疎結合な関係にある企業間のトレーサビリティシステムの提案

百合山 まどか　小金山 美賀　渡邊 裕治

北山 文彦　沼尾 雅之

日本アイ・ビー・エム（株） 東京基礎研究所

〒242-8502 神奈川県大和市下鶴間 1623-14

E-mail: {yuriyama, mkogane, muew, ktym, numao}@jp.ibm.com

あらまし グローバルに展開される製造や輸送において、個々の製品および部品の品質を管理するためにトレーサビリティを実現することが求められている。ある製品の情報を追跡するためには、その製品に関する部品・原料等を扱う複数の企業から情報を収集する必要がある。本稿では、企業間の関係は動的に変更される疎結合なものであるため、企業システムからデータを収集する手段として Web サービスを採用し、トレーサビリティのための標準的な 2 つのインターフェースを定義した。また(1)企業システムのデータを管理するデータソース管理者、(2)製造や輸送等のビジネスプロセスを定義するプロセス定義者、(3)トレーサビリティシステムを利用して情報を検索するアプリケーションを設計する検索要求設計者、(4)アプリケーションを通して検索を実行する検索要求発行者というアクターの役割に応じて、各アクターがそれぞれ持ちうる知識も異なるため、本稿で提案するシステムでは、役割に応じて分割した定義を利用する。

キーワード トレーサビリティ、Web サービス、情報統合、情報検索

A Proposal of Traceability System over Loosely-Coupled Companies

Madoka Yuriyama　Mika Koganeyama　Yuji Watanabe

Fumihiko Kitayama　Masayuki Numao

Tokyo Research Laboratory,　IBM Japan, Ltd.

1623-14, Shimotsuruma, Yamato-shi, Kanagawa 242-8502, Japan

E-mail: {yuriyama, mkogane, muew, ktym, numao}@jp.ibm.com

Abstract Traceability is required for the quality management of each product and parts in global manufacturing and distributors. It is necessary to gather information about parts or materials from multiple companies in order to trace information about a product. Our Traceability system connects to the companies' system via Web Service because the relationship among companies is changed dynamically and loosely-coupled. We defined the two interfaces for Traceability. There are four actors: (1) data source administrator to manage data of companies' system, (2) process definition to define business process, (3) query designer to design application using Traceability system, and (4) query requester to get information by the application. We divided definitions for Traceability System because each actor has different knowledge.

Keyword Traceability, Web Service, Information Integration, Information Retrieval

1. はじめに

グローバル化が加速するビジネス環境の中で、世界にある経営資源を効率よく活用し、最適なところで調達・生産し、市場に届けることが重要となりつつある。それに伴い、グローバルに展開される製造や輸送において、個々の製品および部品の品質を管理するためにトレーサビリティを実現することが求められている。特に、不具合品の早期発見や特定に対応するために、製造業は最終製品、いわゆる完成品だけでなく製品を構成する部品に関しても履歴を辿ることが必要となってきている。トレーサビリティの必要性は製造業や輸送業のみならず、食品、医薬品、個人情報等、多種に渡って求められているが、本稿では主に製造業のトレーサビリティに焦点をあてている。

1.1. トレーサビリティとは

トレーサビリティとは、ISO9000：2000 年度版[1]において、「考慮の対象となっているものの履歴､適用又は所在を追跡できること」と定義されている。さらに「製品のトレーサビリティは材料および部品の源、処理の履歴、出荷後の製品の配送および所在の各項に関連する」と記述されている。本稿における「トレーサビリティ」は主に「製品のトレーサビリティ」を指し、「(最終製品だけでない）製品の材料および部品の源、処理の履歴を追跡できること」を意味する。例えば、ある製品がどの部品を使用して、誰に生産され、どのような流通過程でどのような業者に扱われたかなどといった情報（Supply Chain Visibility）がその好例であろう。

EPC-IS[2][3]に代表される RFID(Radio Frequency Identification)によるトレーサビリティも注目されているが、本稿で扱うトレーサビリティ対象の製品や部品は、その全てに RFID が付けられていることを仮定しない。そのため、たとえば製品から、それを構成する部品の情報を得るためには、製品と部品の組み付け履歴データを取得する必要がある。組み付け履歴データとは、部品 A と部品 B を組みつけて、製品 C を作成した等という情報を含むデータを指す。このように本稿では、RFID さえ読み取れば、関連する全ての情報が取得できるような状況を仮定せず、データを紐付けながら必要な情報を取得していくことでトレーサビリティを実現する。

