파일이 파일 또는 단위 테스트 모드로 실행하고 있었던 것처럼 로그 파일의 나머지를 처리한다. 재생하는 동안의 디스플레이는 아래와 같은 것을 제외하고는 파일 또는 단위 테스트를 실행한 후에 사용자가 보는 동일한 디스플레이이다: (1) 회귀 분석 데이터(regression data)에 대한 비교가 없고, (2) 셀 상태와 같은 일부 정보는 대개는 로그에서 빠질 것이므로 나타나지 않을 것이며, (3) 나타나 있는 규칙 세트의 버전이 현재의 버전이 아닐 수 있기 때문에, 규칙 세트는 재생하는 동안에 판독 전용이 될 것이다.

[0204] 로깅 기능이 인에이블될 때, 생성된 변환식은 logging 문(statement)을 포함하도록 수정된다. 로그 메시지는 임의의 스트링을 로그 포트에 출력하는 함수, write_to_log()을 호출함으로써 생성될 수 있다. write_to_log가 사용될 때, 특정한 포맷의 데이터가 로그 포트에 기록된다. 예를 들면, 로그 포트 포맷의 일 예는 다음과 같다:

[0205] record

[0206] string('|') node;

[0207] string('|') timestamp;

[0208] string('|') component;

[0209] string('|') subcomponent;

[0210] string('|') event_type;

[0211] string('|') event_text;

[0212] end;

[0213] 로깅 정보는 모두 write_to_log()의 호출 시에 명시된 스트링을 포함하는 event_text 필드에 저장되어 있다. 로그 포트 레코드의 다른 필드들은 구성요소에 의해 자동으로 채워지고, BRE에 의해 무시된다.

[0214] event_text 필드의 포맷의 일 예(로그 메시지에 한정됨)는 다음과 같다:

[0215] record

[0216] decimal('|') count; // 이벤트의 수

[0217] record

[0218] decimal('|') recNum; // 레코드 수, 1은 첫 번째 레코드, 0은 범례(legend)

[0219] string(1) opCode; // 아래 참조

[0220] decimal('|') rule set; // 어떤 포함되는 규칙 세트, 0은 주요 규칙 세트

[0221] decimal('|') group; // 어떤 입력 그룹, 어떤 규칙, 등

[0222] decimal('|') index; // 입력 인덱스, 출력 인덱스, 규칙 사례, 등

[0223] string(1) nullFlag; // 널용의 공백 또는 별표("-")

[0224] string('₩x01') value; // 입력 값, 출력 값, 등

[0225] end events[count];

[0226] end;

[0227] 다음은 opCodes의 일 예이다:

[0228] "I"는 입력 변수이다.

[0229] "O"는 출력 변수이다.

[0230] "K"는 룩업 키이다.

[0231] "L"은 룩업 변수이다.

[0232] "R"은 범례가 아닌(non-legend) 레코드에 대한, 트리거된 규칙 사례이다. 범례 레코드의 경우, 이것은 규칙이다.

[0233] "C"는 규칙 셀이며; 항상 사용되는 경우인 범례를 제외하고는, 테스팅 레벨 트레이싱이 인에이블된 때에만 사용된다.

[0234] "P"는 파라미터이다.

[0235] "H"는 칼럼 헤딩이다.

[0236] "N"은 규칙 세트 명칭이며; 범례 레코드들에서만 사용된다.

[0237] "V"는 버전 번호이며; 범례 레코드들에서만 사용된다.

[0238] "D"는 차이 레코드이며; 테스팅할 때만 사용된다.

[0239] 이 예에서는, 입력 레코드마다 하나의 로그 메시지가 있을 것이다. 로그 메시지는 이벤트의 어레이를 포함할 것이다. 입력 레코드마다 이벤트의 대규모 어레이를 만드는, 입력마다, 출력마다, 룩업 필드마다, 실행된 규칙 사례마다, 등 하나의 이벤트가 존재할 것이다.

[0240] 이 예에서는, 로그 이벤트 내의 값(value) 필드는 이진 형태가 아니다. 값 필드는 특수 문자 hex 01 또는 linefeed(라인피드)를 포함하지 않는데, 그 이유는 이들이 이벤트들 또는 로그 레코드들 사이의 구획 문자(delimiter)를 부정확하게 트리거할 것이기 때문이다. 대신, 모든 값이 프린트 가능한 스트링으로 변환된다. 프린트 불가능한 문자들은 16진 형태 스트링(예컨대, 라인피드 -> "₩x0a")으로 변환된다. 이진 변수들은 십진 형태 등으로 변환된다.

[0241] 변환식에 있어 임의의 최적화(내부의 룩업 파일을 사용하는 것과 같은)는 정확한 로깅을 보장하기 위해 필요에 따라 디스에이블될 수도 있다.

