PCO：インターネット上における分散的なスキーマに対応した複合 EC コンテンツ記述言語

倉光君郎[†] 坂村健[††]

本論文は，インターネット上の商店やサービス業者間において，商品やサービスの属性情報（EC コンテンツ）を共有・交換可能にする新しい意味記述言語，PCO（Portable Compound Object）を提案する．PCO 言語は，分散環境におけるスキーマ共有を促進するため，独自のデータモデルである意味コレクション機構（PSCF）を備えている．意味コレクション機構は，分散環境における EC コンテンツ記述に対して，意味仕様の相互互換性，動的な意味拡張性，複合コンテンツの記述性を提供することができる．我々は，PCO 言語を用いて EC コンテンツを記述することで，インターネット上で分散しているコンテンツ提供者・サービス提供者の間で EC コンテンツの共有・交換が可能になることを示した．

PCO: EC Content Description Language Supporting Distributed Schemas across the Internet

Kimio Kuramitsu[†] and Ken Sakamura[††]

This paper proposes a new description language, called PCO (Portable Compound Object) for EC contents, or goods and service information. PCO supports, as an original data model, a semantic collection framework to coordinate schemas authored independently. The framework brings three benefits into EC content description: semantic interoperability, dynamic extensibility and compound representation. We describe that PCO language enables the effective interchange of EC contents across merchants and intermediaries on the Internet.

1. はじめに

近年の急激なインターネットの普及は，電子商取引（Electronic Commerce: EC）の可能性を拡大させている．今日，Web 上の商業サイトでは，多くの仮想店舗が HTML（Hyper Text Markup Language）を用いて，音楽 CD や書籍，チケットなどの商品カタログを提供している[1),2)]．しかし HTML は，Web ページのレイアウトが主な目的であり，「商品やサービスの情報」の意味（semantics）を表現するのには適していない．たとえば，HTML で記述された商品の価格は，人間のみ理解できるが，マシン解釈上は単なる数字列にすぎない．たとえば，HTML では，複数の商店間にまたがって価格条件から検索を行うことが不可能である．

† 東京大学大学院理学系研究科情報科学専攻
Department of Information Science, Graduate School of Science, The University of Tokyo
†† 東京大学総合研究博物館
The University Museum, The University of Tokyo

HTML の意味記述の不完全さは，電子商店（electronic merchants）や電子仲介業者（electronic intermediaries）の間で，「商品やサービス」の情報を相互に交換・共有することの妨げになっている．本研究の目的は，インターネット上の商店やサービス業者間において，「商品やサービスの情報」を共有・交換可能にする新しい意味記述言語を提案することである．

インターネット EC の特徴は，不特定多数の参加者がお互いの「商品やサービス」を提供している点にある．「商品やサービス」の情報を正しく交換するためには，お互いにデータ表現を定義するスキーマを共有する必要がある．しかし，インターネット EC の規模と多様さは，従来の特定分野と提携企業に限られていた EDI（Electronic Data Interchange）[3)~5)]とは異なり，1 つの標準化団体が単一のスキーマを管理するのを許さない．分散的に定義されたスキーマの相互運用性が必要となる．

分散スキーマの相互運用性は，単に分散的にスキーマを定義できるだけでなく，スキーマの再利用性や連結性，分散環境におけるスキーマの配布などにも十

分に対応できなければならない．現在，インターネット上で意味記述を交換するアプローチとして，古くは HTML 文章にメタ情報を埋め込む方法[6)~8)]や最近では XML（eXtensible Markup Language）をベースにした EDI 言語が提案されている[16)~18)]．しかし，これら既存の提案は，分散スキーマに対するサポートに関しては不完全である．

我々は，インターネット EC に適したデータモデルとして，PCO（Portable Compound Object）言語モデルの開発を行ってきた．PCO 言語モデルの特徴は，PCO コンテンツ（データ表現）に対する意味制約を，スキーマから直接かけるのではなく，独自の PCO 意味コレクション機構によって，複数の分散スキーマの動的な構成からかけられる点である．その結果，PCO コンテンツ間の意味相互性（semantic interoperability）の保証が容易になり，既存のスキーマの再設計なしにコンテンツの意味仕様が修正可能となった．

「商品やサービス」には，パッケージ商品など複数の商品が組み合わされた形態のものも多い．PCO 言語モデルのもう 1 つの特徴は，分散スキーマの枠組みに加え,「商品やサービス」の記述に対して，複合記述の機能を備えている点である．PCO 言語では，異種スキーマを階層化して 1 つの意味仕様に連結することが可能で，シンタックス的にもセマンティクス的にも透過的に複合コンテンツを記述できる．

我々は，PCO 言語モデルを提案し，2 つの XML ベースの言語 PCL（Portable Composite Language：PCO のコンテンツ表現と交換形式）と PSL（PCO Specification Language：PCO のスキーマ言語）で実装した．本論文は，PCO 言語モデルとその実装について報告する．2 章では，既存の研究の問題点を論じながら,「商品とサービス」の意味記述への機能要求をまとめる．3 章では，PCO 言語モデルを定義する．4 章では，XML ベースによる PCO 言語モデルのデータ表現とスキーマ定義について述べる．5 章では，PCO 言語モデルの設計の妥当性について論じる．6 章では，サンプル記述を通して，PSL と PCL の記述力の評価を行う．7 章では，本論文を総括する．

2. EC コンテンツの意味記述

EC コンテンツとは，インターネット上の電子商取引において交換される「商品やサービス」の属性情報を表現するデータのことである．本章では，EC コンテンツの特性を分析し，それを意味記述するための言語要求について述べる．

2.1 EC コンテンツの特性分析

まず我々は，民間企業の協力を得て，実際に販売している商品（チケット，本，音楽 CD，衣類，文房具，PC 機器，中古車など）の意味属性の調査を行った☆．その結果，EC コンテンツの特性は，以下の 4 点に集約できることが分かった．

(1) 多様性：実際に取引されている商品やサービスの種類は多岐にわたり，これらがすべて EC コンテンツとして記述対象となる．
(2) カスタマイズ性：同一の記述対象であっても，企業ごとに記述したい商品属性が微妙に異なる．
(3) ダイナミックス性：商取引環境は，つねにダイナミックに変化している．既存の商品でも，法律や商取引慣行の変化によって，表記すべき属性が変わる．
(4) 商品の複合性：商品やサービスには，複数の異なった商品を組み合わせたパック商品と呼ばれる形態がある（例：旅行ツアーやパソコンセットの販売など）．複合コンテンツの記述は，EC 推進事業者の間でも強い要望となっている．

2.2 関 連 研 究

企業は,「商品やサービス」に関する情報を独自のスキーマに基づくデータベースで管理してきた．電子商取引では，これらのデータをネットワーク上で交換することが求められる．しかし，各企業が独自に設計したスキーマは，データのセマンティクス交換において大きな障害になる．本論文が扱う EC コンテンツの意味記述とその交換は，インターネット上におけるデータ記述の意味相互性（semantic interoperability）の応用例と位置付けることができる．

