社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－DD－57 (6)
2006／9／29

インスタンス検索におけるCQLの記述性に関する一考察

細川 晃† 溝口 祐美子† 村山 廣†

†株式会社東芝 研究開発センター 〒212-8582 神奈川県川崎市幸区小向東芝町1
E-mail: †{akira.hosokawa, yumiko.mizoguchi, hiroshi.murayama}@toshiba.co.jp

あらまし 膨大なデータを効率よく分類する手段の一つとして，クラス階層構造が幅広く利用されている．これまでに，製品オントロジーを初めとしたクラス階層で表現されたデータを効率よく表現したり検索したりする手段として，様々なデータモデルとその問い合わせ言語が提案されてきた．

本稿では，これら既存のデータモデル言語およびその問い合わせ言語に比べて，クラス階層におけるメタデータおよびデータの双方を高度に検索・管理する機能を持つCQLを提案する．その評価基準として，製品オントロジーにおける3つの典型的な問い合わせに対する問い合わせ文の記述性を比較した．その結果，これら既存の言語に比べてCQLの高い記述性が確認された．

キーワード CQL，オントロジー，問い合わせ言語

A study on the descriptive power of CQL for retrieving data content

Akira Hosokawa†, Yumiko Mizoguchi†, Hiroshi Murayama†

†Toshiba Corporation 1, Komukai-Toshiba-cho, Saiwai-ku, Kawasaki, 212-8582, Japan
E-mail: †{akira.hosokawa, yumiko.mizoguchi, hiroshi.murayama}@toshiba.co.jp

Abstract Hierarchical class structure is widely used to classify various types of information and data. For this, various data models and query languages have been proposed to effect an efficient representation and search, especially of product ontology. CQL which we propose here provides rich functions for query and management of both metadata and data about a class hierarchy. Particularly in this paper, we compare the CQL with several well-known query languages in terms of three typical queries about product ontology, and the results clearly show the advantage of CQL over others.

Keyword CQL, ontology, query language

1. はじめに

膨大なデータを効率よく分類する手段の一つとして，クラス階層構造が幅広く利用されている．クラス階層構造では，各クラスは自らが定義したプロパティと上位クラスから継承したプロパティとから記述される．データはプロパティの値の集合として，いずれかのクラスに関連付けられる．

このようなクラス階層構造は実ビジネスでも利用されており，例えば製品・部品のライフサイクルを管理するためのデータモデルを規定するISO13584[1]やプラントのライフサイクルを管理するためのデータモデル規定するISO15926[2]のように，国際標準として規格化されたものもある．これらのオントロジーは企業内のデータ管理および企業間取引にも既に利用され始めており，今後ますます広く普及していくものと考えられる．

これまでに，オントロジーとデータを効率よく表現したり，それらを検索したりする手段として，様々なデータモデルやその問い合わせ言語が提案されたり，適用されてきた．例えば，リレーショナルモデルのための言語であるSQL[4]，オブジェクト指

向データモデルのためのOQL[5]，RDF/RDFSのためのSPARQL[6]やRQL[7]などは，その典型的な例であると言える．

我々も，これまでにクラス階層のための宣言的で中立な問い合わせ言語である CQL（Class Query Language）[3]を提案してきた．CQLはSQLを数学的なクラス理論に拡張したものであり，従来の問い合わせ言語に比べてクラス階層におけるメタデータおよびデータの双方の操作や検索を柔軟かつ容易に行うことができる．