[0242] 첫 번째 레코드가 보일 때, 대응하는 규칙 세트의 세부사항을 포함하는 로그 메시지가 생성된다. 이것은 이른바 "범례" 로그 메시지라고 불리며 주어진 규칙 세트에 대한 그래프 실행마다 한번 생성된다. 범례 로그 메시지의 제1 부분은 로그 포맷 버전, 및 규칙 세트 위치와 버전 번호(재생에 요구됨)을 포함한다. 다음에 나오는 것은 각각의 입력, 출력, 규칙, 룩업 파일, 룩업 필드, 파라미터, 등의 명칭을 기록하는 정보일 것이다. 명칭들은 인덱스와 연관되어 있다(1은 첫 번째 입력 변수, 2는 두 번째 입력 변수, 등등). 이것은 로그 파일의 공간을 절약하기 위해, 이벤트들과 연관된 다음의 "트레이싱" 로그 메시지들이 명칭이 아니라 인덱스에 의해 변수들 및 규칙들을 참조할 수 있게 한다.

[0243] 범례 로그 메시지가로그 파일에 기록된 후, 로깅되는 그래프에 의해 처리된 데이터흐름의 첫 번째 및 모든 후속 레코드에 대해, 각각 대응하는 트레이싱 로그 메시지와 연관된, 다음의 로깅 이벤트들 중 어느 것이 발생할 수 있다.

[0244] (1) 입력 레코드들이 로깅되고 있는 경우, 로그 메시지는 각각의 입력 레코드에 대한 입력 변수들의 값을 문서화하는 입력 변수마다 생성된다.

[0245] (2) 파라미터들이 로깅되고 있는 경우, 로그 메시지는 파라미터들의 값을 문서화하는 입력 변수마다 생성된다.

[0246] (3) 포함된 규칙 세트가 있는 경우, 포함된 규칙 세트 내의 규칙들이 샐행된다. 그 규칙들은 이 논리에 따라 회귀적으로 로그 메시지들을 생성할 것이다.

[0247] (4) 사례 상태가 로깅되고 있는 경우, 모든 규칙 내의 사례마다 값이 계산되어 로깅된다.

[0248] (5) 실제 규칙 논리는 일련의 if-then-else 논리 또는 switch 문들 또는 내부 룩업들을 사용하여 실행된다. 어느 규칙 사례가 트리거되었는지가 알려지면 바로 트리거된 규칙 사례에 대한 로그 메시지가 생성된다.

[0249] (6) 또한 어느 규칙 사례가 트리거되었는지가 알려지면 바로, 값들이 규칙의 출력 변수들에 기록되는 동시에, 할당된 값을 문서화하는 각각의 출력(예컨대, 변환된 레코드 내의 값)에 대한 로그 메시지가 생성된다.

[0250] (7) 규칙들을 평가하는 동안에 수행된 룩업 참조들은 서브루틴에 의해 처리된다. 이 서브루틴 논리는 룩업을 수행한 후, 룩업 파일로부터 읽은 변수마다 키로서 사용된 값 및 발견된 모든 룩업 변수의 값을 문서화하는, 로그 메시지를 생성할 것이다. 그후, 서브루틴은 주 논리(main logic)에 룩업 값을 반환할 것이다. 룩업 서브루틴은 동일한 키에 대해 이중의 로그 이벤트 발생을 방지하기 위해 룩업 서브루틴이 이미 불려졌는지 여부를 나타내는 불 변수(Boolean)를 유지할 것이다.

[0251] (8) 트리거된 규칙 사례가 없는 경우, 마지막 else 절(clause)(또는 switch 문에 대한 디폴트 절)이 디폴트 출력 값들을 할당하기 위해 실행될 것이다. 동시에, 각각의 출력에 대해 할당된 값을 문서화하는 로그 메시지들이 생성될 것이다.

[0252] 로그 포트들로부터 제공된 로깅 정보는 다양한 방식으로 사용될 수 있다. 예를 들면, 어떤 경우에는, 테스트 데이터세트로 사용된 데이터세트와 연관된 입력 레코드들의 수는, 그래프 내의 데이터 처리 구성요소의 규칙 세트의 테스팅에 사용하는 테스트 데이터세트와 같은, 주어진 시나리오에 바람직한 것보다 많을 수 있다. 로그 포트들로부터의 로깅 정보는 그래프의 최초 실행 동안에 그래프의 구성요소들에 의해 처리될 수 있다. 이 구성요소들은 로깅 정보를 검사하여 규칙 세트의 규칙들 내의 규칙 사례 각각에 대해 적어도 하나의 트레이싱 로그 메시지를 제공하는 데 필요한 입력 레코드들의 최소 세트를 결정할 수 있다. 예를 들면, 구성요소들은 각 규칙 내의 각 사례가 트리거되는 제1 입력 레코드를 식별한다. 그후 식별된 레코드들은, 모든 다른 입력 레코드가 필터링된, 축소된 테스트 데이터세트와 연관되어어 저장된다. 이 축소된 테스트 데이터세트는 여전히 규칙 세트 내의 동일한 규칙들 및 규칙 사례들의 테스팅을 가능하게 하지만, 훨씬 작고 테스팅에 더욱 효율적으로 사용된다.