従来の EDI システム[3)~5)]では，特定の業界や限定された取引者が標準化団体の定義した意味仕様（スキーマ空間）にあわせて，意味表現された商品カタログや取引データを交換することができた．しかし，オープンなインターネット EC では，一般消費者を含めた不特定多数の参加者の間で取引が行われる．標準化団体があらゆる商品やサービスに対応したスキーマ空間を策定することは不可能である．逆に，分散的に定義されたスキーマ間で意味相互性を保証することが求められる．

我々は，インターネット上で分散的にスキーマを定義可能な言語モデルについて調査を行った[6)~12)]．その結果，分散スキーマとその意味相互性は，主に次の

☆ 同様な調査は，ECOM（Electronic Commerce Promotion Council of Japan）の商品属性情報標準化検討ワーキンググループでも行われている[19)]．

2つのアプローチで提供されていることが分かった．

オントロジー型： スキーマ間の関係は，スキーマの上位記述であるオントロジーやスキーマ辞書で表現する．MMF[6]は，EC コンテンツ記述言語の中でオントロジー型の代表例であり，4 層の言語構造（メタ記述，スキーマ，オントロジー，概念辞書）をサポートしている．

オブジェクト指向型： スキーマ間の関係は，スキーマ設計における継承関係やインタフェースの共有によって表現する．Java 言語[20]は，インターネット上の分散環境に適したオブジェクト指向型言語の代表例であり，プログラムコードの配布以外に，直列化（Serialization）機能によって，オブジェクトを意味付けされたデータとして配布・交換可能である．

またインターネット上のデータ交換における最近の傾向として，XML[11]の応用がさかんになっている．XML は，W3C が規格化を進める SGML の Web 版サブセットであり，独自の言語モデルとして，DTD（Document Type Definition）とその名前空間（namespace）[13]を規定している．しかし，これらの規格は基礎的かつ汎用的であり，セマンティクスを定義するスキーマとしては曖昧な点が多い．そのため，XML 応用言語として，記述対象にあわせ，オントロジーやオブジェクト指向スキーマを拡張した提案がなされている[12),14)]．

2.3 機能分析と課題

本節では，2.1 節の EC コンテンツの特性を具体化した小シナリオを用意し，オントロジー型の代表例として MMF，オブジェクト指向型言語の代表例として Java 言語，さらに XML/DTD の記述機能の分析を行う．

2.3.1 意味仕様の相互互換性

シナリオ (1) 消費者は，パソコン（PC）とガイドブック（BOOK）を同時に購入した．それぞれの意味記述から合計金額を自動的に計算したいが，両者が共通して持つ PRICE 属性は同じ意味を表しているのだろうか？（税込み？税別？）

分散的に定義されたスキーマは，お互いに意味仕様の協調機構がなければ，単に複数のスキーマが無関係に独立しているだけである．分散スキーマ言語は，各スキーマが定義する意味仕様の相互互換性が検証できなければならない．

MMF は，スキーマ定義の上位記述として，オントロジーと概念辞書を用意している．これらを用いて，異種スキーマ間の各属性について，同義や包含関係などを指定できる．しかし，このアプローチは，誰かがオントロジー記述をスキーマの上位記述として管理しなければならないので負担も大きい．そのため，オントロジー型言語は，知識ベースシステムを導入した例も少なくない[14),18)]．

Java は，オブジェクト指向モデルのスキーマの多相性によって，スキーマ設計時に簡単な意味相互性の保証ができるため，特別なスキーマの上位記述を必要としない．たとえば，PC クラスと Book クラスが共通した上位クラスを持ち，そこに Price 属性が定義されているなら，PC クラスと Book クラスの Price の同義性は保証される．しかし，この意味相互性はスキーマ設計に強く依存することになる．

XML/DTD は，個別に定義された DTD 間の関連性を定義する言語的フレームワークは用意されていない．

2.3.2 スキーマの 2 次カスタマイズ性

シナリオ (2) パソコン（PC）の意味仕様を定義したスキーマがすでに存在する．この状況で，新しくノートパソコン（NOTEPC）のスキーマを定義する．

我々は，既存のスキーマを再利用して新しいスキーマを設計することを「2 次カスタマイズ性」と呼ぶ．2 次カスタマイズ性は，スキーマ設計の効率にとって重要であるが，カスタマイズされたスキーマの再配布も問題となる．

XML/DTD や MMF は，既存のスキーマをカスタマイズし，新しいスキーマを設計できない．スキーマの設計者は，PC 用スキーマとは独立して NotePC 用スキーマを定義することになる．これらに対し，Java は，PC クラスの属性を継承することで，NotePC クラスを階層的に設計できる．ただし，このようなスキーマの階層性はスキーマの共有・交換を複雑にする．たとえば，Java Applet では，送信側と受信側でスキーマの共有を実現するため，ネゴシエーション機構が必要である．

2.3.3 意味仕様のダイナミクス

シナリオ (3) 情報家電の普及にともなって，USB や IEEE1394 など新しいインタフェース規格が登場した．パソコンやオーディオ機器のカタログでは，新しいインタフェース規格への対応を追加したい．

プログラミング言語は，長いプログラミング経験からコアライブラリが設計され，現在ではスタックやハッ

表 1 関連研究との比較
Table 1 Comparison of related works.

	本研究	既存の記述言語		
	PCO 言語モデル	MMF	XML1.0 規格	Java（Serialized Object）
スキーマ言語	PSL 言語	SOIF ベース（独自）	SGML 系 DTD	Java 言語
スキーマ間の意味相互性	○	○	×	○
スキーマ設計後，コンテンツの相互性を追加できるか？	○（Collection）	○（オントロジー，概念辞書）	△（Namespace）	×
（2 次）カスタマイズ性	○（オブジェクト指向）	×	×	○（オブジェクト指向）
コンテンツ配布時のスキーマ再配布の回数	0 回：必要なし	1 回：URI リンク	1 回（以上）：URI リンク	複数回：上位クラスも配送する独自機構
スキーマの再設計なしで属性を追加できるか？	○（PCO Semantic Collection Framework）	×	△（Namespace）	×
異スキーマ空間の階層化	○	×	規定なし	○
複合コンテンツの表現	○	×	規定なし	△（セマンティクスのみ）

シュ表など基本クラスの変更の必要性は少ない．しかし EC コンテンツでは，非常に基本スキーマであっても，突然修正しなければならない状況が予想される．

一般に，使用中のスキーマ修正・再設計は，非常に困難な作業である．複数のバージョンの混在を認めれば，分散バージョン管理システムが必要となる．さらに，スキーマの設計モデルがオブジェクト指向の場合，複数のスキーマ階層のうち，どの階層を修正すべきか，複数の設計者の間で意志決定しなければならない．たとえば，パソコンやオーディオ機器に共通する適切な上位クラスがなかった場合，クラス階層の再設計自体も必要になる．MMF，XML/DTD，Java は，このような EC コンテンツのダイナミクスさに対応できるバージョン管理機構や記述フレームワークを用意していない．

