意味的な複合イベント処理を可能とするイベントベースについて

石川 佳治[1,2,a)]　佐々木 勇和[3,b)]　簗井 美咲[1,c)]　高橋 正和[1,d)]　杉浦 健人[1,e)]

概要：本稿では，複雑な意味を持つイベントを表現し，処理・蓄積できるイベントベースの概念を提案する．複合イベントを，オントロジの概念を用いて構造的に定義することがその特徴である．イベントベースは，意味的に定義されたイベントの意味を解釈し，単純なイベントから複合的なイベントを抽出する．イベント検出とその処理には，セマンティック Web の技術とリレーショナルデータベース技術の両者を活用する．本稿では，イベントベースのアイデア，アーキテクチャ，問合せ処理のアプローチを中心に，その構想について説明する．

1. はじめに

複合イベント処理（complex event processing, CEP）は，近年，データベースの研究分野のみならず，産業界を含めた応用分野においても需要なトピックとなっている [8]．その背景の一つとしては，**データストリーム**に関する問合せ処理技術やマイニング技術の継続的な開発 [1], [6], [10], [11] などにより，データストリームに基づく高レベルの応用が実現可能となってきたことによる．一方で，センサ機器の普及やモバイルコンピューティングの進展などにより，イベントを明示的に扱うことが必要な応用領域が拡大しており，より高レベルのイベントの処理要求も生まれている．

複合イベント処理においては，しばしば正規表現，あるいは正規表現を包含するような問合せ言語によりイベントが記述される [20]．また，CQL [2] のようなデータストリームのための問合せ言語を用いてイベントを記述するアプローチもある．問合せ言語にはさまざまなものがあるが，概して，これまでのデータストリーム処理や複合イベント処理においては，基本となる個々の「イベント」は単純なデータアイテム（例：数値など）からなると想定されていた．しかし，実世界における高度化するアプリケーションを考えると，より高レベルの意味的なイベントを表現する必要が生じる．以下ではそのようなドメインの例を挙げる．

例 1 (位置に基づく情報サービス)　位置に基づくサービス（location-based service, LBS）では，移動するユーザや車などに対し，さまざまな情報サービスを提供する．近年ではさらに，ソーシャルネットワークサービスと位置情報を連携した，位置に基づくソーシャルネットワーク（location-based social network, LBSN）も発展しているが，これも LBS の一種であるといえる．

LBS および LBSN においては多様なイベントを扱う必要がある．たとえば，モバイルユーザどうしの移動先での出会いを支援する LBSN においては，「ユーザの移動」という基本的なイベントのみならず，「友人どうしが近くにいる」といった高レベルのイベントを捕捉する必要がある．このイベントの検出には，「二人のユーザが友人である」という背景知識や，「二人の位置が近い」といった情報を用いることが必要となる．しかし，友人関係や近さの関係は，応用分野や実際に構築するアプリケーションに大きく依存しており，必ずしも固定することはできない．

別の応用として，災害時の避難支援のための LBSN を考える．そのような応用では，ユーザの位置の変化のイベント以外に，災害や避難に関するイベントを検出・通知したり，ユーザどうしの連携を図るなどの機能が必要となる．結果として，災害に関連した様々なイベントを扱う必要があるため，応用に応じた多様なイベントを扱えることが重要となる．

例 2 (センシングによる行動認識)　近年，センサ機器を用いて人々の行動を認識することが盛んに行われている．

[1] 名古屋大学大学院情報科学研究科
Graduate School of Information Science, Nagoya University
[2] 国立情報学研究所
National Institute of Informatics
[3] 名古屋大学未来社会創造機構
Institute of Innovation for Future Society, Nagoya University
[a)] ishikawa@is.nagoya-u.ac.jp
[b)] yuya@db.ss.is.nagoya-u.ac.jp
[c)] yanai@db.ss.is.nagoya-u.ac.jp
[d)] takahashi@db.ss.is.nagoya-u.ac.jp
[e)] sugiura@db.ss.is.nagoya-u.ac.jp

用いるセンサ機器により，どのような行動が認識されるかは大きく異なるが，「歩いている」，「走っている」などや，「料理をしている」，「食事をしている」などの行動がある程度の精度で認識できるようになっている．このような行動も一種のイベントと考えられ，イベント処理の対象となりうる．

さらに，行動認識の場合，コンテクストに関する情報や背景知識を活用して，夕方に料理イベントが検出されたら「夕食の準備」という，より洗練されたイベントを検出することなども考えられる．また，その人が朝方に散歩することを習慣としているとき，早朝の「歩いている」というイベントを「散歩」というイベントとして認識することも考えられる．すなわち，複合イベント処理を高度な行動認識にまで応用を広げた場合，コンテクストに関する情報や背景情報も含めて，高レベルのイベントを推論する機能が必要となる．