[0253] 중복 분석

[0254] 어떤 예에서는, 전술한 바와 같이, 규칙 세트에 대해 테스트 데이터세트를 실행할 때, 트리거된 모든 규칙, 즉 모든 입력 조건이 충족되고 또한 우선순위가 더 높은(higher-priority) 규칙 사례가 모든 입력 조건을 충족하지 않는 경우 출력이 생성되었을 것이 규칙 사례들이 추적될 수 있다. 테스트 데이터를 처리한 후, 시스템은 테스트 출력 데이터를 후처리(post-process)하여 어떤 테스트 사례에 의해서도 트리거되지 않았던 모든 규칙 또는 규칙 사례에 대한 리스트를 생성할 수 있다. 이 정보는 어떤 규칙들이 트리거되었는지 여부를 사용자에게 신속하게 보여주기 위해 편집기 내의 규칙들의 표시 상에 오버레이될 수 있다. 이 정보에 의거하여, 사용자는 다른 규칙들에 의해 모호질 수 있는 규칙들, 즉 중복하는 규칙들을 찾을 수 있다. 또한 카운트값(count)들도 각각의 규칙 사례에 대해 표시될 수 있다. 카운트값은, 특히 출력들의 원하는 분포를 얻기 위한 값들의 세부 조정, 및 성능을 조정하기 위해 가장 그럴듯한 규칙 사례들의 식별에 있어, 규칙 사례의 트리거 여부를 아는 것만큼 유용하다.

[0255] 변환식 생성

[0256] 응용예(그래프)에서 비지니스 규칙들은 각각의 규칙 세트를 변환식으로 변환함으로써 평가된다. 그후 변환식은 그래프의 구성요소에 부가된다. 그러한 구성요소는 변환식의 실행을 위해 특정한 방식으로 링크된 표준 구성요소의 세트를 포함하는 서브그래프일 수 있다. 이 그래프들은 그후 예를 들면 조인(join) 및 롤업(rollup)에 키들을 사용하기 위해, 추가의 구성요소들과 함께 사용될 수 있다.

[0257] 변환 코드(transform code)는 비지니스 규칙들로부터 여러 방식으로 생성될 수 있다. 변환식의 내부항목(internal)들이 사용자가 편집하도록(user-edited) 설계되지 않은 예에서, 생성 프로세스는 이해하기 어려운 변환식을 결과로 얻을 수 있지만, 변환식은 규칙들을 하나씩 적용하는 것보다 더욱 효율적으로 규칙을 실행한다. 어떤 경우에는, 생성된 변환식의 성능을 향상시키기 위해 특화된 룩업 파일들 또는 다른 테크놀로지가 사용될 수 있다. 변환식을 생성하는 방법에 대한 어떤 세부사항들은 디플로이먼트에 저장될 수 있다. 복수의 그래프에 사용될 수 있는 규칙 세트는 다른 가능한 사용자들을 위해 복수의 디플로이먼트를 가질 수 있다. 규칙 세트는 또한 변환식이 생성될 때 어떤 규칙들을 사용하는지를 식별하는 디플로이먼트와 함께, 그 일부만이 각각의 디플로이먼트에서 요구되는, 규칙들의 슈퍼 세트(super set)을 포함할 수 있다. 규칙이 (있다면) 약간의 표현식들과 함께, 많은 상수 값을 가지면, if then else 논리 대신에 룩업 테이블이 사용될 수 있다. 이 경우에, 룩업 테이블은 규칙의 일부분이다(별도로 저장되지 않음). 예를 들면, 이런 규칙을 생각해보자:

출발 도시	도착 도시	서비스 등급	표현식	마일리지(출력)
BOS	LAX	First		6000
BOS	LAX	Business		3000
BOS	LAX	Coach	Is Using Miles = yes	0
BOS	LAX	Coach	else	3000
BOS	CHI	First		2000
BOS	CHI	Coach		1000
BOS	NYC	First		1500
BOS	NYC	Business		1000
BOS	NYC	Coach		500

[0258]

[0259] 이 규칙은 아래의 정보를 가지는 메모리내(in-memory) 룩업 테이블을 작성함으로써 처리된다:

from	dest	class	expr	miles
BOS	LAX	1	0	6000
BOS	LAX	2	0	3000
BOS	LAX	3	1	3000
BOS	CHI	1	0	2000
BOS	CHI	2	0	1000
BOS	NYC	1	0	1500
BOS	NYC	2	0	1000
BOS	NYC	3	0	500

[0260]

[0261] 그러면 변환식은 이와 같은 것이 된다:

[0262] int expr = lookup(from, dest, class).expr

[0263] int miles = lookup(from, dest, class).miles

[0264] if(expr == 1 and is_using_miles) miles = 0

[0265] return miles;

[0266] 각 규칙 세트는 명칭을 해당 디플로이먼트의 설정에 관한 세부사항들에 매핑하도록 구성된 디플로이먼트 테이블을 가진다. 도 4를 참조하면, 특정한 실행 정보의 로깅을 원하는 사용자는 각 규칙 세트의 디플로이먼트 테이블에 정의된 속성들에 따라 그래픽 사용자 인터페이스에서 각각의 엔트리를 입력할 수 있다:

[0267] 1. 디플로이먼트 명칭. 규칙 세트 내에서 유일하여야 하는 임의의 스트링.