2.3.4 EC コンテンツの複合化

シナリオ (4) 組立てパソコン（PC）キットは，様々な PC パーツ（CPU，M/B，HDD など）から構成される．キットの提供者は，パーツ提供者が提供する EC コンテンツを再利用することで，簡単にキットのコンテンツを提供したい．

商品やサービスの一般的な形態の 1 つにパック商品がある．EC コンテンツ記述言語は，複数の EC コンテンツを組み合わせて，新しい複合 EC コンテンツを表現できることが求められる．

Java は，コンポジション機能で他の Java オブジェクト（クラス）を含めることができる．つまり，CPU や HDD オブジェクトを PC オブジェクトに埋め込むことができる．しかし，これはセマンティクスレベルの連結であり，シンタックス的には完全に分離している．PC の記述側から，HDD の各属性を直接記述することはできず，HDD はあくまで埋め込まれたオブジェクトとして表現される．

XML は，Namespace によって複数の DTD 空間を同一文章中に含めることができる．しかし，これはシンタックス的にスキーマ空間を混在させたのにすぎない．現在，Namespace と DTD はまったく独立した規格になっているため，2 つの空間をセマンティック的に結合することはできない．そのため，PC DTD の中に HDD DTD があるのか，その逆なのか判断できない．

MMF は，複合コンテンツに対しては，シンタックス的にもセマンティクス的にも対応していない．

2.4 ま と め

EC コンテンツの意味記述には，「商品やサービス」の特性より，4 つの要求（意味相互性，カスタマイズ性，ダイナミック性，意味複合性）を満たす必要がある．しかし，これらの要求は，オントロジー型，オブジェクト指向型の言語モデルでは，十分に対応できないことが分かった．また，XML にオントロジーやオブジェクト指向の枠組みを単純に拡張するだけでは十分でないことも分かった．**表 1** は，本論文で提案する PCO 言語と既存の記述言語の比較をまとめたものである．

3. PCO 言語モデル

我々は，2 章で述べた EC コンテンツ記述への要求を満たすために，PCO 言語モデルを設計した．PCO 言語モデルの特徴は，スキーマにおいてオブジェクト指向モデルを適用した多相化と多層化を実現するだけでなく，PCO 意味コレクション機構（PCO Semantic Collection Framework）と呼ばれる独自の機構によって，コンテンツの意味仕様を動的に調整できる点である．本章では，PCO 言語モデルを定義する．

3.1 スキーマ

意味表現（semantic representation）は，あるデータ値に対して，それがどのような意味であるか識別する修飾子（qualifier）を付加した記述である．PCO 言語モデルでは，修飾子とそのデータ値の 1 対 1 の組合せを意味表現の基本単位とする．たとえば，商品価格の意味記述では，実際の価格（1500）に対して，修飾として Price のように表現可能である．

PCO コンテンツは，複数の意味表現から構成される．意味仕様（semantic specification）は，EC コンテンツの設計図に該当し，修飾子の集合体である．スキーマ（schema）は，複数の修飾子の集合を意味仕様として定義する基本（最小）単位である．意味仕様を外部から区別するための名前を持っている．

表記 1（スキーマ）： 本論文では，スキーマと修飾子をそれぞれ Sn と Qn（n は自然数）で表記する．したがって，スキーマ $(S1)$ に含まれる修飾子 $(Q1, Q2)$ は，式 (1) のように表記する．

$$S1 = (Q1, Q2) \tag{1}$$

3.1.1 スキーマの 2 次拡張性

PCO 言語モデルでは，スキーマ設計の 2 次拡張機構として，オブジェクト指向（OO）モデルの多重継承を採用した．

表記 2（多重継承）： 上位スキーマの多重継承は，+ 記号を用いて表記する．式 (2) は，上位スキーマ $S1$ と $S2$ から，スキーマ $S3$ を派生させた例である．派生スキーマ $(S3)$ は，上位スキーマの定義を再利用し，その差分 $(Q5)$ だけ定義すればよい．

$$\begin{aligned} S1 &= (Q1, Q2) \\ S2 &= (Q3, Q4) \\ S3 &= S1 + S2 + (Q5) \\ &= (Q1, Q2, Q3, Q4, Q5) \end{aligned} \tag{2}$$

3.1.2 名前衝突

オブジェクト指向言語の多重継承では，名前衝突（name conflict）という限界がある．たとえば式 (3) では，$Q2$ は $S1$ と $S2$ で，$Q3$ は $S2$ と $S3$ で，定義されているため，最終的にはどちらの修飾子の意味を継承すべきか曖昧になる．

$$\begin{aligned} S1 &= (Q1, Q2) \\ S2 &= (Q2, Q3) \\ S3 &= S1 + S2 + (Q3) \\ ? &= (Q1, Q2, Q3) \end{aligned} \tag{3}$$

プログラミング言語の場合，優先度制御で名前衝突を避けることもできる．しかし PCO 言語モデルでは，$S1$，$S2$，$S3$ における $Q2$ と $Q3$ の同義性の保証が，意味仕様間の相互性の根拠となる．そのため，修飾子の名前衝突が発生した場合，継承失敗として扱う．

3.2 意味コレクション機構

PCO 言語モデルの特徴は，スキーマとその実体（instance）であるコンテンツの間に，意味コレクション機構（PCO Semantic Collection Framework）を導入した点である．従来の OO モデルでは，スキーマは直接，コンテンツの意味仕様を決定していた．これに対して，PCO 言語モデルでは，意味コレクション機構が仲介し，最終的なコンテンツの意味仕様を決定することになる．

表記 3（コレクション）： コレクションは，コンテンツの意味仕様を決定するスキーマの組合せである．このとき，多重継承と名前衝突の検出が適用される．したがって，コレクション（Collection）は，式 (4) のように表記する．

$$\begin{aligned} S1 &= (Q1, Q2) \\ S2 &= (Q3, Q4) \\ \text{Collection} &= S1 + S2 \\ &= (Q1, Q2, Q3, Q4) \end{aligned} \tag{4}$$

PCO 言語モデルでは，修飾子は同じ名前であっても，それを定義するスキーマが異なれば意味が異なるものと解釈される．つまり，識別子は単独で意味を決定するのではなく，スキーマの制約を受けて意味が決まる．その点で，スキーマは修飾子のコンテクスト（context）の役割を果たしているといえる．したがって，意味コレクションは，機能上「名前なしスキーマ」といえるが，スキーマと異なり修飾子を直接定義することはできない．

3.3 コンポジション：複合化機能

一般に，複合コンテンツの意味仕様は，複数のスキーマを階層的に結合することによって構成を表現される．これには，意味コレクション機構による複数スキーマの多相化とは異なった，複数のスキーマ空間を多層化する機構が必要となる．PCO 言語モデルは，オブジェクト指向モデルのコンポジション概念を導入し，修飾子付きのスキーマによって，スキーマと意味仕様の多層化を実現する．

表記 4（コンポジション）： コンポジションは，スキーマに修飾子として意味をもたせる機能である．本論文では，スキーマ $S1$ に修飾子 $Q4$ を意味付けした場合，:記号を連結子として $Q4 : S1$ のように表記する．さらに，修飾子付きのスキーマは，式 (5) のように修飾子を展開することができる．

$$\begin{aligned} S1 &= (Q1, Q2) \\ S2 &= (Q3, Q4 : S1) \\ &= (Q3, Q4, Q4.Q1, Q4.Q2) \end{aligned} \tag{5}$$

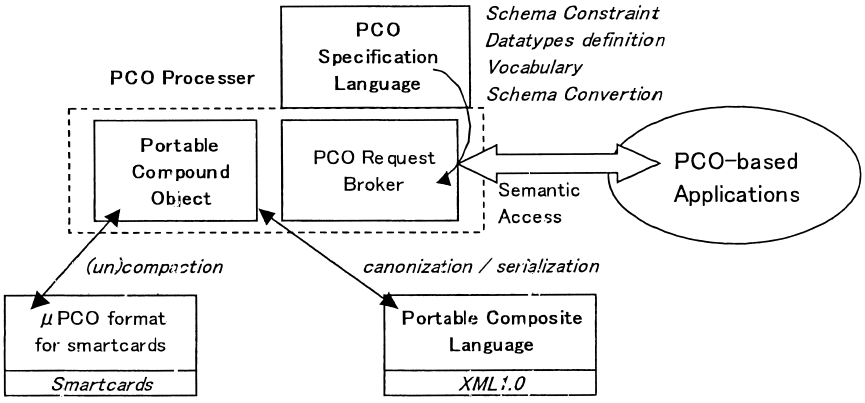

図 1 PCO 言語アーキテクチャ
Fig. 1 PCO language architecture.

注意：$Q4.Q1$ は，$Q4$ によって意味付けされた（限定された）$Q1$ という意味を持つ修飾子になる．

4. PCO 言語の実装

我々は，3 章において，EC コンテンツ記述言語として PCO 言語モデルの論理設計を行った．本章では，PCO 言語モデルのスキーマとコンテンツデータ表現するために，XML ベースの言語 PSL（PCO Specification Language）と PCL（Portable Composite Language）を設計する．**図 1** は，PCO 言語アーキテクチャの概要を示している．

4.1 PCO 言語アーキテクチャ

PCO 言語アーキテクチャは，Portable Compound Object（PCO）と名付けた概念的なデータ構造を中心に構成されている．PCO は，チャンク（Chunk）と呼ばれる記述単位を最小のセマンティクス単位にしている．チャンクは，次の 4 種類のフィールドを 1 組にして構成される．

(a) チャンク識別子は，チャンク値の意味を識別子する．

(b) チャンク型は，チャンク値の構造を表現する付加情報である．PCO の特徴は，文字列（string）や数値（number），複合型（class）などの基本型に意味型を追加できる点である．たとえば，string/phone（電話番号を表す文字列），number/currency-yen（通貨円を表す数値），class/$S1$（$S1$ スキーマの複合型）が意味型の例である．さらに PCO では，チャンク型に MIME タイプを指定して，マルチメディアオブジェクトなどの意味付けが可能である．

(c) チャンク属性は，チャンク値のオプショナル付加情報である．本論文では詳細を述べないが，アクセス権や上書き権を制御するために利用される．

(d) チャンク値は，意味を与えられる実現値である．PCO は，単純な文字列だけでなく，マルチメディアオブジェクトやハイパーリンク，動的な評価式をチャンク値として意味記述できる．

PCO Specification Language（PSL）は，PCO 言語モデルに準じたスキーマ定義のための PCO 上位記述言語である．そのため，PSL スキーマは，PCO（PCL）の意味仕様を直接制約せず，意味コレクションによって間接的に PCO と関連付けられる．PCL は，データ表現としては PSL スキーマの制約を受けないが，意味アクセスのとき PSL スキーマの制約を受け，正しく意味記述にアクセス可能となる．

Portable Composite Language（PCL）は，PCO 言語モデルに従って記述された EC コンテンツを，インターネット上で交換するために XML1.0 形式でエンコーディングしたデータ表現である．ただし PCO 言語アーキテクチャでは，その表現形式を XML に固定せず，用途や目的に応じて変更できるように設計されている．本論文では詳細を述べないが，我々は，PCL とは別に，IC カード（smartcard）に適した記述形式 μPCO フォーマットも設計している．

4.2 PCO Specification Language

PCO 言語は，スキーマ定義の単位を「クラス」と呼ぶ．本論文では，PCO 言語モデルによって抽象的に定義された「スキーマ」に対して，PSL によって具体的に定義された「クラス」と，用語を使い分ける．

4.2.1 クラス定義

PSL では，`<ClassDef>` 要素を用いてクラス定義を行う．`<ClassDef>` 要素には，そのクラスに関するメタ情報（設計者やソースの位置），上位クラス，チャ

ンク定義が含まれる．上位クラスからの継承関係は，<super> 要素を用いて示すことができる．チャンク定義は，<ChunkDef> 要素を用い，チャンク名（識別子）や型（type）などを宣言する．クラス設計者は，オプショナルなチャンク宣言として，重要度や禁止属性などの属性をチャンク宣言に追加することもできる．次のクラス定義は，3 章の式 (2) におけるスキーマを例に，PSL で定義したものである．

```
<ClassDef name="S3">
  <super name="S1" href="http://..."/>
  <super name="S2" href="http://..."/>
  <ChunkDef name="Q5" type="string"/>
</ClassDef>
```

PSL では，コンポジション（スキーマの階層的な結合）の定義は，チャンク定義において，クラス型（class）を指定することで行う．式 (5) におけるコンポジションを含んだスキーマ定義は，PSL では次のようにクラス定義される．

```
<ClassDef name="S1">
  <ChunkDef name="Q1" type="string"/>
  <ChunkDef name="Q2" type="number"/>
</ClassDef>
<ClassDef name="S2">
  <ChunkDef name="Q3" type="string"/>
  <ChunkDef name="Q4" type="class/S1"/>
</ClassDef>
```

4.2.2 名前衝突の抑制と曖昧さの制御

多重継承による名前衝突は，PCO 言語モデルにおいて意味の曖昧さをもたらす．我々は，他のクラスで定義されたチャンクとの意味の同義性を明確に定義するために，<equiv> 要素を追加した．式 (3) のような曖昧な多重継承は，次のように明確に制御できる．

```
<ClassDef name="S1">
  <ChunkDef name="Q1" type="string"/>
  <ChunkDef name="Q2" type="number"/>
</ClassDef>
<ClassDef name="S2">
  <ChunkDef name="Q2" type="number">
    <equiv class="S1"/> </ChunkDef>
  <ChunkDef name="Q3" type="string"/>
</ClassDef>
<ClassDef name="S3">
  <super class="S1">
  <super class="S2">
  <ChunkDef name="Q3" type="string">
    <equiv class="S2"/> </ChunkDef>
</ClassDef>
```

4.3 Portable Composite Language

PCL は，半構造化（semi-structured）データのモデル☆を拡張し，ストラクチャとセマンティクスを完全に独立して設計されている．また，セマンティクスは，ユーザが自由に拡張可能な意味空間とコンテンツを制御するために必要な固定的なメタ情報用の意味空間から構成される．それらは，次の 3 種類の XML タグセット（DTD）によって構成され，XML 名前空間技術[13]によって，お互いの混在を可能としている．

(a) PCL/DTD 空間：PCO 特有のデータ表現（ストラクチャ）をサポートするための固定のタグセット．本論文では，名前空間を省略する．
(b) チャンク空間：EC コンテンツの意味記述を表現するための動的なタグセット．本論文では，名前空間として chunk を用いる．
(c) メタチャンク空間：(b) を制御するためのメタ情報のためのタグセット．本論文では，名前空間として meta を用いる．

次のソースは，完全な PCL 文章の例である．

```
<?xml version="1.0"?>
<PCL
  xmlns="http://PCO/ns/pcl-dtd"
  xmlns:meta="http://PCO/ns/meta-chunks-dtd"
  xmlns:chunk="http://PCO/ns/semantic-collection">

  <meta:Author lang="ja"> 倉光君郎 </meta:Author>
  <meta:Class> Name </meta:Class>

  <chunk:Name lang="ja"> 倉光君郎 </chunk:Name>
  <chunk:Name lang="en"> Kimio Kuramitsu </chunk:Name>
</PCL>
```

4.3.1 チャンク記述

PCL では，チャンク識別子を XML タグ名，チャンク型や属性を XML 属性，チャンク値を XML 要素として表現する．たとえば，チャンク識別子（Name），チャンク型（string），チャンク属性（なし），チャンク値（倉光君郎）のチャンクは，次のように表記される．

```
<chunk:Name type="string"> 倉光君郎 </chunk:Name>
```

PCL のチャンク表現には，PSL クラスで定義されるチャンク型が含まれている．そのため，クラス情報がなくても，コンテンツを字句解釈することができる．

4.3.2 メタチャンクとコレクションの記述

メタチャンクは，「(b) チャンク空間」の制御情報を記述するための意味仕様が固定されたチャンク空間である．PCO プロセッサは，メタチャンクの情報を一般のチャンクと同様にアクセスできる．PCL では，スキーマとコンテンツを多相的に結合する意味コレクションも，メタチャンクで記述する．PCL では，式 (3.4) で例にあげたコレクションは，次のように記述する．

```
<meta:Class type="string">
  <mul><val>S1</val><val>S2</val></mul>
</meta:Class>
```

☆ 半構造化データとは，伝統的な構造化データ（関係データベースの内部レコードやプログラミング言語の構造体）とは異なり，データ表現がスキーマに依存しないデータのことである[21]．

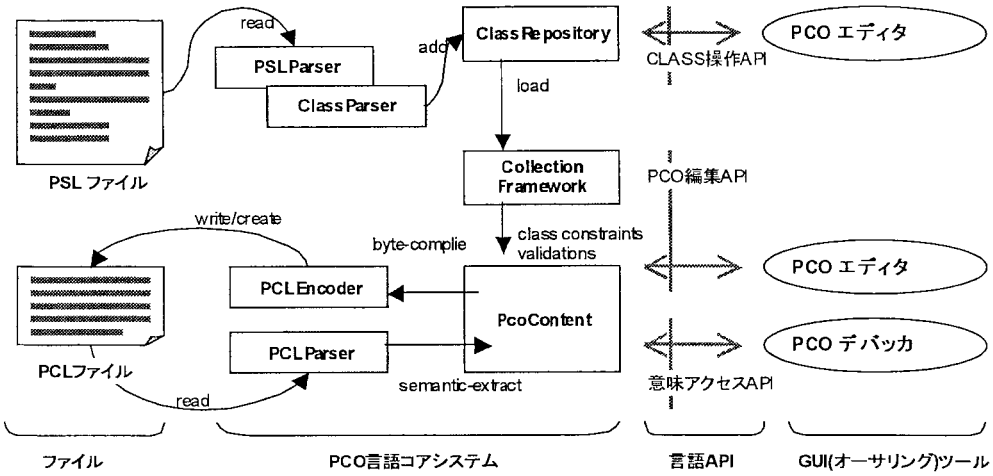

図 2 PCO 言語システムのコンポーネント関係図
Fig. 2 Component relationship in the implemented system.

注意：<mul> と <val> は，多値文字列の構造を表現するための PCL コンテナである．

4.4 複合記述におけるシンタックスの透過性

PCO 言語モデルは，コンポジションを採用し，複合コンテンツを表現可能としている．PCL の特徴は，コンポジションにおけるシンタックスの透過性を提供する点である．ここでは，4.2.1 項で定義されたクラス S1 とクラス S2 による複合記述で考えてみる．

従来のプログラミング言語型コンポジションは，シンタックス的には別途作成したコンテンツを作成し，それを埋め込むことで複合コンテンツを表現することになる．PCL も，他の PCL 文章をチャンク値として埋め込むことで，同様に複合記述が可能になる．

```
<PCL>
<meta:Class> S2 </mata:Class>
<chunk:Q3 type="string"> ABC </chunk:Q3>
<chunk:Q4 type="class/S1">
    <PCL>
    <meta:Class> S1 </meta:Class>
    <chunk:Q1 type="string"> abcdefg
    </chunk:Q1>
    <chunk:Q2 type="number"> 1245678
    </chunk:Q2>
    </PCL>
</chunk:Q4>
</PCL>
```

さらに，PCL では，チャンク構造を階層化することで，次のような同一シンタックス（PCL 文章）内での複合記述も可能である．PCL の特徴は，複合に対する異なった 2 種類の表現を意味的には同じものとして扱える点である．

```
<PCL>
<meta:Class> S2 </mata:Class>
<chunk:Q3 type="string"> ABC </chunk:Q3>
<chunk:Q4 type="class/S1">
    <chunk:Q1 type="string"> abcdefg
    </chunk:Q1>
    <chunk:Q2 type="number"> 1245678
    </chunk:Q2>
</chunk:Q4>
</PCL>
```

4.5 PCO 言語システムの実装

我々は，Java2（JDK1.2）を用いて，PCO 言語システムのプロトタイプ実装を行った．実装の規模は，ソースコードで約 3 万行，クラス数で約 230 個程度である．実装された PCO 言語システムは，簡単な XML 関連のツールを含むライブラリだけでなく，GUI ベースのオーサリングツールを含まれている．**図 2** は，実装された PCO 言語システムのコンポーネント関係を大まかに示している．

5. PCO 言語モデルの設計合理性

本章では，意味コレクション機構によって，PCO 言語モデルが次の 3 点の設計目標を満たすかどうか，その設計合理性を論じる．

(1) コンテンツの記述者は意味仕様をカスタマイズできなければならない．PCO 言語モデルは，その際のスキーマ再設計とその再送を抑制したい．

(2) PCO 言語モデルは，オントロジーなどスキーマの上位記述の必要なしに，動的にコンテンツの意味相互性を保証したい．

(3) PCO 言語モデルは，複雑な分散バージョン管理機構の必要なく，EC コンテンツのダイナミクスにあわせて，新しい意味仕様の記述を可能にしたい．

5.1 カスタマイズ性とスキーマ再配送

ネットワーク上では，送信側と受信側が同じスキーマを持っていた場合のみ，正しく意味表現を伝えることができる．そのため，もしコンテンツの記述者が意味仕様をカスタマイズするため，スキーマを変更した場合，コンテンツと同時に変更されたスキーマも送信しなければならない．これに対し，PCO 言語モデルでは，スキーマ設計者とコンテンツ記述者の役割を完全に分離し，コンテンツ記述者がスキーマを再設計しなくても，意味コレクションの構成を変更することによって，それぞれの意味仕様の調整が可能となっている．

次の例では，既存のスキーマ $(S1, S2, S3)$ に対して，いっさいの修正や再設計を施さなくても，意味コレクション機能によって，2 種類の意味仕様（式 (6) と式 (7)）をカスタマイズできることを示している．実際のデータ交換では，スキーマの再送の代わりに，コンテンツ中に意味コレクションの情報を含めるだけで済む．

$$
\begin{aligned}
&S1 = (Q1, Q2)\\
&S2 = (Q1, Q3, Q4)\\
&S3 = (Q5)\\
&\text{Collection} = S1 + S3 \qquad (6)\\
&\qquad\quad = (Q1, Q2, Q5)\\
&\text{Collection} = S2 + S3 \qquad (7)\\
&\qquad\quad = (Q1, Q3, Q4, Q5)
\end{aligned}
$$

5.2 意味相互性の基本モデル

PCO 言語モデルは，意味仕様の相互性を検証する枠組みとして，スキーマの多相性を利用する．次の例では，式 (8) と式 (9) には，それぞれ共通して $Q1$, $Q2$ が含まれる．このとき，$Q1$ は共通して $S1$ から派生しているので同義と見なせる．つまり，式 (8) と式 (9) は両方とも，共通に $S1$ によって多相的に意味制約されていることになる．

$$
\begin{aligned}
&S1 = (Q1)\\
&S2 = S1 + (Q2)\\
&\quad = (Q1, Q2)\\
&S3 = S1 + (Q2, Q3)\\
&\quad = (Q1, Q2, Q3)\\
&\text{Collection} = S2 \qquad (8)\\
&\qquad\quad = (Q1, Q2)\\
&\text{Collection} = S3 \qquad (9)\\
&\qquad\quad = (Q1, Q2, Q3)
\end{aligned}
$$

ただし，PCO 言語モデルでは式 (8) と式 (9) に含まれるお互いの $Q2$ の関係やより複雑な関係（たとえば式 (8) の $Q2$ と式 (9) の $Q3$ の意味関係など）は，保証されない．しかし，EC コンテンツでは，すべての意味仕様の相互性を検証する必要はなく，価格など一部の意味互換性が保証できれば十分である．その点で，PCO 言語モデルのスキーマ多相性による意味相互性の保証はシンプルであるが，十分だと考えられる．さらに，その特徴は，スキーマ多相化を，従来の OO モデルがスキーマ設計時に限定していた点に対し，コンテンツの記述時にも意味コレクションとして動的に実現できる点である（具体例は，次節も参照）．

5.3 意味仕様の柔軟な追加

意味コレクション機構は，EC コンテンツのダイナミクスにあわせて意味仕様を変更する場合にも大きな利点となる．たとえば，式 (8) と式 (9) に，新しく $Q4$ を追加する場合を考えてみる．

従来の OO モデルでは，一般的に既存のスキーマを修正することになるが，$S1$ と $S2$，$S3$ のどちらに $Q4$ を加えるか，設計者が決定しなければならない．これは，インターネット規模の分散環境では負担が大きくなる．もし $S1$ の派生スキーマの中には，勝手に上位クラスで $Q4$ を追加されたら問題となる場合もあるからだ．OO モデルではさらに，スキーマ設計者間の調整が終わるまで，記述者は $Q4$ に対する記述ができない．

PCO 言語モデルは，$Q4$ を含んだ新しいスキーマ $S4$ を用意することで，既存のスキーマを修正することなしに $Q4$ を追加できる．記述者は，個別にコレクションを調整することで迅速に対応できる（式 (10) と式 (11) を参照）．このとき，共通の $S4$ をコレクションに追加したことで，$Q4$ の相互互換性を保証できる．

$$
\begin{aligned}
&S4 = (Q4)\\
&\text{Collection} = S2 + S4 \qquad (10)\\
&\qquad\quad = (Q1, Q2, Q4)\\
&\text{Collection} = S3 + S4 \qquad (11)\\
&\qquad\quad = (Q1, Q2, Q3, Q4)
\end{aligned}
$$

5.4 ま と め

PCO 言語モデルは，意味コレクション機構の導入により，意味仕様の送信負荷の少ないカスタマイズ性，シンプルで動的な意味相互性の保証，EC コンテンツのダイナミクスに対応した拡張を備えている．

6. サンプル記述と評価

我々は，PCL と PSL を用いて，現在までに様々な商品（本，パソコン，コンサート情報，文房具，衣類など）のサンプル記述を行ってきた．本章では，実際の EC 環境を想定した具体的なシナリオを用意し，それに従ってサンプル記述を行い，PCO 言語モデルの

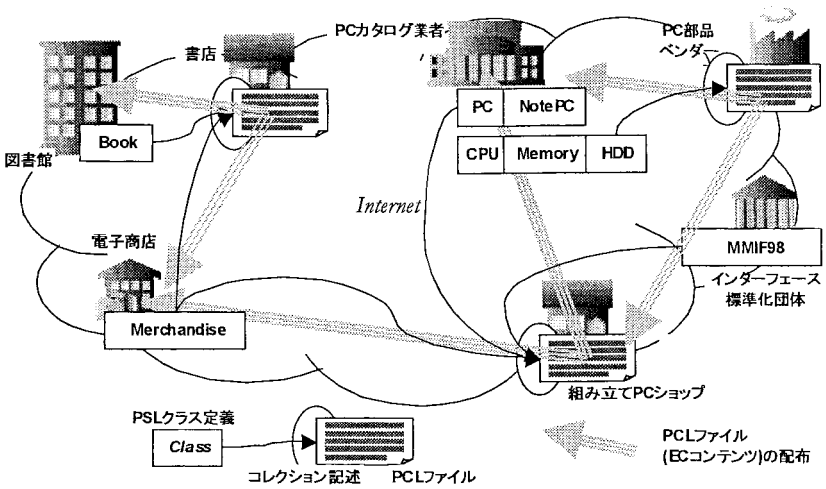

図 3 サンプル記述のためのシナリオ
Fig. 3 Scenario for sample description.

実装を評価する．

6.1 シナリオ

我々は，サンプル記述のため，簡単なインターネット上の相互サービスのシナリオを用意した．これらは，2 章で取りあげたシナリオ (1)〜(4) を検証可能な状況が含まれている．**図 3** は，サンプル記述用シナリオを概観している（文献 22) は，本シナリオに基づいた具体的なショッピング応用例である）．

PCO 言語モデルでは，スキーマの設計者とコンテンツの記述者の役割を分離できる．したがって，我々は，インターネット上の EC 業者を，サービス提供者とコンテンツ提供者の 2 種類に分類している．

コンテンツ提供者は，自店の商品カタログを EC コンテンツとして記述する．本シナリオでは，3 種類のコンテンツ提供者（書店，組立て PC ショップ，PC パーツベンダ）が登場する．それぞれ書籍コンテンツ，PC コンテンツ，CPU コンテンツを記述・提供している．

サービス提供者は，第 3 者が提供している EC コンテンツに対して，付加価値サービスを提供する．本シナリオでは，電子商店，図書館，PC カタログ業者などが登場する．電子商店は，EC コンテンツで記述された商品を電子的に販売する業者である．異種商品を扱う仮想モール（virtual mall）などが該当する．図書館や PC カタログ業者は，複数のコンテンツ提供者が提供した EC コンテンツを集めて，検索や分類サービスを行う．さらに，本シナリオでは，インタフェース標準化団体（I/F 協会）のような，特定の EC サービスを提供しないが，標準スキーマの公表だけ行う組織も想定する．

6.2 PSL によるクラスのサンプル定義

サービス提供者は，各自のサービスを遂行するために必要な意味仕様を PSL クラスとして公開する．次に載せる 6 種類のスキーマは，各サービス業者が作成したクラスの例である．各クラス定義は，それぞれ **<Author>** タグが示すクラス設計者が提供する☆．

```
<ClassDef name="Book">
<Author> 電子図書館 </Author>
<ChunkDef name="Name" type="string">
  <equiv class="Merchandise"> </ChunkDef>
<ChunkDef name="ISBN" type="string/isbn"/>
<ChunkDef name="Author" type="string"/>
</ClassDef>

<ClassDef name="Merchandise">
<Author> 電子商店 </Author>
<ChunkDef name="Name" type="string"/>
<ChunkDef name="Price" type="number/currency-yen"/>
</ClassDef>

<ClassDef name="PC">
<Author> PC カタログ業者</Author>
<ChunkDef name="Name" type="string">
 <equiv class="Merchandise"> </ChunkDef>
<chunkDef name="CPU" type="class/CPU"/>
<ChunkDef name="Memory" type="number/MB"/>
<ChunkDef name="HDD" type="number/GB"/>
</ClassDef>

<ClassDef name="NotePC">
<Author> PC カタログ業者</Author>
<superClass name="PC" locator="..."/>
<ChunkDef name="LCD" type="class//LCD"/>
<chunkDef name="Battery" type="class//Battery"/>
</ClassDef>
```

☆ サンプル記述のため，最小限のクラス定義のみ記述されている．

```
<ClassDef name="CPU">
<Author> PC カタログ業者</Author>
<ChunkDef name="Name" type="string"/>
<chunkDef name="Clock" type="number"/>
</ClassDef>

<ClassDef name="MMIF98">
<Author> I/F 標準協会 </Author>
<ChunkDef name="IEEE1394ports" type="number"/>
<chunkDef name="USBports" type="number"/>
</ClassDef>
```

6.3 PCO コンテンツのサンプル記述

コンテンツ提供者は，インターネット上で分散的に提供される EC サービスを選択し，コレクション機構を用いてそれに対応した意味仕様を調整する．どのクラスを選んだかは，コレクション記述として，メタ情報の `<meta:Class>` タグで記述する．本シナリオでは,「PC 部品ベンダー」と「組立て PC ショップ」,「書店」の 3 コンテンツ提供者は，それぞれ EC コンテンツとして，次の PCL 文章を記述・提供する．

PC パーツ（CPU）コンテンツの例

```
<PCL>
 <meta:Author type="string"> CPU ベンダー
</meta:Author>
 <meta:Class> CPU </meta:Class>
 <chunk:Name> Gmicro300 </chunk:Name>
 <chunk:Clock type="number"> 450 </chunk:Clock>
</PCL>
```

書籍コンテンツの例

```
<PCL>
 <meta:Author type="string"> 書店 </meta:Author>
 <meta:Class>
  <mul><val>Book</val><val>Marchandise</val>
   </mul>
 </meta:Class>
 <chunk:Name> PC 組立てガイド </chunk:Name>
 <chunk:ISBN type="string/isbn"> 4-9009000-64-3
</chunk:ISBN>
 <chunk:Author> 倉光君郎 </chunk:Author>
 <chunk:Price type="number/currency-yen"> 1500
</chunk:Price>
</PCL>
```

組立て PC コンテンツの例

```
<PCL>
 <meta:Author type="string">組立て PC ショップ
</meta:Author>
 <meta:Class>
  <mul> <val>PC</val><val>Marchandise</val>
    <val>MMIF98</val> </mul>
 </meta:Class>
 <chunk:Name> PC 組立てキット</chunk:Name>
 <chunk:CPU type="class/CPU">
<PCL>
   <meta:Author type="string">
      CPU ベンダー </meta:Author>
   <meta:Class> CPU </meta:Class>
   <chunk:Name> Gmicro300 </chunk:Name>
   <chunk:Clock type="number"> 450 </chunk:Clock>
</PCL>
 </chunk:CPU>
 <chunk:Memory type="class/Memory">
  <chunk:Type> DIMM </chunk:Type>
  <chunk:Size type="number/MB"> 64 </chunk:Size>
</chunk:Memory>
<chunk:IEEE1394Ports type="number">
  0</chunk:IEEE1394ports>
<chunk:USBPorts type="number">2</chunk:USBports>
<chunk:Price type="number/currency-yen">
  277000</chunk:Price>
</PCL>
```

6.4 評　　価

我々は，サンプル記述を通して，PSL と PCL の言語機能を検証・評価することができた．

スキーマの 2 次拡張性　PC カタログ業者は，既存の PC クラスを再利用して，NotePC クラスを拡張できた．

オーバライド の有効性　Book クラスと Merchandise クラスには，同じ名前の Name チャンクが共通して含まれているが，`<equiv>` 記述によって，これら 2 つのクラスの Name を同義と見なし，オーバライド可能であった．

動的な意味仕様の変更　組立て PC ショップは, MMIF 98 をコレクションに加えることで，既存のクラスを変更する必要なく，PC コンテンツに新しい属性を記述できた．

複合記述　組立て PC ショップは，複合コンテンツの作成において，CPU など部品ベンダーが提供する EC コンテンツを埋め込む，または Memory などの属性を直接記述ことができた．

意味仕様の相互互換性　電子商店は，書籍コンテンツと PC コンテンツを同じ Merchandise クラスによって制約された価格情報を持つものとして処理できた．

コンテンツの配布効率　PCO コンテンツの配布時には，クラスの再配布の必要性はなかった．たとえば，電子図書館は，書籍コンテンツ中の Merchandise を知らないが，その部分を無視して，必要な Book の意味仕様だけ取り出すことができた．

以上の結果より，PSL と PCL は，複合 EC コンテンツの意味（semantics）を十分に記述でき，その意味表現を効率良く交換できることが示された．

7. 結　　論

インターネット上の電子商取引の拡大にともなって，複数の仮想商店の間で「商品やサービスの情報（EC コンテンツ）」を共有化し，それを相互サービスへ活

用する要求が高まっている．我々は，その要求に応えるため，EC コンテンツの意味記述に適した言語モデル（PCO 言語モデル）を設計した．

PCO 言語モデルは，分散環境におけるスキーマ共有と意味相互性を促進するため，独自の意味コレクション機構を備えている．本論文では，意味コレクション機構によって，EC コンテンツ記述の意味仕様を動的に多相化でき，意味相互性の保証が容易になり，柔軟な意味拡張性も可能になることを示した．さらに，PCO 言語モデルを XML ベースで実現した PSL と PCL のサンプル記述によって，インターネット上で分散しているコンテンツ提供者・サービス提供者の間で，EC コンテンツの複合化が可能になり，効率的に共有・交換可能であることも示せた．

本論文は，PCO 言語の意味記述機能に限定して議論を行った．今後は，PCO 言語の他の特徴的な機能，インターネット上で PCO コンテンツを安全に流通させるための機構などについても，議論を深めていく予定である．そして，最終的には，PCO 言語を利用した，分散的な EC サービスを共有可能な電子市場プラットホームを開発・実証する計画である．

謝辞 本論文の執筆に対して，激励と貴重なコメントをいただいた越塚登先生（東京大学大学院人文科学系研究科）と森洋久氏（東京大学総合研究博物館）に深く感謝いたします．

参 考 文 献

1) Segev, A., Wan, D. and Beam, C.: Electronic catalogs: A technology overview and survey results, *Proc. CIKM '95* (1995).
2) Adam, N. and Yesha, Y.: Strategic directions in electronic commerce and digital libraries: Towards a digital agora, *ACM Computing Surveys*, Vol.28 (1996).
3) Electronic Data Interchange Standards, http://www.premenos.com/standards/.
4) UN/EDIFACT STANDARD DIRECTORY, http://www.unicc.org/unece/trade/untdid/.
5) AMERICAN NATIONAL STANDARD for Electronic Commerce, ANSI X12.13 (1992).
6) Sakata, T., Tada, H. and Ohtake, T.: Metadata mediation: Representation and protocol, *Proc. 6th International World Wide Web conference* (Apr. 1997).
7) Resnick, P. and Miller, J.: PICS: Internet access controls without censorship. *Comm. ACM*, Vol.39, pp.87–93 (Oct. 1996).
8) Hardy, D.R., Schwartz, M.F. and Wessels, D.: *Harvest user's manual* (Jan. 1996).
9) Internet Mail Consortium: vCard – the electronic business card version 2.1 (Sept. 1996).
10) The OBI Consortium: Open buying on the Internet (OBI): Technical specifications release V1.1, http://www.obi.org/.
11) Bray, T. and Paoli, J.: Extensible Markup Language, *W3C Working Draft* (Aug. 1997).
12) Brickley, D., Cuha, R.V. and Layman, A.: Resource description framework (RDF) schema specification, *W3C Working Draft* (Oct. 1998).
13) Bray, T., Hollander, D. and Layman, A.: Namespaces in XML, *W3C Working Draft* (1998).
14) Singh, N.: Unifying Heterogeneous information models, *Comm. ACM*, Vol.41, No.5 (1998).
15) Cover, R.: Managing names and ontologies: An XML registry and repository, http://www.sun.com/981201/.
16) Meltzer, B. and Glushko, R.: XML and electronic commerce: Enabling the network economy, *ACM SIGMOD Record*, Vol.27, No.4, pp.21–24 (Dec. 1998).
17) Dogac, A., Durusoy, I., Arpinar, S., Tatbul, N., Koksal, P., Cingil, I. and Dimililer, N.: A workflow-based electronic marketplace on the web, *ACM SIGMOD Record*, Vol.27, No.4, pp.25–31 (Dec. 1998).
18) Glushko, R.J., Tenenbaum, J.M. and Meltzer, B.: An XML framework for agent-based e-commerce, *Comm. ACM*, Vol.42, No.3 (1999).
19) ECOM (Ed.): Reports on the standardization of merchandise properties, H9-WG2-1 (1998). (in Japanese)
20) Sun Soft: Java Object Serialization Specification. Revision 1.4.1 (Oct. 1997).
21) Buneman, P.: Semistructued data, Tutorial of ACM PODS'97, http://www.cis.upenn.edu/db/.
22) Kuramitsu, K. and Sakamura, K.: Digiket: Decentralized Architecture for Worldwide Electronic Market, *Proc. IEEE COMPSAC '99* (1999).

(平成 11 年 4 月 27 日受付)
(平成 11 年 11 月 4 日採録)

倉光　君郎（学生会員）

1972 年生．1998 年東京大学大学院理学系研究科情報科学専攻修士課程修了．現在，同専攻博士課程在学中．電子商取引の分野に興味を持つ．ACM，IEEE Computer Society 学生会員．

坂村　　健（正会員）

東京大学総合研究博物館教授．1984 年より TRON プロジェクトリーダーとして新しい概念に基づくコンピュータ体系の構築に精力を注ぐ．さらに，最近はコンピュータ技術を駆使したデジタルミュージアムの構築を，東京大学総合研究博物館において手がける．