以上の例に示したような応用領域を考えた場合，高度なイベント処理を行うには以下のような機能が必要となる．

- 対象領域やアプリケーションに応じて，イベントを定義する機能：これには，イベントを組み合わせた複合イベントを定義する機能も含まれる．
- イベント定義に基づき，複合イベントを検出・処理する機能

すなわち，従来のデータストリームや複合イベント処理に比べ，より意味的なレベルでのイベント処理を行う必要がある．

このような要求を踏まえ，本研究グループでは高レベルのイベントを定義し，問合せおよび蓄積を可能とする**イベントベース**（eventbase）について研究を進めている．その特徴は，意味的なイベントの定義のために**オントロジ**（ontology）の概念を導入し，オントロジを考慮したイベント処理を行う点にある．本稿では，このイベントベースの構想と概要について述べる．イベントベースシステムでは，現在のオントロジ技術を踏まえ，**RDF**（resource descriptin framework）形式に基づくデータストリームが入力として与えられるものと想定する．高レベルのイベント記述を RDF データに対する問合せ処理に変換し，また一方でリレーショナルデータベースの問合せ技術も活用する点が特徴である．

本稿の構成は次のようになる．2 節ではシステムのアーキテクチャについて述べ，3 節ではイベントオントロジの考え方を例を挙げて説明する．4 節では問合せ処理のアイデアについて述べ，5 節では今後の発展のための議論を行う．最後に 6 節で結論づける．

2. システムアーキテクチャ

2.1 はじめに

本プロジェクトにおいて現在想定しているイベントベースのアーキテクチャについて述べる．**図 1** がその全体像である．システムは，イベントベースシステムが提供する部分と，応用分野やアプリケーションに応じて実装され提供されるアプリケーション依存の部分に分けられる．

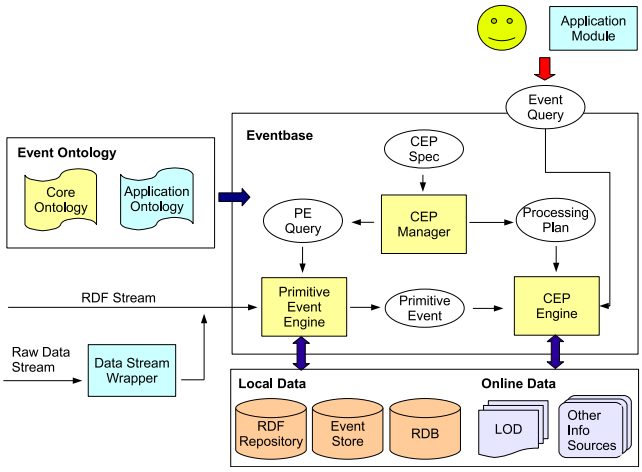

図 1 システムアーキテクチャ

入力データストリームは，LBS への応用であれば GPS により取得される位置情報ストリームなどであり，行動認識への応用であれば，信号レベルの認識処理により得られる基本的な行動イベントとなる．入力データストリームは RDF データのストリームであることを想定する．

中心となるのが**イベントベースシステム**（eventbase system）であり，イベントベースの各種機能を実現する．

2.2 複合イベント指定

CEP マネジャ（CEP manager）は，モニタリングする複合イベントの管理を行うモジュールである．CEP マネジャには，ユーザから直接的に，もしくはアプリケーションから API を通じて，モニタリングすべき複合イベントの内容が指定される．これを**複合イベント指定**（complex event specification）と呼ぶことにする．複合イベント指定は，一種の**連続的問合せ**（continual query）と考えることもでき，システムにおいて継続して実行される．複合イベント指定では，検出された複合イベントを**イベントストア**（event store）に蓄積するか否か，また，蓄積する場合にはどの程度の期間保持するかなどの指定が可能であるとする．

2.3 イベントオントロジ

本システムでは，複合イベントの概念をオントロジを用いて定義する．それぞれの複合イベントは，**プリミティブイベント**（primitive event）と呼ぶ，それ以上分解できない単純なイベントを組み合わせて定義される．そこで用いられるのが**オントロジ**（ontology）の概念である．本システムのオントロジには，大別して，イベントベースシステムが提供する基本的なイベント概念および関連する概念を定義する**コアオントロジ**（core ontology）と，アプ

リケーションおよびアプリケーションのドメインに依存して実装される**アプリケーションオントロジ**（application ontology）がある．これらを総称して**イベントオントロジ**（event ontology）と呼ぶ．イベントオントロジの詳細については 3 節で述べる．