[0268] 2. 입력 데이터세트 명칭. 입력 변수 테이블 내에 열거된 입력 데이터세트가 복수개 있으면, 디플로이먼트 테이블 내의 각 엔트리는 어느 입력 데이터세트가 그 디플로이먼트에 사용되는지를 나타낸다.

[0269] 3. 출력 데이터세트 명칭. 출력 변수 테이블 내에 열거된 출력 데이터세트가 복수개 있으면, 디플로이먼트 테이블 내의 각 엔트리는 어느 출력 데이터세트가 그 디플로이먼트에 사용되는지를 나타낸다.

[0270] 4. 포함되는 규칙 세트 각각에 대한 디플로이먼트 명칭. 각각의 포함되는 규칙 세트에 대해, 포함하는 규칙 세트의 대응하는 디플로이먼트 각각에 어느 디플로이먼트가 사용되어야 하는지를 나타내어야 한다.

[0271] 5. 구성요소의 타켓 위치 및 생성될 변환식 파일.

[0272] 어떤 예에서는, 항상 적어도 하나의 디플로이먼트, 지명된 디폴트(named default)가 있다. 이것이 명시된 다른 디플로이먼트가 없을 때 사용되는 디플로이먼트이다.

[0273] 예시적인 일 실시예로, 다음이 변환식 생성에 대한 기본적인 사항들이다. 먼저, 규칙 세트에서 계산되는 출력들에 대한 규칙들이 생성된다. 모든 다른 출력은 변환식에서 와일드카드 규칙(wildcard rule)로 처리될 것이다. 일반적으로, 내부적으로만 사용되는 출력 변수들은 생성된 변환식 내에 지역 변수들이 생성되도록 한다. 그렇긴 하지만, 생성된 변환식은, 필요에 따라, 중복 계산을 피하기 위해 더 많은 지역 변수들을 포함할 수 있다(예를 들면, 최적화가 공간을 넘어 속도에 대한 것인 경우).

[0274] 변환식 타입에 의존하는 약간의 변환식 차이가 약간 있다:

[0275] ● 리포맷(reformat) - 입력은 "in0"이라고 불리며, 입력 변수들은 "in.field"와 같은 기술 명칭을 가질 수 있다. 출력은 "out"이라고 불리며, 출력 변수들은 "out.field"와 같은 기술 명칭을 가질 수 있다.

[0276] ● 조인(join) - 두 개의 입력은 "in0" 및 "in1"라고 불린다. 출력은 "out"이라고 불리며, 와일드카드 규칙은 in0가 "out"에 복사된다고 가정한다. 구성요소가 생성될 때 모든 파라미터가 설정될 수 있다. 규칙 세트는 조인(join)에 대한 입력마다 하나씩, 복수의 입력 세트를 가질 것이다. 규칙 세트는 또한 조인 타입(join-type), 입력들의 중복에 대해 제거(de-duped) 여부, 및 조인(join)에 키로서 사용된 필드들의 비지니스 명칭(어떤 예에서, 이것은 각 입력 세트에 존재하여야 함)을 지정할 것이다. 또한, 사용자는 입력 필터로서 사용되는 입력 각각에 대한 표현식을 지정하도록 허용될 수 있다.

[0277] ● 롤업(rollup) - 입력은 "in0"으로 불리고 출력은 "out"으로 불린다. 롤업 타입(rollup-type) 타입 규칙 세트의 경우, 사용자가 집계 함수(aggregation function)를 사용하는 것이 허용된다(다른 변환식 타입에서는 지원되지 않음). 사용자가 기술 명칭이 "input_select" 또는 "output_select"인 출력 변수를 생성하면, "input_select" 및/또는 "output_select" 함수가 그 출력들을 계산하는 규칙들의 논리와 함께 변환식에 추가된다. 이 두 함수 모두의 입력은 "in0"로 불린다(output_select가 보통 자신의 파라미터를 "out"으로 명명할지라도).

[0278] ● 필터(filter) - 두 개의 미리 정해진 상수 중 하나가 출력이다. 필터 타입(filter-type) 변환식의 유일한 출력 변수는, 출력을 전달하기 위해 영이 아니고(non-zero) 또한 널이 아닌(non-NULL), 타입 구성요소의 "select"이다.

[0279] 추가적인 변환식 타입들도 구현될 수 있다:

[0280] ● 스캔(scan) - 스캔 타입(scan-type) 규칙 세트들의 경우, 사용자가 레코드들 간에 기억되는 하나

이상의 출력에 대한 값들을 지정할 수 있다. 이 출력들에 대한 값은 보통 레코드마다 계산될 것이다. 그러나, 출력 각각에 대해 마지막 레코드로부터 출력들의 값을 포함하는 추가의 내장(bult-in) 입력들이 생성될 것이다. 이것은 사용자가 예를 들면 상태가 그때 다음 레코드의 입력으로서 이용 가능한 출력 변수에 부분 합계를 저장함으로써, 복수의 레코드에 걸쳐 필드의 합계를 계산할 수 있도록 해준다.