本稿の目的は，ISO13584に従った製品オントロジーを実例として，CQLと既存のいくつかの典型的な問い合わせ言語の記述性を検証，確認することである．ここで，一般にオントロジーに対するオペレーションのほとんどは検索であり，特にインスタンスを検索する場合が多い．また，クラス階層およびデータの操作に比べると，個々のユーザの要求の違いにより，検索の条件は多岐に渡る．例えば，これまで実際の製品オントロジーを運用した中で，「オブジェクト指向的に下位クラスを含めたクラスのインスタンスを検索したい」という要求だけでなく，「あるクラスのインスタンスを検索したいが，より汎用的な製品があればそれも含めて検索したい」という要求や，「ある特徴（プロパティ）に着目して，その特徴を持つクラスのインスタンスを検索したい」という要求が得られた．従って，このようなクラス階層とそのインスタンスを検索する手段を提供する際には，ユーザが所望のデータを問い合わせるために記述するクエリは簡単かつ簡潔である方が望ましい．従って，本稿では製品オントロジーにおけるインスタンス検索を対象として，その典型的な3つの問い合わせに対する問い合わせ文を作成し，これらを比較することでCQLの記述性を評価する．

2. CQL

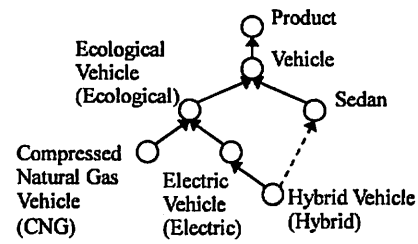
2.1. CQLの概要

CQL（Class Query Language）[3]はSQLを数学的なクラス理論に拡張した宣言的で中立的な言語であり，オントロジーおよびそれに従って表現されたデータを包括的に操作・問い合わせする機能を有する．CQLでは，クラス C はメンバー x についての真偽値を返すプロパティ $p(x)$ を用いて数学的に次のように定義される．

$$C = \{x \mid p(x)\}$$

CQLでは，各クラスおよび各プロパティにそれぞれBSU（Basic Semantic Unit）コードと呼ばれる固有のユニークなIDが割り当てられ，これによりそれぞれが一意に識別される．

CQLのクラス階層はUniversal Class（Universe）を最上位クラスとするアサイクリックなグラフ構造で表され，オブジェクト指向的に上位クラスの持つプロパティを下位クラスが継承する単純継承階層で構成される．そのため，各クラスはたかだか1つの親を持つが，アサイクリックなグラフ構造であることを満たしていれば，他のクラスからプロパティを直接輸入することができ，輸入されたプロパティも他のプロパティと同様に子孫のクラスに継承される．



Product (pid, name, maker, price)
Vehicle (weight, max_speed)
Sedan (displacement, engine_type)
Electric (motor_max_power, motor_type)

図 1 CQLのデータモデル

図1はこれらの特徴を踏まえたCQLのクラス階層およびクラスで定義されるプロパティの例であり，ISO13584に従った製品オントロジーを表している．ここで，円はクラスを，実線の矢印は矢印の先端を親とする継承関係を，破線の矢印は矢印の先端をプロパティの輸入先とする輸入関係を表す．また，図の下部はクラス名とそのクラスで定義されたプロパティ名の対応関係を表す．なお，前述のように本来クラスおよびプロパティの識別にはBSUコードを用いるが，分かりやすさのため，本稿ではBSUコードの代わりに名称を用いてそれぞれを識別するものとする．また，本稿ではUniversal Classの表記を省略する．

2.2. CQLの特徴

CQLの文法はSQLのそれとよく似ており，シンプルで容易に理解することができる一方で，特徴的な多くの機能を有する．本項では，インスタンスの検索に有効な機能に絞ってこのうちの2つを言及する．

2.2.1. 検索範囲の指定

CQLは，他のオブジェクト指向的な問い合わせ言語と同様，子孫クラスを含めた検索のための機能を提供する．これに加えて，継承関係における先祖クラスの検索機能を提供する．更に，プロパティの輸入関係を含めた上位クラスおよび下位クラスの検索もサポートする．

表1 検索範囲の指定に用いられる接尾辞

接尾辞	機能説明
%	指定されたクラスとその子孫クラスのみを検索
!	指定されたクラスとその先祖クラスのみを検索
*	指定されたクラスとプロパティの輸入関係（輸入元）を含めた子孫クラスを検索
$	指定されたクラスとプロパティの輸入関係（輸入先）を含めた先祖クラスを検索