2.4 フィルタリング問合せ

前述のようにオントロジを用いて記述された複合イベント指定は，CEP マネジャに与えられる．CEP マネジャは，与えられた複合イベント指定をもとに，実際にどのようにイベント検出を行うかをプランニングし，**プリミティブイベント問合せ**（primitive event query）および **CEP 処理プラン**（CEP processing plan）を構築する．プリミティブイベント問合せを略して **PE 問合せ**（PE query）と呼ぶ．プランニング処理では，複合イベント指定に含まれている複合イベントをプリミティブイベントに分解することにより，入力ストリームに対して適用すべき処理を抽出する．PE 問合せは，RDF ストリームに対して実行される連続的問合せであり，SPARQL をベースとした言語で記述される．PE 問合せは，**RDF リポジトリ**（RDF repository）内に格納されている永続的な RDF データ，イベントストア内のイベントデータ，外部の情報源なども参照して構築される．

2.5 PE エンジン

構築されたイベント問合せは**プリミティブイベントエンジン**（primitive event engine）に渡される．プリミティブイベントエンジンを略して **PE エンジン**と呼ぶ．PE エンジンの役割は，入力として得られる RDF ストリームから**プリミティブイベント**（primitive event）を抽出することである．入力には，ユーザから必要とされているイベント以外のイベントも多数含まれうるため，フィルタリング処理が適用される．検出されたプリミティブイベントは，逐次 CEP エンジンに送られる．後述するように，PE エンジンは C-SPARQL [3] に準じた問合せ処理能力を有するものとする．

2.6 CEP エンジン

CEP エンジン（CEP engine）は，ストリーム的に送付されるプリミティブイベントを受け取り，処理プランにより指定された処理を実施する．一つには，複合イベントの抽出であり，新たなプリミティブイベントをもとに複合イベントの導出処理をトリガーし，処理する．もう一つの役割は，処理プランにより指定された複合イベントおよびプリミティブイベントの**イベントストア**（event store）への蓄積処理である．また，古くて必要なくなったイベントベース中のイベントを削除する処理なども CEP エンジンの役割となる．

2.7 ユーザレベルの問合せ

ユーザおよびアプリケーションは，直接的に，もしくは API を通じてイベント問合せを発行する．イベント問合せとしてはスナップショット問合せと連続的問合せがある．**スナップショット問合せ**（snapshot query）は，イベントストアに現在蓄積されているイベント集合に対して適用される問合せである．一方，**連続的問合せ**（continual query）は，CEP エンジンにより継続的にモニタリングされるイベント（複合イベント指定に基づく）に対し，連続的に適用される問合せである．これらは，CEP エンジンにより実行処理される．

問合せ処理の具体的なイメージについては，4 節で述べる．

3. イベントオントロジ

3.1 イベントベースにおけるオントロジの活用

1 節で述べたように，応用領域および個々のアプリケーションに応じて，異なる多様なイベントの概念が用いられる．時間情報を扱うことは共通しているものの，時間の扱いですら必ずしも一致していない．このような理由から，アプリケーション分野に適応したイベント処理機構を構築するには，柔軟なイベントの定義・利用が可能でなければならない．そこで本研究では，イベントやそれに関連する意味的な概念を拡張可能な形で実現するために，**オントロジ**（ontology）を活用する．現在のオントロジ技術はセマンティック Web 技術とも関連が深く，また，Linked Open Data (LOD) を活用するにも親和性が高い．複合イベントの記述において，外部の情報源として LOD を有効利用することも考えられる．

イベントオントロジという概念自体は新規のものではなく，たとえば [12] では，イベントオントロジの概念的モデリングを行っている．ただし，この研究は知識レベルでのイベントの表現に着目したものであり，精緻ではあるものの複合イベント処理などの処理の側面は考慮されていない．

3.2 事例研究：LBSN オントロジ

3.2.1 LBSN オントロジの目的

イベントオントロジに関する事例研究の一環として，本研究グループでは，対象ドメインを**位置に基づくソーシャルネットワーク**（location-based social network, LBSN）とした，ドメインレベルのオントロジの構築を進めている [21], [22]．**LBSN オントロジ**（LBSN ontology）と呼ぶこのオントロジでは，LBSN の各種アプリケーションに見られるような時空間的なイベントや，移動するユーザ間の連携などを表現するための基盤となるフレームワークを示すことを目的としている．以下では，LBSN オントロジの概略について述べる．