[0281] 또, 스캔 타입 규칙 세트들의 경우, 사용자가 선택적인 키를 명시할 수 있다. 이 키는 그룹 레코드들에 사용되는 하나 이상의 필드이다. 스캔 타입 규칙 세트에 키가 지정되는 경우, 레코들 간에 기억된 모든 출력의 상태는 각각의 유일한 키 값에 대해 다를 것이다. 예를 들면, 키가 고객 번호이고, 하나의 출력이 고객 각각에 대한 모든 트랜잭션의 합계를 계산하는 데 사용되는 경우, 드러나지 않게, 합계의 일 부분이 유일한 고객 번호마다 저장될 것이므로, 각 고객에 대해 상이한 합계가 계산될 수 있다.

[0282] ● 분류(classification) - 규칙은 N개 출력을 가지고 변환식은 각각의 레코드에 대해 어떤 출력이 사용되는지를 결정한다. 이 구성요소의 경우, 시스템이 out::classify(in) 함수를 생성한다. 출력은 어느 출력 포트(하나 이상 있을 수 있음)이 사용되어야 하는지를 나타내는 단일 정수(single integer) 값이다. 출력 0은 0번째 포트, 출력 1은 첫 번째 포트, 등을 의미한다. Classification 타입 변환식의 유일한 출력 변수는 출력 포트(제로 베이스, zero-based)의 인덱스가 되는, 타입 구성요소의 "select"이다. 이것은 출력에 두 개가 아닌 N개 값이 사용되는 것을 제외하고는 filter(필터)와 유사하다.

[0283] ● 함수(function) - 함수 타입(function-type) 규칙 세트는 변환식 파일로 바뀔 수 있지만, 구성요소 변환식으로서는 아니다. 대신, 함수 타입 규칙 세트가 변환식으로 바뀔 때, 구성된 변환식 파일이 다른 변환식들에 포함되도록 설계된다. 각 출력 변수는 함수로 바뀐다. 그 함수들에 대한 입력들은 규칙의 타입에 의존한다. 함수 타입 규칙 세트가 변환식으로 바뀔 때, 구성된 변환식 파일이 다른 변환식들에 포함되도록 설계된다. 각출력 변수는 함수로 바뀐다. 그 함수들에 대한 입력들은 규칙의 타입에 의존한다. 함수 타입 규칙의 경우, 입력들은 테이블에 열거된 순서대로의 규칙에 대한 입력들이다. 비함수(non-function) 규칙들의 경우, 각 출력 함수는 입력 변수들에 대응하는 모든 필드를 가지는 레코드로 지명된, 단일 입력을 취한다.

[0284] 규칙 세트가 다시 그래프의 일부가 될 변환식을 생성하기 위해 사용될 때, 그래프 구성요소는 규칙 세트 및 디플로이먼트의 명칭을 포함한다. 그래프 개발자는 그후 생성된 변환식 대신에 구성요소에서 규칙 세트를 편집할 수 있다. 규칙 세트에 대한 변경사항들은 변환식이 다시 생성되게 한다. 어떤 예에서, 사용자는 BRE에 의해 생성되었던 구성요소 상에서 시프트 키를 누른채 더블 클릭(shift-double click, 이하 시프트 더블 클릭이라 함)할 수 있다. 이 시프트 더블 클릭은 GDE(Graphical Develop Environment)로 하여금 EME 명칭, 규칙 세트 명칭 및 디플로이먼트에서는 통과하는, BRE를 시작하게 한다. 일 예에서는, 커맨드 라인 인터페이스를 각 BRE 프로세스를 시작하기 위해 사용할 수 있지만; 다른 프로세스간(inter-process) 통신 메커니즘을 채용할 수 도 있다.

[0285] 전술한 로깅 접근법은 컴퓨터 시스템에서 수행하기 위한 소프트웨어를 사용하여 구현될 수 있다. 예를 들면, 이 소프트웨어는 각각이 적어도 하나의 프로세서, 적어도 하나의 데이터 스토리지 시스템(휘발성 및 비휘발성 메모리 및/또는 스토리지 요소들 포함), 적어도 하나의 입력 디바이스나 포트, 및 적어도 하나의 출력 디바이스나 포트를 포함하는 하나 이상의 프로그램된 또는 프로그램 가능한 컴퓨터 시스템들(분산형, 클라이언트/서버, 또는 그리드와 같은 다양한 아키텍처로 된 것일 수 있음)에서 실행하는 하나 이상의 컴퓨터 프로그램 내의 프로시저(procedure)들을 구성한다. 소프트웨어는 예를 들면 계산 그래프의 설계 및 배치에 관련된 다른 서비스를 제공하는 규모가 더 큰 프로그램의 하나 이상의 모듈을 구성할 수 있다. 그래프의 노드들 및 요소들은 컴퓨터로 판독 가능한 매체에 저장된 데이터 구조체들 또는 데이터 저장소에 저장된 데이터 모델을 따르는 다른 구조화된 데이터로서 구현될 수 있다.