1.2. トレーサビリティを実現する上での課題

トレーサビリティを実現するとは、ビジネスプロセスの実行に伴い生成されるデータを紐付けすることにより、データを統合し統一的なビューを提供することと考えられる。トレーサビリティに関するデータは、その対象のライフサイクルに応じて様々なデータソース中に蓄積されるため、分散したデータソースの統合が不可欠である。そのためには以下にあげる潜在的な課題がある。

1 複数のデータソースにまたがったデータの間の関連を解決する必要がある。例えば、ある製品を構成する部品情報のトレーサビリティにおいては、組み付け履歴データベースにある組み付けデータと部品情報データベースにある部品の詳細データ（部品 A は 2005/3/21 に工場 X から納入された等）は、異なるデータソースにまたがっている可能性がある。こういったデータソース間関連は、今日のグローバルに展開される製造業においては、単一企業・単一アプリケーション内でのデータソースの違いにとどまらず、複数企業・複数アプリケーションにわたる可能性が高い。

2 データソース毎に異なる表現を統合する必要がある。例えば、修理作業履歴は修理を担当する企業毎に異なる表現を用いている可能性がある。その一方で、修理履歴としてデータを統合するには、各企業の表現の違いを吸収して統合する必要がある。

3 データソース間の関連は時間に応じて変化する可能性がある。データソース間の関連は、データソースにデータを登録する主体（企業・アプリケーション・人）がどういうビジネスプロセスに沿って行動しているかに依存する。ところが、ビジネスプロセスは時間に応じて動的に変化するため、データソース間関係もそれに伴い柔軟に変更できる必要がある。こういった場合、複数データソースから必要なデータを取得するために検索処理を設計すること自体が極めて煩雑な作業となる。

本稿では上記の課題を解決するために、分散したデータソースに保持されるデータに対して、統一的に検索手段を実現する基盤を提供するシステムを提案する。以下、2 章ではトレーサビリティのためのデータを提供するデータソースについて、3 章ではトレーサビリティシステムのアクターについて、4 章では本システムにおけるモデル定義について、5 章では実装したシステム構成について記述し、6 章でまとめる。

2. トレーサビリティのためのデータソース

トレーサビリティシステムは追跡対象およびそれに関連する個体のデータを、それらを取り扱ったデータソースに問い合わせをして収集する必要がある。一方、問い合わせを受けるデータソースはトレーサビリティシステムから問い合わせがくると、それに応じてデータを返す。

2.1. Web サービス経由のデータ収集

単一企業・単一アプリケーション内でのトレーサビリティの場合は、トレーサビリティシステムとデータソースは密接な関係にあるので、データベースやファイルシステムに直接アクセスしてデータを収集することも可能だろう。しかし本稿がターゲットとするようなグローバルに展開される製造や輸送におけるデータは複数企業・複数アプリケーションで取り扱われてい

るケースが多く、それらの企業間の関係は疎な関係である。取引状況や契約状況により、企業間の関係は動的に変化するため、データソース間を密に結合させるのは難しい。そこで本稿ではトレーサビリティシステムとデータソースの間のコネクションに Web サービスを採用した。

2.2. Web サービスのインターフェース

トレーサビリティシステムがデータソースに対する問い合わせで取得したいデータは、当然問い合わせ条件に合致するデータ一覧であるが、常に可能な限りの全ての情報を返すことはパフォーマンスの悪化につながるため、望ましくない。取得目的に応じて返答の構造を定義する必要がある。トレーサビリティシステムがデータソースに問い合わせをする目的は主に 2 つに分類できる。

- データソースで管理している詳細情報の取得すること。
  例えば、修理作業履歴を保持しているデータソースに対して、車の ID を入力して、その修理作業の詳細データを取得するなど。

- 次のデータソースへの問い合わせ条件に使用できるデータを取得すること。
  例えば、エンジン工場の出荷履歴を保持しているデータソースに対して、エンジンの型番と出荷年月を入力して、該当するエンジンの ID 一覧を取得するなど。