3.3 LBSN オントロジの概要

2 節で述べたように，本システムでは，オントロジをコアオントロジとアプリケーションオントロジに分けている．LBSN オントロジも，一般に LBSN に共通する要素をまとめた **LBSN コアオントロジ**と，個別のアプリケーションもしくはアプリケーションドメインに対して構築される **LBSN アプリケーションオントロジ**に分けて考える．それぞれ，本システムのコアオントロジ，およびアプリケーションオントロジに包含されるものと考える．

LBSN オントロジを設計する上での方針は，関連する領域で既に提案されているオントロジを有効に活用するということである．LBSN を考える上で基本となる時空間情報に関する知識・語彙については，2 次元空間情報の表現のために Open Geospatial Consortium (OGC) と ISO が策定した標準である **Simple Features** [19] を，まず基盤として想定する．Simple Features は，地理・空間の概念を表現するために GeoSPARQL [4], [7] に取り込まれており，LBSN オントロジにおいて活用するのに妥当なものであると考えられる．また，時間に関する基本概念を表現するためには **Time Ontology** [18] を使用する．Time Ontology は LBS などに関するオントロジ構築のプロジェクトでしばしば活用されている．一方，LBSN におけるソーシャルネットワークに関する部分については，友達関係を表現するために **FOAF** (Friend of a Friend) [5] を利用する．

ただし，既存のオントロジの組合せだけでは LBSN において出現する多様な概念を表現することはできない．たとえばモバイルユーザに対する訪問すべき店舗の推薦といった応用を考えた場合，ユーザの位置情報などだけでなく，店舗の位置情報や種別を考える必要があり，また，ユーザと店舗の間の推薦に関する知識も表現する必要がある．これらは既存のオントロジには含まれない部分であるため，アプリケーションオントロジにおいて，既存のオントロジを用いて体系を構築していく必要がある．

3.4 イベントに関する知識の表現

ここで，本研究の重要なポイントであるイベントの概念について説明する．イベントの表現については，オントロジ記述の枠組みとして RDF を使用する一方，一般的なオントロジの記法からは逸脱するがルールベース的な表現も用いてイベント記述を行う．これは，問合せ処理におけるデータベース的な処理とも関係するものであり，その詳細については次節で述べる．ここでは LBSN を想定して，例をもとに考え方を説明する．

まず，イベントには，それ以上分解できないような**プリミティブイベント**（primitive event）が存在すると考える．これはオントロジのクラス階層としては `Event` クラスのサブクラスである `PrimitiveEvent` として位置づけられる．一例として，「時刻 T において人物 P が位置 L に存在した」というプリミティブイベントを考え，これを

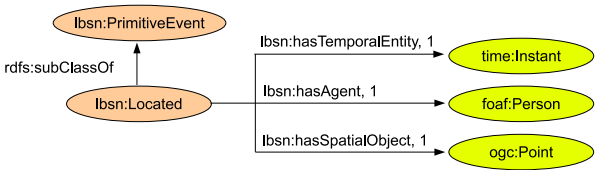

図 2 `Located` クラス

$$Located(P, L, T)$$

という述語で表現する．実際には，**図 2** で示すように，*Located* 述語は RDF を用いて `PrimitiveEvent` のサブクラスの `Located` として定義する [22].

次に，**複合イベント**（complex event）の定義例を示す．2 人の人物がある時点で近くにいるという，LBSN においてよく使用されるイベントを `ClosePeople` と呼び，以下のようなルールを用いて定義する [21].

$$\begin{aligned} &ClosePeople(P_1, P_2, L_1, L_2, T_1, T_2) \\ &\quad := Located(P_1, L_1, T_1) \wedge Located(P_2, L_2, T_2) \\ &\qquad \wedge NearLoc(L_1, L_2) \wedge NearTime(T_1, T_2) \end{aligned}$$

ここで，$NearLoc(L_1, L_2)$ および $NearTime(T_1, T_2)$ はプリミティブイベントではなく，それぞれ位置と時刻が近い場合に真になる述語である．これら二つの述語は，LBSN でしばしば用いられる概念であるため，LBSN コアオントロジに含めるが，個々のアプリケーションに応じて実体化することを想定する．距離が近い，時間が近い，という概念は，アプリケーションドメインおよび個々のアプリケーションに依存する度合いが強いため，拡張性により対応する．

複合イベントからさらに複合イベントを抽出することも考えられる．以下の *CloseFriends* は，友人どうしである 2 名がある時点で近くにいたというイベントを定義する．

$$\begin{aligned} &CloseFriends(P_1, P_2, L_1, L_2, T_1, T_2) \\ &\quad := ClosePeople(P_1, P_2, L_1, L_2, T_1, T_2) \\ &\qquad \wedge Friends(P_1, P_2) \end{aligned}$$

ここで *Friends* は 2 人が友人であるときに真となる述語であり，応用ドメインに応じて具体的に定義される．デフォルトとしては FOAF を用いることを想定する．

次の例は，複合イベントの定義において RDB 等に格納されたデータを活用する例である．$NearbyShop(P, S)$ は，ある人 P が店舗 S の近くに来たことに対応するイベントである．

$$\begin{aligned} &NearbyShop(P, S) \\ &\quad := Located(P, L_1, T) \wedge Shop(S, L_2, [T_1, T_2]) \\ &\qquad \wedge NearLoc(L_1, L_2) \wedge T \in [T_1, T_2] \end{aligned}$$

ここで，$Shop(S, L_2, [T_1, T_2])$ は，店舗 S が場所 L_2 にあり，その営業時間が $[T_1, T_2]$ であることを表している．

以上のように，ルールを用いて宣言的に複合イベントを定義することが本アプローチの特徴となっている．上記の

ようなアプローチでの複合イベントの表現能力はさほど強力なものではない．イベントの表現能力の拡張に関しては 5 節で述べる．

3.5 議論：行動オントロジについて

本プロジェクトでは，センサから取得した情報などに基づいてユーザの行動を取得し，行動履歴データとして蓄積・活用することについても研究を進めている [14]．このような行動認識の分野においても，アプリケーションや応用領域に応じてさまざまな概念が用いられることから，オントロジの活用が重要な役割を果たしうる．[13] では，行動認識のための広範囲なサーベイが行われており，現状の取り組みについての分析が行われている．多くの試みが存在するものの，これまでの行動オントロジ研究は，本研究のようにイベント処理を目的としたものではなく，行動に対するラベル付けをより高度化しようという立場に立っているものが多く，この点が本研究とは大きく異なるものとなっている．

4. 問合せ処理

イベントベースにおける問合せ処理は，2 節で述べたように，PE エンジンと CEP エンジンにおける処理の二つで分担する形で実行する．以下では順にその概要について述べる．

4.1 PE エンジンにおける問合せ処理

この節は，[16], [17] で述べた内容を踏まえて拡張したものである．**図 3** に，モバイルユーザが施設等の POI（Point of Interest）にチェックインしたことを示すイベントの例を示す．これは RDF の N-Triples 形式に基づくデータであり，チェックインイベント（Checkin1）には，時刻（Instant1），ユーザ名（User1），施設情報（Feature1）が含まれる．施設情報には空間上の点（Point1）が対応する．時刻や点の情報は，それぞれ対応するオントロジに基づいて記述する．図 1 に示したイベントベースの入力である RDF ストリームには，たとえばこのようなエントリが送付されてくる．

PE エンジンでは，入力の RDF ストリームに対し連続的な問合せを実行する．ここでは，フィルタリング処理が行われる例として，「あるユーザ A の近くにモバイルユーザがチェックインした」というプリミティブイベントを検出することを考える．タプル表現では，このイベントを

$$CheckIn\_Nearby(\text{“A”}, P, T, F)$$

と表記する．“A” はユーザ A を示す文字列，T はチェックイン時刻，P はチェックインした人，F はチェックイン地点を表す地理的特徴（geometric feature）である．図 3 の RDF データからこのプリミティブイベントを検出する問合せが**図 4** である．

```
Checkin1  rdf:type             lbsn:checkin;
          lbsn:hasTime         Instant1;
          lbsn:hasAgent        User1;
          lbsn:hasFeature      Feature1.
Instant1  rdf:type             time:Instant;
          time:inXSDDateTime   "2014-06-30T15:00:00"
                                       ^^xsd:dateTime.
Feature1  rdf:type             lbsn:Bakery;
          lbsn:name            "Motoyama Bakery";
          geo:hasGeometry      Point1.
Point1    rdf:type             geo:Point;
          geo:asWKT            "POINT(35.163, 136.962)"
                                       ^^sf:wktLiteral.
```

図 3 チェックインイベントの例

```
REGISTER Query CheckIn_Nearby
CONSTRUCT {
 ?checkin  rdf:type         lbsn:checkin;
           lbsn:hasTime     ?time;
           lbsn:hasAgent    ?person;
           lbsn:hasFeature ?feature.
}
FROM STREAM  <http://lbsn.org/checkin.trdf>
                      [RANGE 30m STEP 5m]
WHERE {
 ?checkin  rdf:type         lbsn:checkin;
           lbsn:hasTime     ?time;
           lbsn:hasAgent    ?person;
           lbsn:hasFeature ?feature.
 ?feature    geo:hasGeometry ?geo.
 BIND(geof:buffer(CURLOC_A, 200, uom:meter) as ?buff).
 [a geo:Geometry ; geo:asWKT ?buff]
           geof:sfContains ?geo.
}
```

図 4 PE 問合せ CheckIn_Nearby

この問合せは，SPARQL を RDF ストリームに対する連続的問合せに拡張した **C-SPARQL** [3] の構文を用いている．FROM STREAM 句では，チェックインイベントの RDF ストリームを取得し，[RANGE 30m STEP 5m] で，30 分の幅で 5 分毎でスライドする時間窓を設定している．近くにいるかどうかは，ユーザの現在地から半径 200m の範囲を表す ?buff を作成し，その範囲に友人が存在しているかを geof:sfContains という **GeoSPARQL** [4] の述語を使用して判断している．CURLOC_A は ユーザ A の現在位置（問合せの評価時に動的に変化する）の略記とする．

CEP マネジャにおける問合せのプランニングにおいては，与えられた複合イベント指定 $CheckIn\_Nearby(\text{“A”}, P, T, F)$ に対してどのように図 4 の PE 問合せを構築するかが問題となる．本研究では，現状では，各プリミティブイベントに対応する PE 問合せのテンプレートを準備することを考えている．図 4 において，ユーザ A の位置情報に対応する CURLOC_A は，実際にはテンプレート問合せにおける変数に A の位置を埋め込んだものとする．また，BIND 句に出てくる 200 という

定数は「近い」という概念の閾値を表す定数であるが，これもテンプレート上では変数であり，問合せプランニング時に，該当するオントロジから情報を抽出して具体化される．一方，[RANGE 30m STEP 5m] という時間窓は，複合イベント指定には特に依存しておらず，システム側で設定されたものとする．もし複合イベント指定に時間窓に影響するような条件等が含まれている場合には，時間窓の設定も変わってくる可能性がある．

複合イベント指定からどのように PE 問合せで処理できる部分を抽出するかを判断するのも CEP マネジャの仕事である．たとえば，

$$Friend\_CheckIn\_Nearby(P_1, P_2, T, F) := CheckIn\_Nearby(P_1, P_2, T, F) \wedge Friends(P_1, P_2)$$

という複合イベントの定義を具体化することで，$Friend\_CheckIn\_Nearby$(“A”, P, T, F) という複合イベント指定が CEP マネジャに与えられたとする．このような場合，$CheckIn\_Nearby$ については図 4 のように，データストリームをシーケンシャルにスキャンすることで処理が可能であるが，$Friends$ までを評価しようとすると静的な情報源（FOAF のオントロジ）との結合処理が発生してしまう．そのため，このような場合には，$CheckIn\_Nearby$ のプリミティブイベントの検出のみをストリーム的にこのフェーズで実施し，$Friends$ の条件適用は CEP エンジンによる処理に任せるものとする．このようにして，PE エンジンではデータストリームに対する効率的な実行処理を行う．

以上のような処理により，プリミティブイベントが，フラットなタプル構造を持った RDF データとして検出される．2 節で述べたように，これが CEP エンジンに送られる．

4.2 CEP エンジンにおける問合せ処理

4.2.1 イベント蓄積・廃棄のポリシー

CEP エンジンには，プリミティブイベントを逐次的に受け取り処理を進める．プリミティブイベントには，

(1) CEP エンジンで処理を行った後（例：新たな複合イベントを生成），即座に廃棄できるもの
(2) 明示的に削除されるまで半永久的に保持しなければならないもの
(3) 時間窓の指定に応じて，一定期間保存しなければならないもの

の三通りがある．CEP マネジャにおける問合せプランニングの時点において，与えられた複合イベント指定に応じて，どのプリミティブイベントを上記のどのポリシーで管理するかを決定する必要がある．基本的には，複合イベント指定を与えるユーザが，どのプリミティブイベントを永続的，あるいは時間窓で保存するかを明示的に指定することになる．ただし，ユーザおよびアプリケーションから与えられるイベント問合せの時間窓などの指定によっては，より長い時間保持するなどの必要が出てくる可能性がある．また，問合せ処理の都合上，イベントを保持しておく場合も発生すると考えられる．プリミティブイベント同様，複合イベントについても同様に 3 種類のポリシーが考えられる．

4.2.2 RDB を用いた問合せ処理

CEP エンジンにおける問合せ処理は，リレーショナルベースシステムを用いて実現する．イベントがフラットなタプル構造であることから，各イベントの種類をリレーションに対応づける．たとえば，3 節で示した $ClosePeople$ という複合イベントは，

```
SELECT *
FROM Located x, Located y
WHERE NearLoc(x.loc, y.loc) AND
      NearTime(x.time, y.time)
```

という SQL 問合せに対応する．ただし，NearLoc, NearTime は，それぞれの述語を実装するユーザ定義関数である．オントロジにおけるこれら述語の定義が単純であり，たとえば $NearLoc$ が 200m 以内，$NearTime$ が 5 分以内であれば真となるような条件であるなら，さらに簡単化でき，

```
SELECT *
FROM Located x, Located y
WHERE spatial_dist(x.loc, y.loc) <= 200 AND
      time_diff(x.time, y.time) <= 5
```

といった問合せ記述も可能である．しかし一般には，述語の評価の際にオントロジを参照することも必要となることがあり，ユーザ定義関数が重い述語になってしまう可能性は存在する．

さらに，上の問合せがビュー ClosePeople として登録されると，それを用いて 3 節で示した $CloseFriends$ 複合イベントは

```
SELECT *
FROM ClosePeople p
WHERE Friends(p.person1, p.person2)
```

と記述できる．ここでは FOAF による友人か否かの判断をユーザ定義述語のように表現している．別の考え方としては，Friends に対するリレーションを構築し，ClosePeople と Friends の結合処理として表現することも考えられる．

リレーショナルデータベースの枠組みを用いる利点には，問合せ処理の効率化だけでなく，見通しが立てやすいことがある．特に，複合イベントを**実体化ビュー**（materialized view）としてイベントストア上で表現することは有効であ

ると考えられる．3 節で示したような複合イベントの記述は，いわゆる **SPJ 問合せ**（select-project-join query）に対応しており，インクリメンタルな実体化ビューの管理が容易に実現できる．たとえば上記の例において `ClosePeople` 実体化ビューに対してタプル（イベント）が追加・削除されたとき，それが `CloseFriends` のタプルに与える影響（差分）をインクリメンタルに求めることができる．これにより，新たに変化した情報のみをユーザやアプリケーションに伝えることが可能となる．

5. 議論

これまで，イベントベースの概念とその構想について述べてきたが，今後開発すべき多くの課題が存在する．ここでは重要なトピックを取り上げて議論する．

5.1 高度なイベント記述

3 節で示した複合イベントの例は，基本的にはリレーショナルデータベースの SPJ 問合せに変換できる単純なものであった．ただし，ユーザ定義の述語が入ってくる点は大きく異なっている．リレーショナルデータベースが効率的に処理できる範囲で複合イベント処理が記述されていることから，その問合せ処理には RDBMS の能力が活用できるという利点があった．しかし，より高度なイベントを表現したいという要求もある．

例として，以下の複合イベントを考える．

$$StayInRoom10(P,R,T) := (P,R,T'),$$
$$\text{if } \forall T' \in [T, T+10], Located(P,R,T')$$

これは，時刻 T から 10 単位時間の間，人物 P が同じ部屋 R 内にいたときに検出されるイベントである．このような問合せを蓄積されたイベント集合上でリレーショナルデータベースの問合せとして実行することは難しいことではない[*1]．しかし，この複合イベント指定をストリームデータに適用するためには，PE 問合せの導出手法を新たに開発する必要が出てくる．また，PE エンジンに，単純なストリーム処理のみならず，集約演算など高度なストリーム処理機能を持たせるよう拡張する必要も発生する．

複合イベントを宣言的に記述ということが本研究のアプローチの特徴であるが，表現能力と処理能力のトレードオフをとることが重要な課題となる．

5.2 グラフ構造の複合イベントの生成

本研究のアプローチでは，プリミティブイベントを検出した時点でイベントはフラットなタプルになり，その後の複合イベントの導出においても出てくるデータはフラットなタプル構造のデータである．リレーショナルデータベースを基盤とすることにより，イベントの高速な処理や，よりすっきりした蓄積・廃棄処理が可能になると考えられるが，出力としての複合イベントとして，グラフ構造の RDF 形式の表現がほしいという要求も考えられる．

このような要求については，リレーショナルデータベース上の XML ビューに対する問合せ技術 [9], [15] が参考になるのではないかと考えられる．これらの研究では，データ本体はリレーショナルデータベースにあるが，問合せはその上に構築された XML ビューに対して出されることを想定している．本研究におけるイベントストアはリレーショナルデータベースに基づいており，その上にグラフ構造の RDF ビューを構築するなら，同様のフレームワークが活用できる．

6. まとめと今後の課題

本稿では，我々の研究グループで研究開発を進めているイベントベースについて，その背景，構想，基本的なアプローチについて述べた．今後はまず，参照アーキテクチャをもとに，個々の要素技術を洗練することを進める．本プロジェクトの目標の一つとしては，LBSN などを対象としてアプリケーションに特化した機能をより盛り込み，実世界で要求される機能を実現するためのフレームワークを構築することがある．また，行動認識システムなどと連携して，多数のユーザの行動の履歴を蓄積管理し，マイニング技術などと連携することも考えられる．

謝辞 本研究の一部は，科学研究費（25280039, 26540043）および JST COI STREAM プログラムによる．

[*1] ただし，同じ部屋に長い時間人が留まった場合は，時刻ごとに多くのイベントが検出されてしまうという問題はある

参考文献

[1] Aggarwal, C. C.(ed.): *Data Streams: Models and Algorithms*, Springer (2006).
[2] Arasu, A., Babu, S. and Widom, J.: The CQL Continuous Query Language: Semantic Foundations and Query Execution, Technical report, Stanford University (2003).
[3] Barbieri, D. F., Braga, D., Ceri, S., Valle, E. D. and Grossniklaus, M.: Querying RDF Streams with C-SPARQL, *SIGMOD Record*, Vol. 39, No. 1, pp. 20–26 (2010).
[4] Battle, R. and Kolas, D.: Enabling the Geospatial Semantic Web with Parliament and GeoSPARQL, *Semantic Web*, Vol. 3, No. 4, pp. 355–370 (2012).
[5] Brickley, D. and Miller, L.: FOAF Vocabulary Specification 0.99, `http://xmlns.com/foaf/spec/` (2014).
[6] Chakravarthy, S. and Jiang, Q.: *Stream Data Processing: A Quality of Service Perspective - Modeling, Scheduling, Load Shedding, and Complex Event Processing*, Springer (2009).
[7] Consortium, O. G.: GeoSPARQL - A Geographic Query Language for RDF Data, `http://www.opengeospatial.org/standards/geosparql`.
[8] Cugola, G. and Margara, A.: Processing Flows of Information: From Data Stream to Complex Event Processing, *ACM Computing Surveys*, Vol. 44, No. 3 (2012).
[9] Fernandez, M., Morishima, A. and Suciu, D.: Efficient

Evaluation of XML Middle-ware Queries, *ACM SIGMOD*, pp. 103–114 (2001).
[10] Garofalakis, M., Gehrke, J. and Rastogi, R.: *Data Stream Management: Processing High-Speed Data Streams*, Springer (2007).
[11] Golab, L. and Özsu, M. T.: *Data Stream Management*, Morgan & Claypool (2010).
[12] Guizzardi, G., Wagner, G., de Almeida Falbo, R., Guizzardi, R. S. and ao Paulo A. Almeida, J.: Towards Ontological Foundations for the Conceptual Modeling of Events, *ER Conf.*, LNCS, Vol. 8217, pp. 327–341 (2013).
[13] Rodríguez, N. D., Cuéllar, M. P., Lilius, J. and Calvo-Flores, M. D.: A Survey on Ontologies for Human Behavior Recognition, *ACM Compting Surveys*, Vol. 46, No. 4 (2014).
[14] 佐々木勇和，簗井美咲，高橋正和，杉浦健人，石川佳治：行動オントロジによるセンサデータからの複合イベント検出について，第 13 回情報科学技術フォーラム（FIT 2014） (2014).
[15] Shanmugasundaram, J., Kiernan, J., Shekita, E., Fan, C. and Funderburk, J.: Querying XML Views of Relational Data, *VLDB* (Apers, P. M. G., Atzeni, P., Ceri, S., Paraboschi, S., Ramamonaharao, K. and Snodgrass, R. T., eds.), pp. 261–270 (2001).
[16] 高橋正和，簗井美咲，稲葉鉄平，石川佳治：オントロジーを利用したイベント処理システムの提案，情報処理学会第 76 回全国大会 (2014).
[17] 高橋正和，簗井美咲，佐々木勇和，石川佳治：RDF ストリーム上での複合イベント検出，第 13 回情報科学技術フォーラム（FIT 2014） (2014).
[18] W3C: Time Ontology in OWL, `http://www.w3.org/TR/owl-time/`.
[19] Wikipedia: Simple Features, `http://en.wikipedia.org/wiki/Simple_Features`.
[20] Wu, E., Diao, Y. and Rizvi, S.: High-Performance Complex Event Processing over Streams, *ACM SIGMOD*, pp. 407–418 (2006).
[21] 簗井美咲，高橋正和，稲葉鉄平，石川佳治：LBSN オントロジーの設計，情報処理学会第 76 回全国大会 (2014).
[22] 簗井美咲，高橋正和，佐々木勇和，石川佳治：LBSN オントロジの構築，第 13 回情報科学技術フォーラム（FIT 2014） (2014).