[0286] 소프트웨어는 CD-ROM과 같이 범용 또는 전용(general or special purpos)의 프로그램 가능한 컴퓨터로 판독 가능한 저장 매체 상에 제공되거나, 또는 네트워크의 통신 매체를 통해 실행되는 곳인 컴퓨터에 배포(전파 신호로 인코딩됨)될 수 있다. 모든 함수는 전용 컴퓨터에서, 또는 코프로세서와 같은 전용하드웨어를 사용하여 수행될 수 있다. 소프트웨어는 분산 방식으로 구현되어, 소프트웨어 의해 지정된 계산의 다른 부분들이 다른 컴퓨터들에서 수행될 수 있다. 그러한 컴퓨터 프로그램 각각은 본 명세서에서 설명한 프로시저들을 수행하기 위해 저장 매체 또는 디바이스가 컴퓨터 시스템에 의해 판독될 때 컴퓨터를 설정 및 운영하기 위해, 범용 또는 전용의 프로그램 가능한 컴퓨터에 의해 판독 가능한 저장 매체 또는 디바이스(예컨대, 고체 메모리 또는 매체, 또는 자기 또는 광 매체)에 저장되거나 다운로드되는 것이 바람직하다. 본 발명의 시스템은 또한, 저장 매체가 컴퓨터 시스템을 본 명세서에서 설명한 기능들을 수행하도록 특정의 미리 정해진 방식으로 동작시키도록 구성되는

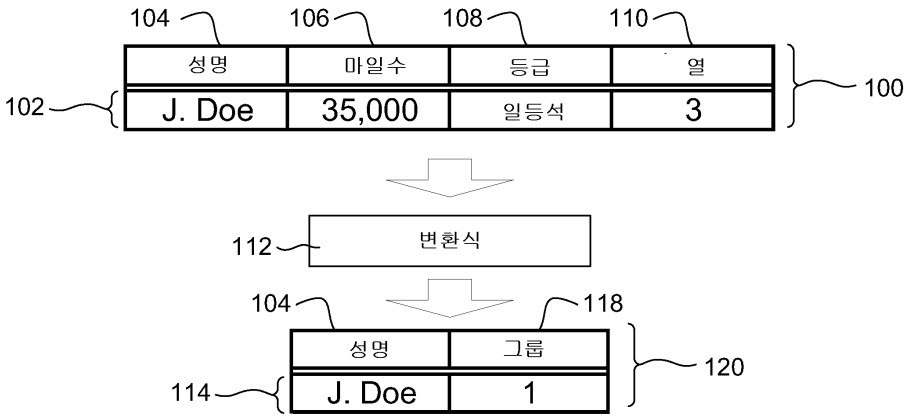
경우에, 컴퓨터 프로그램으로 구성된, 컴퓨터로 판독 가능한 저장 매체로 구현되는 것으로 생각될 수 있다.

[0287] 본 발명에 대한 많은 실시예를 설명하였다. 하지만, 본 발명의 사상 및 범위를 벗어나지 않은 다양한 변경이 가능하다는 것을 알 것이다. 예를 들면, 전술한 단계들 중 어떤 것은 순서와 무관할 수 있으며, 따라서 설명한 것과는 다른 순서로 실행될 수 있다.

[0288] 이상의 설명는 예시를 위한 것으로 첨부된 특허청구범위에 의해 정해지는 본 발명의 범위를 한정하려는 것이 아님을 알아야 한다. 예를 들면, 전술한 많은 기능 단계들은 전체 처리에 실질적으로 영향을 미치지 않고 다른 순서로 수행될 수 있다. 다른 실시예들은 이하의 청구항들의 범위 내이다.