前者の場合は、詳細データを返答として返すため、柔軟なデータ構造が望ましいと考えて、本稿では XML 文書を返答の形式として選択した。一方、後者の場合は、必要なデータセットは限定されるため、データ構造の柔軟性よりも処理の高速化を優先して、テーブル構造を返答の形式とした。本稿で提案するトレーサビリティシステムでは、以下の 2 つのインターフェースをデータソースは提供し、システムは取得目的に応じたインターフェースの Web サービスを呼び出す。

- getRecordObjects インターフェース
  トレーサビリティシステムから入力される検索条件（文字列で指定）に対して、該当するデータを XML 文書で返す。返答の XML 文書の構造（スキーマ）はあらかじめ定義しておき、トレーサビリティシステムに登録する必要がある。

- findRecords インターフェース
  トレーサビリティシステムから入力される検索条件（文字列で指定）に対して、該当するデータを、テーブル構造で返す。テーブル定義はデータソース固有のため、トレーサビリティシステムに登録する。

3. トレーサビリティシステムのアクター

本章では、トレーサビリティシステムのアクターを持ちうる知識により分類する。

3.1. 検索処理要求の概要

本稿で提案するトレーサビリティシステムにおける検索処理要求の大まかな流れを図 1に示す。トレーサビリティシステムに対する検索要求はクエリ設計とクエリ発行の 2 段階で行われる。まず、どういった形式の検索要求を行うかを設計するアクター（検索要求設計者）がクエリの設計情報をトレーサビリティシステムに入力し、それに基づくクエリ手段を実際に検索要求の発行主体（検索要求発行者）に提示する(1)。ここで提示されたクエリ手段に基づくクエリ要求が実際にトレーサビリティシステムに入力されると(2)、システムは分散したデータソースから必要なWebサービスを呼び出してデータを取得し、複数のデータソースから収集したデータを統合し(3)、結果を応答する(4)。

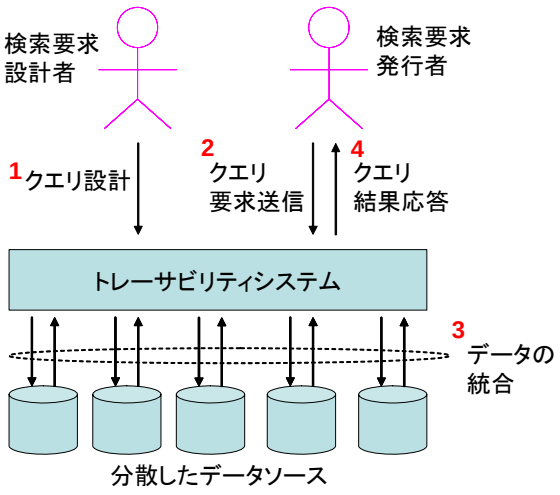

図 1　検索要求処理の流れ

3.2. アクターの分類

トレーサビリティシステムにおけるアクターを持ちうる知識により分類する。

1　データソース管理者
トレーサビリティシステムが接続対象とするデータソースを管理するアクターである。デ

ータソース管理者は、データソースの構成情報やデータソースがどのような問い合わせを受け付けられるかなどの知識を有する。

2　プロセス定義者

ビジネスプロセスがどのようなフローで処理されるか知っているアクターである。またビジネスプロセスの実行に伴いどんな情報が生成されるのか、また各データソースに保持されているデータをビジネスの形態に併せてどう統合するのが妥当か、といったビジネスモデルの設計も行える。

3　検索要求設計者

トレーサビリティシステムに対する検索要求を設計するアクターである。例えば、トレーサビリティシステムを利用して情報を検索するアプリケーションを設計するアプリ開発者などが該当する。クエリ設計は、プロセス定義者が定義したビジネスモデルに基づいて設計される。クエリパラメータとして何を入力し、検索結果としてシステムからどのような情報が出力されるべきかを知っている。

4　検索要求発行者

アプリケーションなどを通じて、トレーサビリティシステムに対する検索要求を発行するアクターである。検索要求の発行は、検索要求設計者の定義したクエリ設計に基づき行われる。検索要求発行者は、クエリ設計に対応したクエリパラメータを含むクエリ要求をシステムに入力することで必要な情報を取得する。

なお各アクターは必ずしも物理的に別の主体であることを意味しない。即ち、検索要求の設計者は、ビジネスモデルの知識もあり、プロセス定義者である場合もあるし、検索要求の発行者にもなりうる。またデータソース管理者がプロセス定義者として、システムにプロセスやデータモデルに関する定義を登録する場合もある。

3.3. アクターとデータ定義の関係

疎結合な関係にある企業間のトレーサビリティを実現するためには、異なる役割のアクターが独立に、かつ柔軟に必要な定義を設定し、運用できる必要がある。定義の詳細については、4 章に記述するが、ここではアクターとデータ定義の関係の概略図である図 2 を参照しながら簡単に説明する。

- データソース管理者は自らが管理するデータソース（a1, a2, b, c1,…）の定義をシステムに登録する。

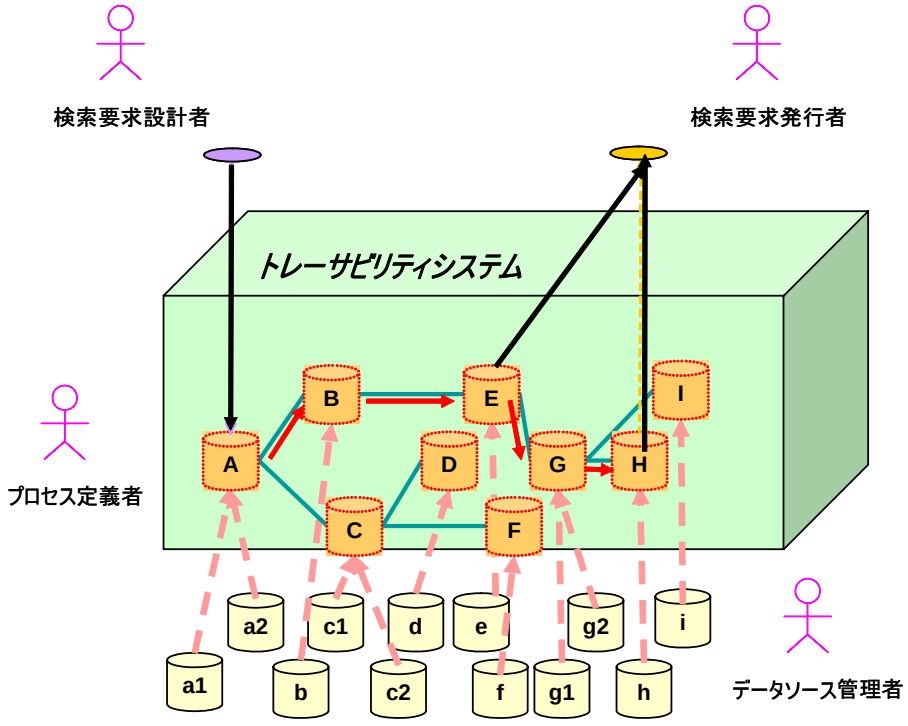

図 2　アクターとデータ定義の関係

・ プロセス定義者は以下の定義を設定する。
  - ビジネスプロセスにおける標準データモデル(A, B, C, …)
  - プロセスのフローの定義(A と B、A と C、B と E、…を結ぶ線)
  - データソースのデータと標準的なデータモデルとの変換定義(a1 と A、a2 と A、b と B、…を結ぶ点線)

・ 検索要求設計者はプロセス定義者が定義した標準データモデルやフローの定義から使用したいもの(A, B, E, G, H)を選択し、クエリパラメータ(A の中の要素 X)、クエリの出力(E の中の要素 Y、H の中の要素 Z)を定義してクエリ設計を行う。

・ 検索要求発行者はクエリ設計に対応したクエリパラメータ(A の要素 X = “xxx”)を入力して、出力(E の要素 Y = “yyy”、H の要素 Z = “zzz”)を得る。

各アクターは自らの持ちうる知識の範囲内で必要な定義をすることができ、責任範囲外で変更が生じても、多大な影響を受けない。例えば、標準データモデル B に対応する新しいデータソース b’がデータソース管理者により登録されてもフローの定義やクエリ設定には影響が及ばない場合が多い。

4. データモデル

本章では、トレーサビリティシステムのデータモデルについて述べる。トレーサビリティシステムがクエリ要求に応じてクエリ結果を返すために必要な定義を誰(アクター)がどのような定義を設定するかについて記述する。

4.1. データソースの定義

トレーサビリティシステムが統合を行う対象のデータソースが保持するデータの統合単位を「レコードオブジェクト」と呼ぶ。データソースが組み付け履歴 DB の場合の、組み付け履歴レコードがレコードオブジェクトに相当する。

データソースおよびレコードオブジェクトに次のような前提を置く。

1 トレーサビリティシステムに接続する全てのデータソースは互いに異なる識別子(データソース ID)により識別可能であるものとする。

2 レコードオブジェクトはデータソースが提供するデータ内の主個体単位のデータで XML 文書形式である。その構造はスキーマ(レコードスキーマ)で定義される。

例えば、車の組み付け履歴データベースの場合は、車の ID 単位でレコードオブジェクトを構成し、修理作業履歴を保持しているデータソースの場合は修理履歴 ID 単位もしくは車の ID 単位でレコードオブジェクトを構成する。

3 データソースは 1 個の getRecordObjects インターフェースの Web サービスと 0 個以上の findRecords インターフェースの Web サービスを提供する。getRecordObjects インターフェースの Web サービスが返す XML 文書はレコードオブジェクトの集合を含むものである。

企業システムのデータソースに関する知識があるデータソース管理者は、トレーサビリティシステムに以下のような情報を含むデータソースの定義を登録する。

・ データソース ID
・ レコードスキーマ
・ getRecordObjects インターフェースの Web サービス
  - 入力として受け付ける検索条件
  - 出力の XML 文書の構造(スキーマ)
  - Web サービスの接続情報(WSDL)

・ findRecords インターフェースの各 Web サービス
  - 入力として受け付ける検索条件
  - 出力のテーブル構造
  - Web サービスの接続情報(WSDL)

4.2. 標準データモデルおよび変換の定義

データソースが出力するレコードオブジェクトはレコードスキーマをスキーマ定義とする。このレコードスキーマはデータソースが独立に定義するものである。例えば、A 社のエンジン組み立て履歴を示すレコードオブジェクトは、A 社が独自に定義したレコードスキーマを持つと考えられる。一方で、一般的な「エンジンの組み立て履歴」のデータ形式があるとすれば、A 社の定義するレコードスキーマから標準的なデータ形式への変換が定義される必要がある。こういったレコードスキーマでスキーマ定義されるレコードオブジェクトを別のデータ形式で表現する場合に、この変換

先のデータ形式のことをレコードビューと呼ぶ。またレコードビューへの変換自体をレコード変換定義と呼ぶ。レコード変換定義はレコードオブジェクトをどのようにして変換してレコードビューのデータ形式に変換するかを定義する。レコード変換定義は単一のレコードオブジェクトだけでなく複数のレコードオブジェクトからの変換や多段のレコード変換定義を記述することもできる。

例を用いて説明する。図 3では、エンジンに関する情報を持つ二つのデータソース（ID を RDS_E1 と RDS_E2 とする）が存在する。RDS_E1 は（エンジンID、出荷日）として、また RDS_E2 は（EngineID、発送日）として、エンジンに関するレコードを異なる表現を用いて格納している。各レコードはトレーサビリティシステムへデータを出力する際にはレコードオブジェクトとして各レコードスキーマに則った形で出力される。すなわち、RDS_E1 の出力するレコードオブジェクトは SCH_E1 というレコードスキーマにより型付けされている。同様に、RDS_E2 の出力するレコードオブジェクトも SCH_E2 というレコードスキーマにより型付けされている。

次にレコードビューとレコード変換定義についてみることにする。RDS_E1 と RDS_E2 はそれぞれ独立のレコードスキーマ：SCH_E1 と SCH_E2 を用いてレコードオブジェクトを表現している。一方、SCH_E1 も SCH_E2 も本質的には同じ内容を表現していることから、これを統合したデータ形式 SCH_E として表現したい。SCH_E1 で型付けされたレコードオブジェクトを SCH_E で型付けされたレコードビューオブジェクトに変換するための変換ルールがレコードビュー変換定義 Tr_E1_to_E に記述される。同様に、SCH_E2 で型付けされたレコードオブジェクトからの変換は Tr_E2_to_E に記述される。なおレコード変換定義で変換されてレコードビューで型付けされたデータオブジェクトをレコードビューオブジェクトと呼ぶ。レコードオブジェクトからレコードビューオブジェクトへの変換だけではなく、レコードビューオブジェクトからの変換や、複数のレコードビューオブジェクトから単一のレコードビューオブジェクトへの変換などもレコード変換定義に記述することができる。

これらの定義はビジネスモデルやビジネスプロセスの知識を持つプロセス定義者が行う。ビジネス定義者が独自の定義を行ってもよいし、業界の標準データモデルなどが存在する場合はそれをレコードビューの定義として用いることも考えられる。

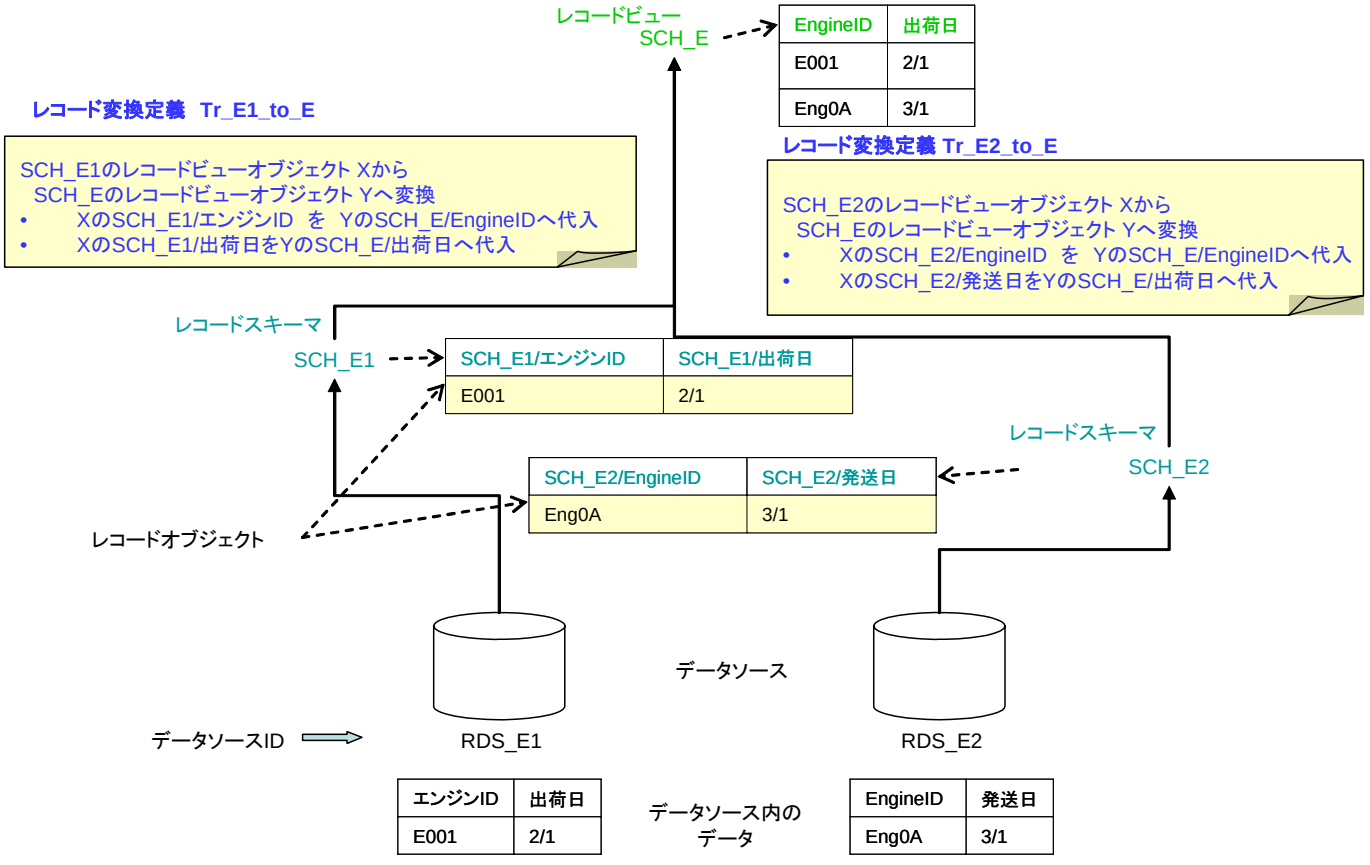

図 3　レコードスキーマ、レコードビュー、レコード変換定義の例

4.3. データ関連の定義

トレーサビリティでは、異なるデータソースに存在する複数のレコードオブジェクトを特定の関連に基づいて統合する処理が頻出する。例えば、組み付け情報のトレーサビリティの場合、最終製品の組み付け情報と、部品の製造情報を統合する必要があるが、これら二つの情報をどうやって統合するかは、各情報がどういった構造しているか、あるいはデータソース間に部品納入→製品組み立てという関係があるか、といった情報に基づいて決定される。こういったデータソースをまたがる関連の定義を「レコードリレーション」と呼ぶ。レコードリレーションは、「2つのレコードオブジェクトが関連する」か否かを判定するためのブーリアン関数と考えることができる。

レコードリレーションは、プロセスフローの知識があるプロセス定義者が、レコードビューの定義を参照して定義する。例えば、車の標準的なデータモデルとして定義したレコードビュー CAR_VIEW に適合したレコードビューオブジェクト X とエンジンのレコードビュー ENGINE_VIEW に適合したレコードビューオブジェクト Y を考えたとき、「X.ENGINE_ID と Y.ENGINE_ID の間には同じ値が入っている」という条件が満足されたときに、「X と Y は関連している」とみなすことになる。つまり、レコードリレーションは、その条件を定義するものであり、図 4はそれをレコードビュー間のリンク（線）として表現している。

4.4. クエリ設計とクエリ要求の定義

データベース検索において、事前にクエリの形式を入力しておくことでクエリの実行戦略の最適化を行うことは多くの場面で利用されている（JDBC における PreparedStatement 等）。とりわけ、トレーサビリティにおいては、分散したデータソースに保持されるレコードに対してクエリを発行する必要が生じするため、実行されるクエリのパターンに応じた最適化の余地が大きい。従って、本稿が提案するシステムにおいてはクエリの設計情報と、パラメータ割り当てを分離して取り扱う方針を採用している。クエリの設計情報をクエリ設計、パラメータをクエリパラメータと呼ぶ。例えば、「車の VIN(Vehicle Identification Number)から車に含まれるエンジンの製造年月日を取得」がクエリ設計、「車の VIN= 1GNDM19W7VB229863」がクエリパラメータとなる。なお VIN とは、アメリカで販売される車に付与されている 17 桁の記号＋番号のことである。

トレーサビリティシステムを利用して情報を検索するアプリケーションを設計する検索要求設計者が、トレーサビリティシステムに対する入力（クエリパラメータ）とシステムからの出力をクエリ設計として定義する。クエリ設計はクエリ設計 ID を識別子として持つ。一方、システムに対する検索要求をクエリ要求と呼び、アプリケーションを通して検索を実行する検索要求発行者が検索実行時にクエリパラメータを入力することで作成される。クエリ要求はクエリ ID 及びクエリパラメータを含むデータである。

5. システム構成

本章では、提案システムの構成について記述する。

5.1. コンポーネント

提案システムのコンポーネント図を図 5に示す。本システムは大別すると以下の 4 つのコンポーネントから構成されている。

- データ検索インターフェース
  トレーサビリティシステムを利用して情報を検索するアプリケーションからのクエリ要求を受け付けて、実行結果を返す Web サービスを提供する。

- データコレクター
  データソースが提供する Web サービスを呼び出して、データを取得する。

- 定義管理部
  各アクターにより本システムに登録される各種定義を管理する。

- 検索エンジン

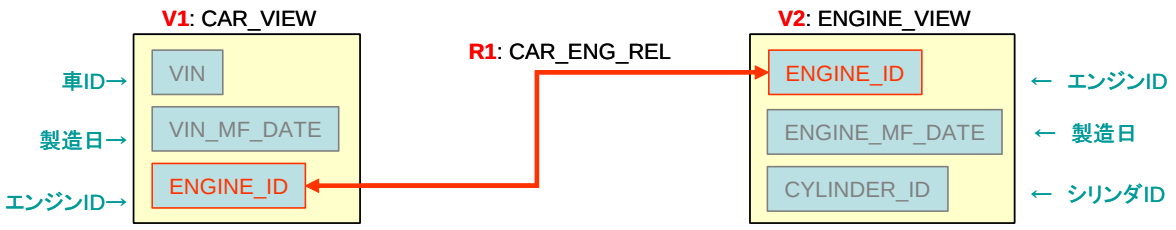

図 4　レコードリレーションの例

各種定義を基に、クエリ設計からどのデータソースのどの Web サービスをどのような手順で呼び出せばよいかという検索ストラテジーを組み立てる。データ検索インターフェースを通じて、アプリケーションからクエリ要求を受け取ると、検索ストラテジーに従って、データコレクターを呼び出し、各データソースから必要なデータを取得し、結果を構成してアプリケーションへ出力する。

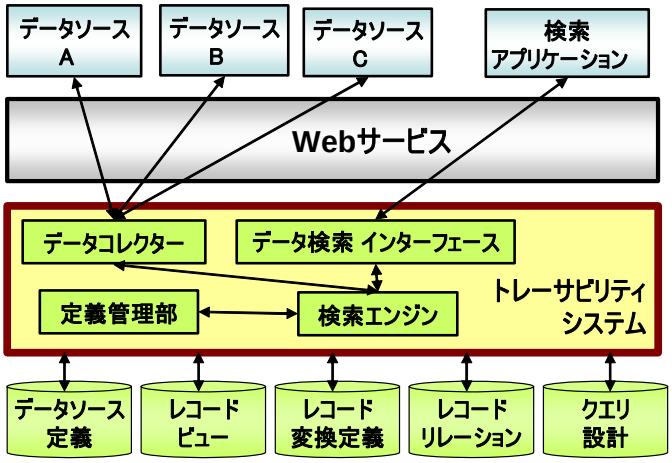

図 5　システム構成図

なお実装には、データコレクターに IBM 社の WebSphere® Information Integrator V8.2[4]を適用した。またトレーサビリティシステム全体をエンタープライズアプリケーションとして構築し、WebSphere Application Server V5.2[5]上に展開することで運用した。

5.2. コンポーネントフロー

クエリ設計の登録時とクエリ要求の実行時の流れを示す。あらかじめデータソース定義、レコードビュー、レコード変換定義、レコードリレーションの登録は終了し、定義管理部からアクセスできる状態にしておく（0)。検索要求設計者がクエリ設計をトレーサビリティシステムに登録すると（1)、検索エンジンは関係のある定義類(データソース定義､レコードビュー、レコード変換定義、レコードリレーション）を定義管理部からロードし（2)、どのデータソースのどの Web サービスをどのような手順で呼び出せばよいかという検索ストラテジーを組み立てて、保存する（3)。

つづいて、検索要求発行者が、登録済みのクエリ手段に基づくクエリ要求を検索インターフェースの Web サービスを通じて本システムに入力する（4)。検索エンジンはクエリ ID から検索ストラテジーをロードし、データコレクターを通じて分散したデータソースから必要な Web サービスを呼び出してデータを取得した後、複数のデータソースから収集したデータを統合し（5)、結果をアプリケーションに返す（6)。

6. おわりに

本稿では疎結合な関係にある企業間のトレーサビリティを実現するため、以下のような特徴を持つシステムを提案した。

- トレーサビリティのための２種類のインターフェースを Web サービスで提供するデータソースが検索対象
- 検索処理、ビジネスモデル、データソースという３つの異なる構成定義を独立したアクターにより実現
- ビジネスプロセスに関連して生成されるデータに対して標準的なビューとデータソース間の関連に基づく分散検索の検索要求設計
- 頻繁なビジネスプロセスの変更に伴う検索手順の煩雑な変更手続きを大幅に軽減

本システムにより、分散したデータソースに保持されるデータに対して、統一的に検索手段を用いてトレーサビリティが実現される。

謝辞

本研究について御討議いただいた松澤裕史氏、伊藤貴之氏、Christine Robson さんに感謝致します。

文　献

[1] ISO9000:2000 品質マネジメントシステム – 基本と用語.

[2] EPCglobal, http://www.epcglobalus.org/

[3] EPC Tag Data Standards Version 1.1 Rev. 1.24, http://www.epcglobalinc.org/standards_technology/EPCTagDataSpecification11rev124.pdf

[4] IBM WebSphere Information Integrator V8.2, http://www-06.ibm.com/jp/software/websphere/ii/v82/

[5] IBM WebSphere Application Server V5.2, http://www-06.ibm.com/jp/software/websphere/wv5/appserv_v5.html

† IBM, WebSphere は IBM Corporation の商標である