表1は前述の検索範囲の指定に用いられる4種類の接尾辞とそれぞれの機能を示したものであり，FROM句内で必要に応じてクラスのBSUコードに続けて記述される．これらの接尾辞を利用することで，上位クラスの検索，下位クラスの検索だけでなく，それらにプロパティの輸入関係を含めるか否かを簡潔に指定することができる．

表2 検索範囲の指定の例

範囲の指定	検索対象クラス
Vehicle	Vehicle
Sedan$	Sedan
Hybrid!	Product, Vehicle, Ecological, Electric, Hybrid
Sedan*	Sedan, Hybrid
Hybrid%	Product, Vehicle, Ecological, Electric, Hybrid, Sedan

表2は，図1のクラス階層を対象として，表1の接尾辞を用いて検索範囲の指定をした場合の検索対象クラスを表したものである．この例では，下位クラスの検索ではSedanクラスを，上位クラスの検索ではHybridクラスをそれぞれ起点とした．

2.2.2. クラス間のブール演算

CQLはFROM句内での複数のクラス間の積（intersection），和（union），差（subtraction）の3種のブール演算をサポートしており，検索範囲を指定する接尾辞とともに用いることで，複雑な検索範囲の指定を集合論的な観点から容易に行うことができる．ここで，前述したクラスの定義を用いて二つのクラス $A=\{x|\phi(x)\}$ および $B=\{x|\varphi(x)\}$ を表した場合，これらのクラス間の積，和，差はそれぞれ数学的に以下のように定義される．

$$A \cap B = \{x \mid \phi(x) \wedge \varphi(x)\}$$
$$A \cup B = \{x \mid \phi(x) \vee \varphi(x)\}$$
$$A - B = \{x \mid \phi(x) \wedge \neg\varphi(x)\}$$

なお，リレーショナルモデルでもテーブル間のブール演算が定義されているが，これは基本的に対象となるテーブル間でカラムの集合が等しい場合に限られる．

表3は，CQLのブール演算で用いられる演算子を表す．

表3 ブール演算に用いられる演算子

演算の種類	演算子
積（intersection）	AND または &
和（union）	+ または ,
差（subtraction）	-

表4は，図1のクラス階層を対象として，表3の演算子を用いて2つのクラス間のブール演算をした場合の検索対象クラスを表したものである．

表4 クラス間のブール演算による検索範囲指定の例

範囲の指定	検索対象クラス
Ecological* & Sedan*	Hybrid
Ecological* + Sedan*	Ecological, LNG, Electric, Sedan, Hybrid
Ecological* - Sedan*	Ecological, LNG, Electric

3. 既存の問い合わせ言語

本章では，CQL と比較されうる，様々なデータモデルに対する既存の問い合わせ言語を略説する．

3.1. SQL

SQL[4]はリレーショナルデータベースにおいてデータの操作や検索を行うための言語であり，ISO によってその標準仕様が策定されている．これまでに幾度かの改訂が行われてきており，その過程でオブジェクト指向の機能が取り入れられてきた．

SQL で図 1 のようなクラス階層のインスタンスを検索する場合，前述のオブジェクト指向の機能を利用する方法が挙げられる．すなわち，ユーザ定義型と型付き表を利用し，クラス階層の各クラスをユーザ定義型，各クラスのインスタンスの集合をユーザ定義型から生成されるテーブルとして実現することができる．しかし，この方法では指定されたクラスとその下位方向への検索しか行えず，クラス階層のための検索機能としては不十分である．

一方，継承関係やプロパティの輸入関係の情報をデータとしてそれぞれテーブルに記憶し，これとインスタンスを記憶するテーブルとを用いることで，CQL と同等の検索を行うことができる．すなわち，SQL 標準で規定されている WITH 句による再帰問い合わせを用いて，継承関係にあるクラスやプロパティの輸入関係にあるクラスを抽出することができる．

3.2. OQL

OQL(Object Query Language)[5]は ODMG（Object Database Management Group）によって策定された，オブジェクト指向データベース（OODB）を問い合わせるための宣言的な問い合わせ言語である．OQL は SQL を拡張したものであるが，SQL と異なりデータの更新や削除などの操作機能は有していない．OQL の問い合わせ構文は SQL と同様に SELECT-FROM-WHERE で構成され，問い合わせの結果として，リテラルの集合または構造化オブジェクトの集合を返す．

3.3. RQL

RQL（RDF Query Language）[7]は，RDF[8]および RDF Schema のための問い合わせ言語である．RQL の構文は OQL をベースとしており、select-from-where で構成され、その結果は RDF グラフではなく変数の集合として返される。RQL の変数にはスキーマを問い合わせるためのクラス変数、プロパティ変数、インスタンス変数の 3 つがあり、クラス変数は「$変数名」で、プロパティ変数は「@変数名」で表される。問い合わせ対象の RDF グラフは from 句内に記述され、where 句で条件が指定される。

RQL の特徴として，スキーマと RDF データとを密に連結した問い合わせを行える点が挙げられる．たとえば「あるクラスとそのサブクラスのインスタンスのみ」といった問い合わせを容易に行うことができる．

なお，これまで RDF の問い合わせ言語は多くのものが提案されてきたが，現在は 2006 年 4 月に W3C で勧告候補になった SPARQL（SPARQL Protocol And RDF Query Language）[6]が RDF の標準的な問い合わせ言語となりつつある．しかし，SPARQL は言語仕様自体に rdfs:subClassOf などの推移型プロパティ (transitive property)をたどる機能が盛り込まれておらず，この機能を実装側で用意される推論機構に委任している．従って，本稿では SPARQL ではなく，継承関係に沿った検索を機能として持つ RQL を比較の対象とする．

4. インスタンス検索におけるクエリの比較

本稿では，図 1 に示した製品オントロジーモデルを対象にインスタンスの検索を行った場合を想定する．

4.1. 準備

まず，それぞれの問い合わせ言語で検索するための準備として，図 1 のクラス階層をそれぞれのデータモデルで記述する．なお，簡単のために以下では各クラスでのプロパティの輸入先は高々1 つのクラスのみとして考える．

4.1.1. リレーショナルモデルによる表現

図 1 のクラス階層に対して SQL で問い合わせができるように，リレーショナルモデルで表現する．ここでは，表 5 のテーブル（CLASS_TABLE とする）を用意し，クラス階層の情報を記述した．ここで，id カラムには各クラスの ID を，superclass カラムには

親クラスのIDを，import_fromカラムにはプロパティの輸入先クラスを記述するものとする．プロパティの情報を記憶するテーブル（PROPERTY_TABLEとする）およびインスタンスを記憶するテーブル（INSTANCE_TABLEとする）では，それぞれ対応するクラスのIDを記憶するカラムを用意し，これを用いてクラスと関連付ける．なお，紙面の都合上，これらのテーブルの構成図は省略する．

表 5　CLASS_TABLEの例

id	superclass	import_from
Product		
Vehicle	Product	
Ecological	Vehicle	
Sedan	Vehicle	
CNG	Ecological	
Electric	Ecological	
Hybrid	Electric	Sedan

4.1.2.　オブジェクトモデルによる表現

図1のクラス階層の各クラスをオブジェクトモデルのクラス，各プロパティをメソッド，各インスタンスをオブジェクトと見なし，OQLでインスタンスを検索できるように，オブジェクトモデルで表現する．オブジェクトモデルでは，図1の継承関係をオブジェクトモデルのクラスの継承を用いて表すことができるが，そのままではプロパティの輸入関係を表すことができない．従って，プロパティの輸入関係を表す方法として，オブジェクトモデルのinterfaceを用いる．

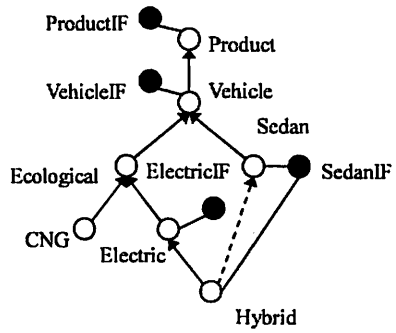


図 2　オブジェクトモデルによる図 1のモデルの表現

図 2は，オブジェクトモデルで図 1のモデルを表した例である．図中の黒丸はinterfaceを表し，それぞれのinterfaceでメソッドとしてプロパティが宣言される．また，白塗りの円（クラス）から黒塗りの円（interface）への実線は，クラスがinterfaceを実装していることを示す．プロパティの輸入関係を含む検索を行う場合は，該当するinterfaceを通して検索を行えばよい．

4.1.3.　RDF/RDFSによる表現

図 3は，図 1のモデルをRDF/RDFSで表した例である．図 3の円はRDFのクラスを，長方形はRDFのリテラルを表す．また，実線の矢印は矢印の先端を親とする継承関係（すなわちrdfs:subClassOf）を，破線の矢印は矢印の先端を輸入元とするプロパティの輸入関係表すRDFのプロパティ（RDFプロパティの名称もimport_fromとする）を，点線の矢印はそれぞれのプロパティに該当するRDFのプロパティを表す．なお，この図では簡単のため，各クラスやプロパティの名前空間接頭辞を省略した．

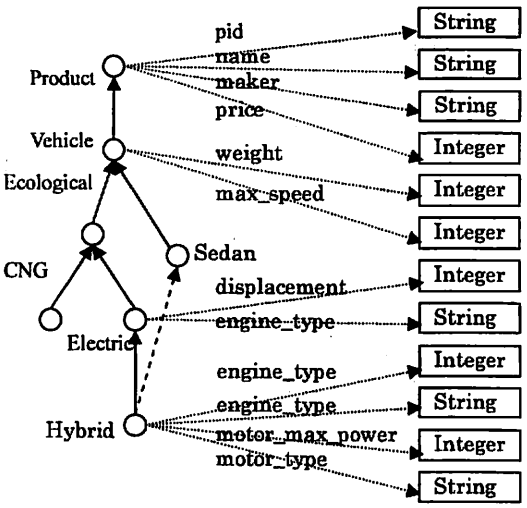


図 3　RDF/RDFSによる図 1のモデルの表現

4.2.　クエリの比較

本稿では，3種の典型的なインスタンス検索について各問い合わせ言語を用いてクエリを記述し，それらを比較する．

4.2.1.　下位クラスを含めた検索

まず，指定されたクラスとそれ以下のクラスの

インスタンスを検索する際のクエリを比較する．この検索の例題として，以下のものを用意した．

例題　Vehicle クラスの特徴を持つクラスのインスタンスの中で maker プロパティを持ち、さらにその値が'X 社'のものを返す

表 6 は，この例題に対して，各問い合わせ言語で記述したクエリを示す．これらを比較すると，CQL や OQL が簡単なクエリで記述できるのに対して，SQL では再帰検索のための WITH 句を記述する必要があり，クエリが複雑になっているのが分かる．一方，RQL は FROM 句で指定した RDF トリプルに該当するものを抽出した後，WHERE 句で検索するクラスの範囲やプロパティの値の条件を指定する必要があり，CQL や OQL に比べると煩雑なクエリになっている．

表 6　下位クラスを含めた検索のクエリ

言語	クエリ
CQL	select * from Vehicle* where maker = 'X 社';
SQL	WITH RECURSIVE 中間表(id) AS (SELECT id FROM CLASS_TABLE WHERE name = 'Vehicle' UNION SELECT CT.id FROM 中間表, CLASS_TABLE AS CT WHERE CT.superclass = 中間表.id OR CT.import_from= 中間表.id) SELECT DISTINCT * FROM INSTANCE_TABLE WHERE class_id IN (SELECT id FROM 中間表) AND maker = 'X 社';
OQL	SELECT * FROM VehicleIF v WHERE ((Product)v).maker = "X 社";
RQL	SELECT DISTINCT X FROM $C1{X}.maker.{Y}, $C1{X}.import_from.$C2 WHERE ($C1 <= Vehicle OR $C2 <= Vehicle) AND Y = "X 社"";

4.2.2.　クラス間のブール演算を用いた検索

次に，指定された複数のクラス間でブール演算を行い，その結果選択されたクラスのインスタンスを検索する際のクエリを比較する．この検索の例題として，以下のものを用意した．

例題　Ecological クラスおよび Sedan クラスの特徴を持つクラスのインスタンスを返す

表 7 は，この例題に対して，各問い合わせ言語で記述したクエリの例を示す．

表 7　クラス間のブール演算を用いた検索のクエリ

言語	クエリ
CQL	select * from Ecological* AND Sedan*;
SQL	WITH RECURSIVE 中間表 A(id) AS (SELECT id FROM CLASS_TABLE WHERE name = 'Ecological' UNION SELECT CT.id FROM 中間表 A, CLASS_TABLE AS CT WHERE CT.superclass = 中間表 A.id OR CT.import_from= 中間表 A.id) WITH RECURSIVE 中間表 B(id) AS (SELECT id FROM CLASS_TABLE WHERE name = 'Sedan' UNION SELECT CT.id FROM 中間表 B, CLASS_TABLE AS CT WHERE CT.superclass = 中間表 B.id OR CT.import_from= 中間表 B.id) SELECT DISTINCT * FROM INSTANCE_TABLE WHERE class_id IN (SELECT 中間表 A.id FROM 中間表 A, 中間表 B WHERE 中間表 A.id = 中間表 B.id);
OQL	SELECT (Vehicle)e FROM ElectricIF e, SedanIF s WHERE ((Product)e).pid = ((Product)s).pid;
RQL	SELECT DISTINCT X FROM $C1{X}.import_from.$C2 WHERE ($C1 <= Ecological OR $C2 <= Ecological) AND ($C1 <= Sedan OR $C2 <= Sedan);

各問い合わせ言語のクエリを見ると，クラス間のブール演算機能を持つ CQL は極めて簡単なクエリ

でこの検索を実現できることがわかる．一方，SQLは，クラス間のブール演算を行うために，指定されたクラス数と同じ回数の再帰問い合わせを行ってそれぞれのクラスの集合を得た後，それらに対して集合演算を行う必要がある．また，OQLはFROM句でブール演算を行うことができず，FROM句で指定されたクラスやインターフェースから抽出されるインスタンスに対して，WHERE句でブール演算用の条件を記述する必要がある．一方，RQLでも，FROM句で一度該当するRDFグラフを抽出した後，WHERE句でブール演算を行う必要がある．

4.2.3. 上位クラスを含めた検索

最後に，指定されたクラスとそれ以上のクラスのインスタンスを検索する際のクエリを比較する．この検索の例題として，以下のものを用意した．

例題 Hybrid クラスとその上位クラスのインスタンスの中で motor_max_power プロパティを持ち、さらにその値が 30 より大きいものを返す

表 8 は，この例題に対して，各問い合わせ言語で記述したクエリの例を示す．

表 8 上位クラスを含めた検索のクエリ

言語	クエリ
CQL	select * from Hybrid% where motor_max_power > 30;
SQL	WITH RECURSIVE 中間表(id) AS (SELECT id FROM CLASS_TABLE WHERE name = 'Hybrid' UNION SELECT CT.id FROM 中間表, CLASS_TABLE AS CT WHERE CT.id = 中間表.superclass OR CT.id = 中間表.import_from) SELECT DISTINCT * FROM INSTANCE_TABLE WHERE class_id IN (SELECT id FROM 中間表);
OQL	
RQL	SELECT DISTINCT X FROM $C1{X}.m otor_max_power{Y}, ^Hybrid.import_from.^$C2 WHERE (Hybrid <= $C1 OR $C2 <= $C1) AND Y > 30;

ここで，OQLは，クラス階層の上位の方向に向かって検索する手段を提供していないため，結果を斜線で表示している．上位クラスの検索についても，CQLはFROM句で簡潔に記述することができるが，他の問い合わせ言語は，下位クラスを検索するクエリと同様，複雑な問い合わせ文を記述する必要がある．

4.3. 考察

これまで見てきたように，CQLはクラス階層における検索を少ない記述量で容易に行える．これは，2.2節で説明したCQLの2つの特徴によるものであり，これらを組み合わせることで，クラス階層が複雑になったり，指定されるクラスの数が増えたりしても容易にクエリを記述することができる．

SQLでクラス階層を表す方法として，クラス階層の継承関係やプロパティの輸入関係をデータとしてテーブルに持たせたが，テーブルを再帰的に問い合わせるための WITH 句を記述する必要があるため，クエリの記述性が低下してしまうことが分かる．ここで，実装上のテクニックとして，テーブルへのデータの持たせ方を改良したり付加情報を与えたりすることで，本稿で挙げたクエリよりも少ない記述で問い合わせることができる．しかし，これらの手法は言語仕様の範疇を越えているため，今回は対象としていない．

OQLはオブジェクト指向に従って下位クラスを含めたインスタンスの検索を行うことができるが，基本的に上位クラスを検索したり，継承関係以外のクラス間の関係を扱ったりすることができないため，現実の製品データの問い合わせにOQLをそのまま用いるには機能が不十分であると言える．

RQLは上位クラスおよび下位クラスの検索をサポートしているが，CQLと比較してクエリは複雑になる．特に，RQLではクラス変数やプロパティ変数などを用いて，RDFの足りプル単位でクエリを作成するため，複雑な検索を行う際に変数の数が多くなり，クエリの記述性や可読性が低下してしまう．

5. まとめ

本稿では，我々が提案しているCQLを紹介し，階層上に分類されたクラスのインスタンスを検索する際に記述するクエリを調べることで，インスタン

ス検索におけるCQLの記述性を確認した．

まず，下位クラスのインスタンスの検索，クラス間のブール演算によるインスタンスの検索，上位クラスのインスタンスの検索の3種類の典型的なインスタンス検索の例題を設定した．次に，現在広く利用されているデータモデル，すなわちリレーショナルデータモデル，オブジェクトデータモデル，RDF/RDFSの3つのデータモデルでクラス階層構造を表現し，それぞれの問い合わせ言語であるSQL，OQL，RQLで前述の例題に対するクエリを記述した．CQLのクエリをこれらと比較した結果，インスタンス検索におけるCQLの記述性の高さとその簡潔さを確認できた．

本稿ではその頻度の高さや重要性からインスタンスの検索のみを対象に比較・検証したが，CQLは他にもメタデータ検索を初めとする様々な強力な機能を持つ．今後，これらの包括的な比較・評価を行い，その有効性を示していく．

文　献

[1] ISO Standard: ISO13584 Industrial automation systems and integration -Parts Library-"

[2] ISO Standard: ISO15926 Life Cycle Data for Oil & Gas Production Facilities

[3] Y. Mizoguchi-Shimogori, H. Murayama, N. Minamino, "Class Query Language and its application to ISO13584 Parts Library Standard," ECEC2002 Concurrent Engineering: System integration for profit, pp128-135, 2002

[4] 土田正士，小寺孝 著:"-SQL最新標準規格-SQL2003ハンドブック，" ソフト・リサーチ・センター，2004

[5] R. G. G. Cattell, D. K. Barry, M. Berler, J. Eastman, D. Jordan, C. Russell, O. Schadow, T. Stanienda, and F. Velez, "The Object Data Standard: ODMG 3.0, " Morgan Kaufmann Publishers, 2000

[6] SPARQL Query Language for RDF: http://www.w3.org/TR/rdf-sparql-query/, 2006

[7] G. Karvounarakis, A. Magkanaraki, S. Alexaki, V. Christophides, D. Plexousakis, M. Scholl, and K. Tolle. Querying the Semantic Web with RQL. Computer Networks, pp617–640, Aug. 2003

[8] 神崎正英 著:"セマンティック・ウェブのための RDF/OWL 入門," 森北出版, 2005