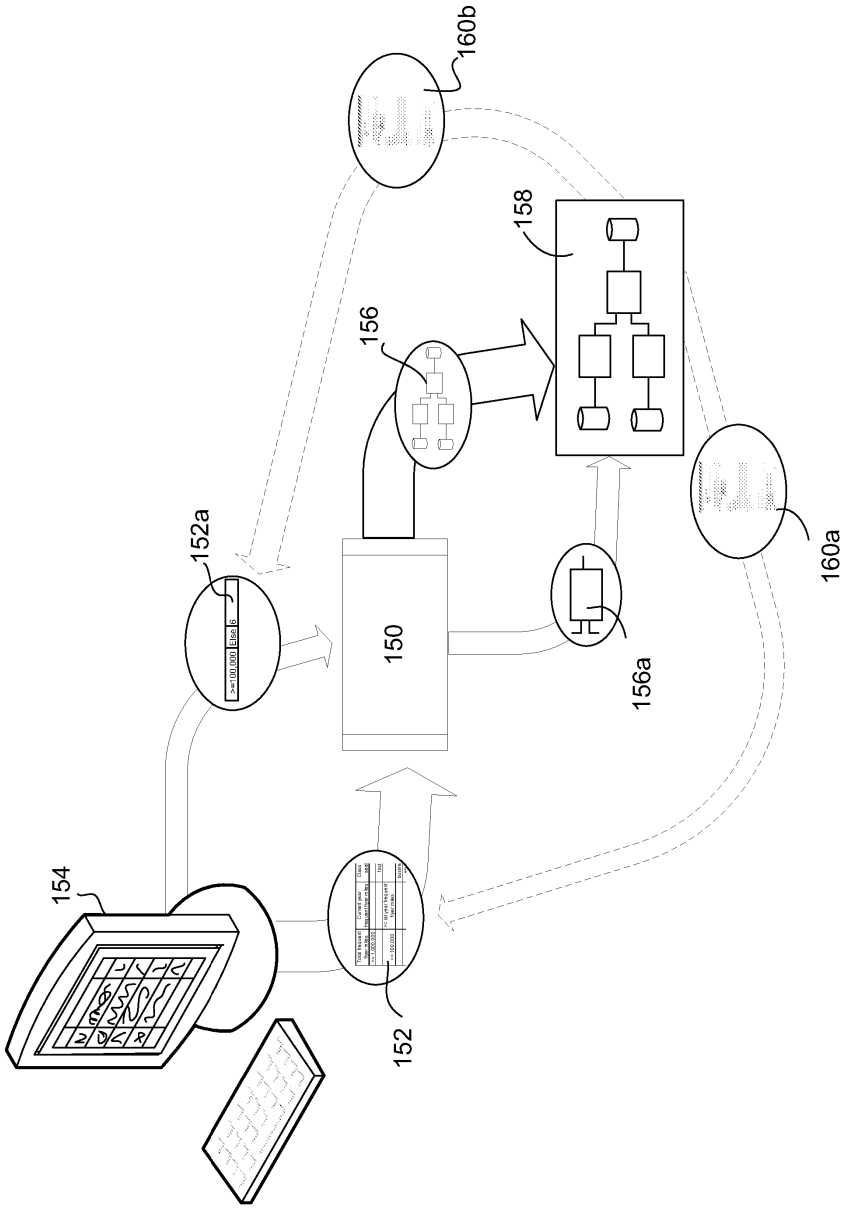
도면

도면1a



도면1b

도면1c

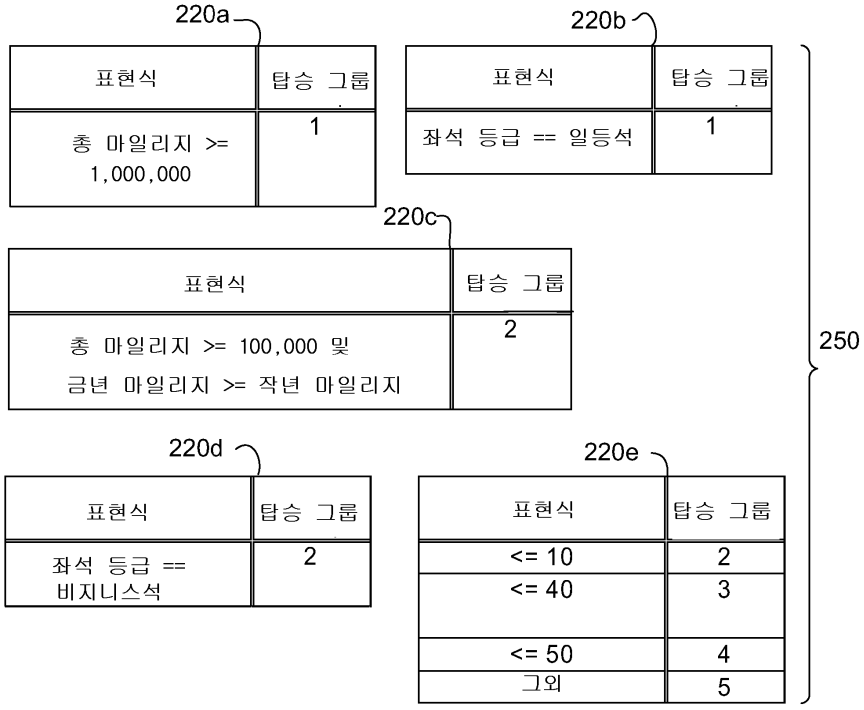


도면2a

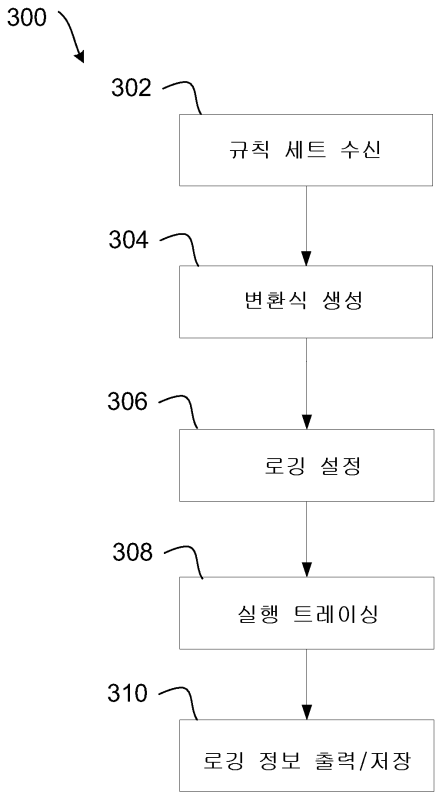
	202 총 마일리지	204 금년 마일리지	206 좌석 등급	208 좌석열	212 탑승 그룹
210a	>= 1,000,000				1
210b			일등석		1
210c	>= 100,000	>= 작년 마일리지			2
210d			비지니스석		2
210e			그외	<= 10	2
210f			동일	<= 40	3
210g			동일	<= 50	4
210h			동일	그외	5

200

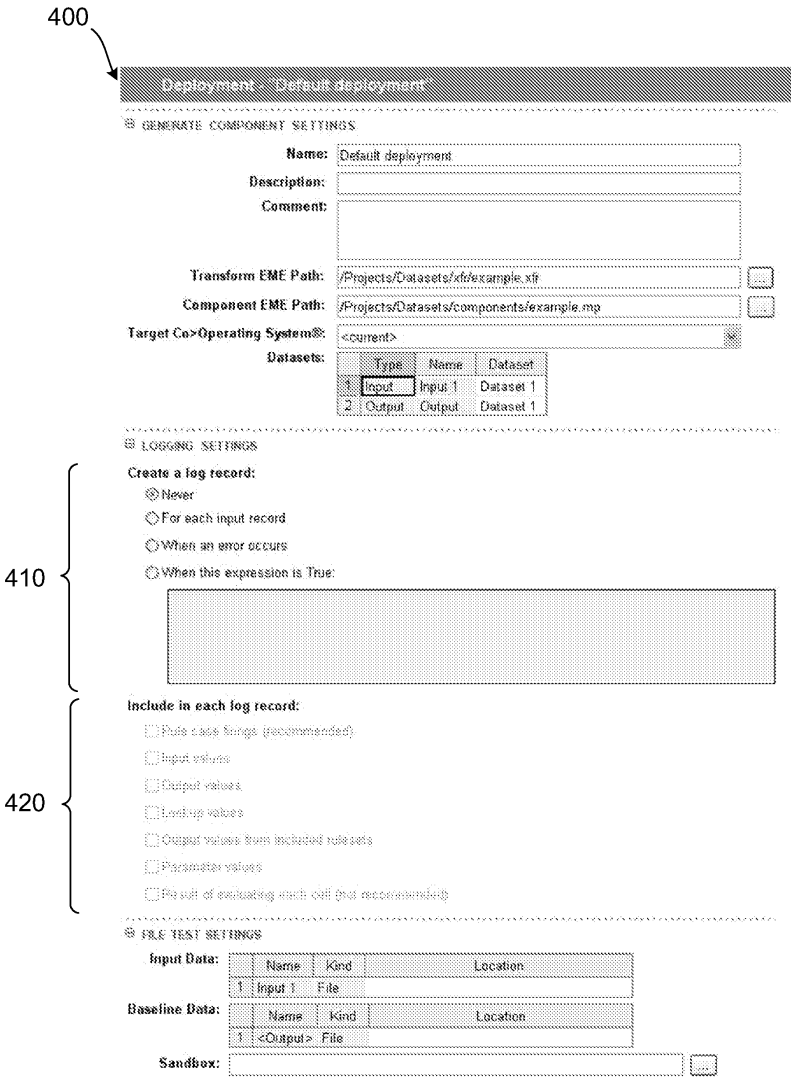
도면2b


220a
표현식
탑승 그룹
총 마일리지 >= 1,000,000
1
220b
표현식
탑승 그룹
좌석 등급 == 일등석
1
220c
표현식
탑승 그룹
총 마일리지 >= 100,000 및 금년 마일리지 >= 작년 마일리지
2
220d
표현식
탑승 그룹
좌석 등급 == 비지니스석
2
220e
표현식
탑승 그룹
<= 10
2
<= 40
3
<= 50
4
그외
5
250

도면3


300
302
규칙 세트 수신
304
변환식 생성
306
로깅 설정
308
실행 트레이싱
310
로깅 정보 출력/저장

도면4


400
Deployment - Default deployment
GENERATE COMPONENT SETTINGS
Name:
Default deployment
Description:
Comment:
Transform EME Path:
/Projects/Datasets/xfr/example.xfr
Component EME Path:
/Projects/Datasets/components/example.mp
Target Co>Operating System®:
<current>
Datasets:
Type
Name
Dataset
1 Input Input 1 Dataset 1
2 Output Output Dataset 1
LOGGING SETTINGS
Create a log record:
Never
For each input record
When an error occurs
When this expression is True:
410
Include in each log record:
Rule case hings (recommended)
Input values
Output values
Lookup values
Output values from included rulesets
Parameter values
Result of evaluating each cell (not recommended)
420
FILE TEST SETTINGS
Input Data:
Name
Kind
Location
1 Input 1 File
Baseline Data:
Name
Kind
Location
1 <Output> File
Sandbox: