RFID 書込みによるユビキタス環境DBサーバの負荷低減手法

古賀　浩史†　　山田　佑樹†　　富井　尚志††

† 横浜国立大学大学院環境情報学府情報メディア環境学専攻
†† 横浜国立大学大学院環境情報研究院
〒 240-8501 横浜市保土ヶ谷区常盤台 79-7
E-mail: †{d09hc020, d09hc044}@ynu.ac.jp, ††tommy@ynu.ac.jp

あらまし　電子タグ付き空間ではタグ検知をトリガに DB サーバに問合せを行うことで，実空間情報を取得・蓄積して利用する．しかし，タグが大量に埋め込まれた環境ではその為の DB アクセスが頻発する．問合せの解はタグ毎に一定となる場合もあるが，それでもリアルタイムな実空間情報の取得する為にタグ検知の度に DB アクセスを行う．この結果，サーバリソースの浪費に繋がる．我々はこの問題に対し RFID タグの記憶領域を用いたサーバ負荷の低減手法を提案する．提案手法では，タグの記憶領域を頻発する問合せの解を格納する環境内に分散したキャッシュメモリとして活用する．その結果，実空間情報の取得に伴う問合せ数を減らし，サーバ負荷を低減できると考えられる．本稿では提案手法による実証実験を行い，その実現性と有用性を示した．

キーワード　ユビキタスコンピューティング，時空間 DB，RFID，RFID ユーザメモリ

Load Reduction Method for Ubiquitous Environment DB Servers by RFID Writing

Hiroshi KOGA†, Yuki YAMADA†, and Takashi TOMII††

† Department of Information Media and Environment Sciences, Graduate School of Environment and Information Sciences, Yokohama National University
†† Research Institute of Environment and Information Sciences, Yokohama National University
79-7 Tokiwadai, Hodogaya-ku, Yokohama 240-8501 Japan
E-mail: †{d09hc020, d09hc044}@ynu.ac.jp, ††tommy@ynu.ac.jp

Abstract　In RFID-tagged space, we can acquire object locations and users' actions. These data are stored and used for user assistance. In generally such a space, queries are sent to database server for getting information about detected tags when readers detect tags. However, in the case of the environment where thousands of tags were embedded, access concentration problems arise when a large quantity of tags are detected in a short time. In this paper, to solve the problem, we propose a method using the RFID tag memory as a cache memory. In RFID tag memory, we write the results of queries triggered by detecting tags. By using the method, unnecessary DB accesses are prevented and consequently a load of the DB server is reduced. We implemented this method after discussing implementation issues for it, and evaluated the effectiveness of it by experiments.

Key words　Ubiquitous Computing, Spatio-temporal DB，RFID, RFID Tag Memory

1. はじめに

近年，センサ・計算機技術の発達によりユビキタス環境が現実的になってきた．ユビキタス環境では，実空間情報を利用することでユーザに対してより高度な支援が可能になる．そこで実空間情報の取得方法として，一意の識別性を持つ RFID タグを利用した電子タグ付き空間が考えられている [1].

電子タグ付き空間とは，身の回りのあらゆる物体・領域に電子タグ (RFID タグ) が埋め込まれた空間であり，タグが持つ Unique ID(UID) とタグが貼付された事物とがデータベース (DB) 内で関連付けられて管理されている．環境に埋め込まれたタグを RFID リーダにより検知していくことで実空間における物体の位置情報や移動履歴を，また加速度センサや照度センサ等のセンサ機器と組み合わせることで利用者の行動を取得・蓄積することができる．この蓄積された情報を利用することで，より高度で多様なユーザ支援が可能となる．そしてリアルタイムな情報を反映したサービスを提供する為には，タグ検知の度に DB サーバへ問合せを行い，実空間情報を取得・蓄積をする必要がある．

一般的に電子タグ付き空間では，タグ検知をトリガにタグが示す事象やそれに付随する情報を取得する為のクエリが発行される．発行したクエリの解を DB サーバから得ることで単なるセンサデータを利用者にとって有益な実空間情報として抽出

し取得していく．従って，環境内に埋め込まれたタグ枚数や空間利用者数の増加に伴いタグの検知頻度が高くなると，DB アクセスも高頻度で生じることになる．しかし，タグ検知により生じる問合せは，環境毎に問合せ内容が決まっており，その解もタグ毎に一定となる定型質問クエリとなる場合がある．それでもリアルタイムな実空間情報を取得する為にタグ検知の度に DB アクセスを行う．その為，環境内に埋め込まれたタグ枚数や空間利用者数の増加に伴いタグの検知頻度が高くなると，DB アクセスも高頻度で生じることになる．このように，DB から取得する解が一定であるにも関わらずタグ検知の度に DB サーバに問合せを行うことは，サーバリソースの浪費に繋がる．サーバリソースとは，CPU やメモリ，ハードディスクなどのことを言うが，本研究では特に CPU の使用率を指す．また，電子タグ付き空間の DB ではタグ ID と実空間の事物を関連付けて管理する．従って，DB に格納されるデータの規模は，埋め込まれたタグ枚数や実空間に存在する事物数の増加に伴い爆発的に増大していく．その為，膨大な情報量やそれに対する処理量からは，単にサーバ負荷だけでなくサーバの消費電力の増大が問題となる．近年では省電力化に対する関心の高まりから，サーバの省電力化に関する様々な研究 [2] [3] も行われている．

そこで，我々は RFID タグの記憶領域 (RFID メモリ) を利用した電子タグ付き空間における DB サーバの負荷低減手法を提案する．RFID タグは個々のタグが僅かながらも記憶領域を持ち，これを利用した様々な研究 [4] [5] [6] が行われている．Mamei らは，RFID メモリ自体に情報を載せることでユビキタス環境における物体追跡等を可能にするシステムを提案をしている [5]．本手法の特徴は RFID メモリを頻発するクエリの解を格納する分散メモリとして活用する点にある．環境内に分散した RFID メモリをキャッシュメモリのように利用することで，実空間情報の取得による DB への問合せをローカルに解決でき，DB サーバの負荷を低減できる．本稿では，提案手法をオフィス形式の電子タグ付き空間に適用・実装し，サーバの負荷状態を示す CPU 使用率と併せて消費電力の観点から提案手法に対して実証実験を行うことで有用性と実現性を評価した．

2. 関 連 研 究

2.1 電子タグ付き空間

本稿では，RFID タグが物体や領域に貼付され，リーダでそれらを読取ることにより様々な情報を取得できる空間のことを電子タグ付き空間と呼ぶ．例えば，物流の現場では，製造物にタグを貼り，タグに製造日時・流通経路等の情報を関連付けることでトレーサビリティの向上を目指す取り組みが行われている [7]．医療分野では，患者にタグ付きのリストバンドを付与することによって患者とタグ ID を結び付け，更に投与薬物の種類，投与時間，疾患情報等を患者 ID と関連付けることで的確で迅速な処置が可能になる [8]．また，Tagged World [1] では，部屋内の様々な場所に配置された RFID タグと人が携帯するリーダによって人の行動を認識し，その意図を推定することで行動支援等を行うシステムが研究・提案されている．

2.2 概念共有環境 CONSENT

実空間内における物体の位置情報や状態，利用者の行動といった情報をセンサ値から取得する研究は多く行われている．しかし，センサ値だけではそれが何を表しているのかという意味情報，すなわち環境の「こうあるべき」という状態や利用者が行動に込めた「こうであるつもり」といった情報（以下，概念）を抽出することは困難である．そこで我々はこれまで，電子タグ付き空間を代表としたユビキタス環境からのセンサ値と上述した概念情報を統合したユビキタス環境 DB として概念共有環境 CONSENT（以下，CONSENT）を提唱してきた [9]．刻々と変化する環境内の情報をユビキタス環境 DB で共有する為の実空間情報の取得方法として，空間のあらゆる物体・場所に RFID タグを貼付し，利用者に RFID リーダを装着することにした．また RFID タグには，安価で半永久的な利用が可能な点と検知範囲がある程度狭い方が利用者が触れた物体のみを検知し易い点からパッシブ型の HF 帯 RFID タグ (ISO15693) を用いた．CONSENT では利き手首にリーダを装着した利用者により日常生活の中から大量のタグ検知反応が発生する．発生したタグ検知列を基にし，その度に DB サーバに問合せを行うことで物体の位置情報・移動履歴や利用者の行動をリアルタイムに取得・蓄積していく．そして蓄積された情報を基に利用者への支援を行う．

2.3 RFID の記憶領域

RFID タグは一意の識別性や非接触で複数枚の同時読取りが可能な点，またパッシブタグの場合半永久的に利用できるといった特徴を持つ．更に，個々のタグが僅かながら書き換え可能な記憶領域を持っている．通信距離や記憶容量は規格等により異なるが，ISO15693 規格の RFID の場合数 cm から数十 cm の通信距離があり，ユーザが書き換え可能なメモリ容量は数十 Byte から数 KByte となっている．Mamei らは環境内にばら撒かれた RFID タグの記憶領域を分散記憶として利用することで，空間内の移動エージェントを協調させるシステムを設計・構築した [4]．同様に環境内に分散させたタグに書込みを行い，マーキングをしていくことで行方不明になった物体の追跡を可能にする研究も行われている [5]．しかし，タグへの書込みにはリーダとの通信中にタグが検知範囲外に出ると通信が中断されるといった不安定性の問題がある．そしてこの問題はタグに書込まれたデータの不確実性にも繋がる．Ryu らはこの問題に対し，Continuous Query Index(CQI) スキーマにより書込み途中のタグを検出し，その処理を再開し完遂させるモデルを提案・実装した [6]．このようにタグの記憶領域を効率的に利用しようとする研究が盛んに行われている．

3. DB サーバ負荷低減手法の設計

3.1 問 題 設 定

電子タグ付き空間ではタグ検知の度に DB サーバに問合せを行うことでリアルタイムな実空間情報を得る．しかし，その問合せから得る解は一定となる場合もある．特に更新頻度の低い情報に関しては，問合せによって得る解というのは一定であるが，それでもタグ検知の度に DB サーバに対してクエリを発行する．一方，空間内で利用されるタグ枚数や利用者人数の増加に伴いタグの検知頻度が上昇すると，タグ検知をトリガにした問合せも高頻度で生じることになる．その結果，得られる解は決まっているにも関わらず大量の DB アクセスが発生することになり，サーバリソースの浪費に繋がる．また，電子タグ付き空間における DB はタグ ID と実空間の事物を関連付けて管理する．従って，DB に格納されるデータ規模は，利用タグ枚数や実空間に存在するオブジェクト数の増加に伴い巨大化していく．このように爆発的増加をみせる情報量やそれに対する膨大な処理量から生じる問題は単にサーバ負荷だけではない．近年のプロセッサでは CPU の稼働状況が消費電力に与える影響性が高く，省電力化に対する社会的関心の高まりから膨大な情報量を処理するデータセンタ等における消費電力の増大が問題視されている．以上の問題からも，DB サーバの負荷を低減することは非常に重要であると言える．そこで，本研究ではタグ検知をトリガに生じる問合せの解を予め RFID メモリに書込むことを提案する．しかし，タグ検知時に生じる問合せとその解は環境に依存して異なる．2.2 節で述べた CONSENT でも，利用者間で概念の共有を可能にする有用なシステムである

と同時に上述したサーバ負荷の問題を内包する．従って本稿では CONSENT に適応して設計を行う．

3.2 CONSENT の基本モデル

本節では，2.2 節で述べた環境における空間利用者の行動取得を実現する DB モデルについて述べる．本研究では利用者の行動のアトミックな要素を以下のように定義する．

Action_EE：Action_EE とは，「誰が (Who)」「いつ (When)」「どこで (Where)」「何によって (What)」「何をした (How)」の要素からなる組合せである．(Where は無くてもよい)．

タグを検出したリーダの個体識別番号，タグの検知時間，検知タグから「誰が (Who)」「いつ (When)」「どこで (Where)」「何によって (What)」は容易に取得できる．「何をした (How)」については，タグの検知列と加速度センサを利用して人の振る舞いを推定する手法 [10] を用いた．

CONSENT の DB 構造の概念図を図 1 にを示す．CONSENT の DB では，センサデータと物体の持つ意味や Action_EE の意図等の概念的要素を統合して蓄積すると共に大量に発生するデータを効率的に活用する為，以下の三層構造モデルとした．

意味層：物体や Action_EE の意味情報をオントロジによって明示的・体系的に記述する．物体に関する概念を表す Thing Ontology，行動に関する概念を表す Action Ontology，そしてオントロジ間の関係を表す Relation Ontology から成り，各々が汎化されたクラスのノードから特化されたクラスのノードへ階層化された is-a 関係のクラス階層構造をとる．これらを参照する形で，空間内に存在する物体の意味やその物体が何をする為に使えるかという知識を表現できる．この層で定義したことは利用者全体に共通して言えることであり，各々の物体に対して何ができるかという意味を表わすことから意味層と呼ぶ．図 1 を例にとると，「机に参考書を置くことができる」という概念表現は，「Action Ontology：置く」から派生した「Action Ontology：参考書を机に置く」と「Thing Ontology：机」「Relation Ontology：Where」及び「Relation Ontology：What」「Thing Ontology：参考書」間の関係付けによって可能となる．ここで，ある物体に対して予め設定した可能な行動（Action）の候補群を Action_Ontology 候補と定義する．これらの設計はオントロジ編集者が適用する電子タグ付き空間に合わせて行う．

MM 層（マルチメディアデータ層）：環境内で利用されるタグデータ (事物に貼付されたタグの ID) や，Action_EE に付随するマルチメディアデータとして RFID タグの検知反応データ (検知タグや検知時間)，加速度センサの生データなどを蓄積する．これらは意味層とは独立して蓄積される．

EE 層 (存在エンティティ層)：実空間に存在する事物や利用者が実際に行った Action_EE の一つ一つをエンティティとして捉え，存在エンティティ（EE）と呼ぶ．EE 層では EE を意味情報及びマルチメディアデータと関連付けて蓄積する．この EE は，Thing Ontology のインスタンスを表わす Thing_EE と Action Ontology のインスタンスを表わす Action_EE とに大別される．Thing_EE は実空間に存在する物体の一つ一つを表し，これにより実空間上の各物体がどういった意味情報を持つのかが定義される．Action_EE は Action_Ontology 候補に基き利用者が実際に行った行動の一つ一つを表わし，これらを積み重ねることで利用者の行動履歴となる．また，EE は全て ID によって管理されており，特に物体に関するエンティティを物体 EEID と定義する．この物体 EEID と MM 層で蓄積されるタグ ID とは 1:n の関係をとる．これは，タグの検知漏れを可能な限りなくせるよう一つの物体に複数枚のタグを貼付する環境を想定している為である．

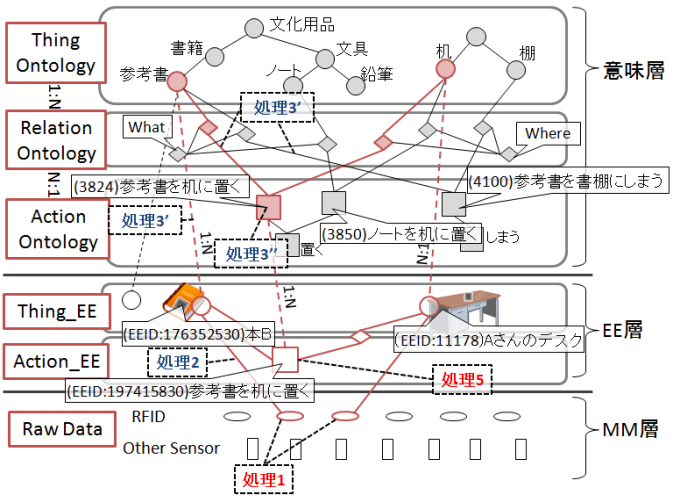


図 1　CONSENT における DB 概念図

以上で述べたオントロジ，マルチメディアデータ，存在エンティティの関連付けは全て ID によって行われており，これにより高度な蓄積及び検索を可能にしている．

3.3 実空間情報の取得による DB アクセス

実空間情報のうち利用者が行った Action_EE を取得する為に，タグ検知をトリガとして生じる DB アクセスについて述べる．Action_EE 取得の為の一連の処理を図 2 に示す．これらは利用者が装着したリーダによるタグの検知から，利用者の行動の最小単位である Action_EE を取得する為に必要な処理である．図 2 について図 1 を用いた具体例を示しながら説明していく．

処理 1：生データの挿入

検知タグが持つタグ ID 及びタグ検知時間を生データとして挿入する．

処理 2：検知タグが示す物体 EEID の問合せ

検知タグに関連付けられた実空間の物体を表わす物体 EEID を EE 層の Thing_EE から取得する．これはタグ ID が決まれば一意の解が得られる．図 1 では，タグ検知列から「A さんのデスク」と「本 B」を検出した．

処理 3：Action_Ontology 候補の問合せ

検出物体に関連付けられた Action_Ontology 候補を取得する．図 1 の場合，意味層において「Thing Ontology：参考書」と「Action Ontology：参考書を書棚にしまう」が「Relation Ontology：What」により関連付けられる (処理 3')．且つ「Action Ontology：参考書を書棚にしまう」が「Relation Ontology：Where」により「Thing Ontology：書棚」と関連付けられているならば，「参考書を書棚にしまう」というオントロジは「参考書」を What，「書棚」を Where として成り立つことを示す (処理 3")．つまり，図 1 では「参考書」と「机」の二つの物体間に共通して関連付けられた行動は「参考書を机に置く」のみとなる．このようにしてタグ検知列から Action_Ontology 候補を得る．

処理 4：実際に行われた Action_EE の推定

処理 3 により得た Action_Ontology 候補から他のセンサ類 (加速度センサや照度センサ等) による認識エンジンを用いて実際に行われた Action_EE を一意に決定する．本研究では，加速度を用いた行動推定手法 [10] を利用した．

処理 5：実際に行われた Action_EE の挿入

処理 4 から実際に行われた Action_EE を認識できれば，それを一つのエンティティとして EE 層の Action_EE に蓄積する．

以上の処理により，利用者が物体や場所に対して行った Ac-

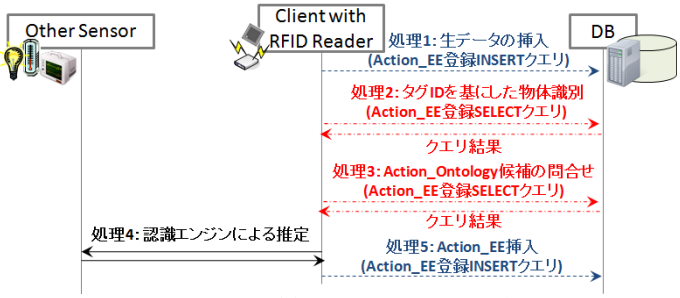
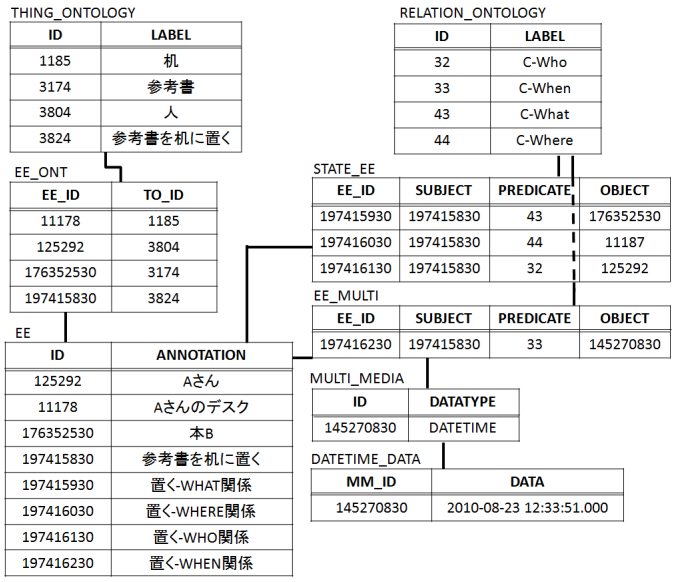


図 2　Action_EE 登録トランザクションの流れ



THING_ONTOLOGY

ID	LABEL
1185	机
3174	参考書
3804	人
3824	参考書を机に置く

RELATION_ONTOLOGY

ID	LABEL
32	C-Who
33	C-When
43	C-What
44	C-Where

EE_ONT

EE_ID	TO_ID
11178	1185
125292	3804
176352530	3174
197415830	3824

STATE_EE

EE_ID	SUBJECT	PREDICATE	OBJECT
197415930	197415830	43	176352530
197416030	197415830	44	11187
197416130	197415830	32	125292

EE_MULTI

EE_ID	SUBJECT	PREDICATE	OBJECT
197416230	197415830	33	145270830

EE

ID	ANNOTATION
125292	Aさん
11178	Aさんのデスク
176352530	本B
197415830	参考書を机に置く
197415930	置く-WHAT関係
197416030	置く-WHERE関係
197416130	置く-WHO関係
197416230	置く-WHEN関係

MULTI_MEDIA

ID	DATATYPE
145270830	DATETIME

DATETIME_DATA

MM_ID	DATA
145270830	2010-08-23 12:33:51.000

図 3　状態を表すインスタンス集合

tion_EE を DB に蓄積していく．ここで，意味層で予め定義された Action_Ontology 候補をインスタンス化し，Action_EE として DB に蓄積する際のモデルを図 3 に示した．図 3 では，2010-08-23 12:33:51 に A さん (EEID:125292) が A さんのデスク (EEID:11178) に本 B(EEID:176352530) を置く (EEID:197415830)[参考書を机に置く] という Action_EE のインスタンスを表わしている．このモデルにより利用者が行った Action_EE や実空間から得られた物体の状態といった情報を DB 化することが可能になる．

3.4　実空間情報の取得に伴う DB アクセスの削減対象

3.3 節で述べた図 2 の一連の処理はタグ検知の度に実行される．ここで，本稿では図 2 における処理 1 から 5 による DB アクセスを以下のように定義する．

Action_EE 登録トランザクション：図 2 で示した処理 1(生データの挿入) から処理 5(Action_EE 挿入) までの一連の処理に伴う DB アクセスを，タグ検知をトリガに発生する一件の Action_EE 登録トランザクションとする．

また，Action_EE 登録トランザクションにより生じる DB アクセスを以下の二種類に大別して定義する．

Action_EE 登録 SELECT クエリ (図 2 の赤一点鎖線)：Action_EE 登録トランザクションの処理 2 及び 3 で生じ，タグが示す物体 EEID 及び Action_Ontology 候補を取得する為に発行される．一方，CONSENT においてオントロジの更新が行われることは稀であり，この問合せによって得る解は DB で予め定義された一定の解となる．すなわち，この部分は RFID メモリに予め解を書込むことによって問合せをローカルに解決できれば DB 負荷の低減に繋がる．

Action_EE 登録 INSERT クエリ (図 2 の青破線)：Action_EE 登録トランザクションの処理 1 及び 5 にあたり，センサ値の生データと空間利用者の行動履歴を蓄積する為に発行される．これは実空間情報の蓄積にあたり必須となる部分である．

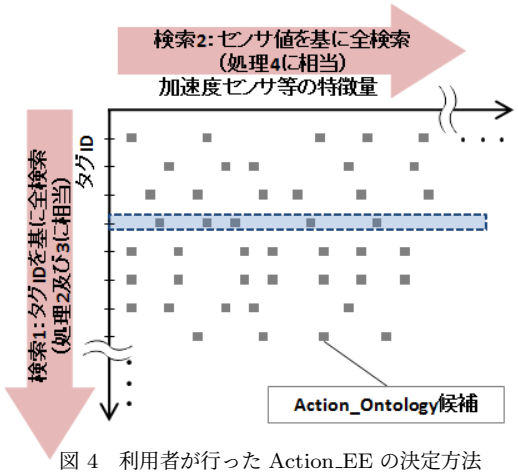


図 4　利用者が行った Action_EE の決定方法

そこで，本研究では RFID メモリに Action_EE 登録 SELECT クエリの解を書込むことで DB サーバ負荷を低減させる手法を提案する．ここで，検知タグを基に利用者が行った Action_EE を決定する方法を簡略化した図を図 4 に示す．縦軸は離散的なタグ ID 空間を，横軸は連続的な値をとる加速度センサ等の特徴量を表す．縦軸の目盛一つ一つは環境内で利用されているタグ ID を示し，個々のプロット点は Action_Ontology 候補を表している．つまり，図 4 の横一列に並んだプロット点はある一枚のタグに対応付けらている Action_Ontology 候補を指す．タグ検知から利用者が行った Action_EE を決定する方法について簡潔に述べる．まず検知タグ ID からタグ ID 空間に対して全検索を行い検知タグに対応する Action_Ontology 候補を抽出する (図 4 の検索 1)．次に抽出した Action_Ontology 候補に対して加速度センサから得た特徴量を基に検索を行い，利用者が実際に行った Action_EE に該当する Action_Ontology 候補を一意に決定する (図 4 の検索 2)．

DB サーバの負荷集中問題に対し，提案手法以外にも分散 DB として設計する方法や強力なクライアント PC にキャッシングすることでサーバ処理の一部を外部で行う等の解決策が考えられる．しかし，これらの手法では検知タグ ID を基に ID 空間の検索を行う必要があり，その検索はサーバでなくともどこかの情報端末上で必ず行われることになる．そして検索対象となる ID 空間が膨大になるとやはり検索によるコストが生じる．これに対し，提案手法では RFID メモリを読出すことでタグ検知と同時に検知タグに対応する Action_Ontology 候補 (図 4 の点線で囲まれた領域) が取得できるという利点がある．すなわち，通常ならば ID 空間と特徴量空間の二次元探索を要する Action_EE の決定を特徴量空間の一次元探索として扱うことができ，DB の物理スキーマ設計に依らず DB サーバ負荷の低減を可能にする．

3.5　RFID メモリを利用した実空間情報の取得

本研究では Action_EE 登録 SELECT クエリ (図 2 の処理 2・3) の解を予め RFID メモリに書込むことを提案する．処理 2 の解はタグが示す物体 EEID である．物体 EEID とタグ ID の関係は 1:n だが，一枚のタグが示す物体は一意に決まる．従って，ある一枚のタグに対する処理 2 の解は単一の解となる．処理 3 の解は検出した物体に関する Action_Ontology 候補である．これは予め定義された複数の解となるが，解は一意な集合として得られる．すなわち，タグ一枚につき複数の Action_Ontology 候補 ID を記述する必要がある．これらを RFID メモリに格納しておくことで，本来はタグ検知の度に DB へ問合せを行う処理

をローカルで解決することができ，DB アクセスは Action_EE 登録 INSERT クエリ (図 2 の処理 1・5) のセンサ値生データと実空間で行われた Action_EE の挿入だけとなる．

RFID メモリから Action_EE を取得する場合，認識エンジンによる Action_EE の推定 (図 2 の処理 4) を行う前に Action_Ontology 候補をある程度絞り込む必要がある．ここで，二枚のタグ A，B を検知した場合における Action_Ontology 候補の抽出モデルを考える．まず，タグ A に書込まれた Action_Ontology 候補を取得し，次にタグ B に書込まれた Action_Ontology 候補を取得する．このままでは，二枚のタグから読出した Action_Ontology 候補全てを対象に認識エンジンによる Action_EE の推定を行う必要がある．当然候補が多ければ，その認識精度は下がる．そこで，Action_Ontology 候補を RFID メモリに記述する際，図 5 のように Action_Ontology 候補を分類し，あとで積集合処理を行い易いように編成しておく．Action_Ontology 候補の分類を以下のように定義した．

What Action：検知タグに対応する物体を「What」とした Action_EE が可能な Action Ontology．図 1 では「Thing Ontology：参考書」から見た「Action Ontology：参考書を机に置く」や「Action Ontology：参考書を書棚にしまう」が該当する．

What Action regardless of Where：What Action に該当する Action Ontology でも特に「Relation Ontology:Where」に関連付けられていないものを指す．

Where Action：検知タグに対応する物体を「Where」とした Action_EE が可能な Action Ontology．図 1 では「Thing Ontology：机」から見た「Action Ontology：参考書を机に置く」や「Action Ontology：ノートを机に置く」が該当する．

以上の分類はある一つの物体から見た場合，排他的であり，複数の分類にまたがる操作オントロジは存在しない．例えば，「Thing Ontology：参考書」に関係付けられた「Action Ontology:参考書を机に置く」という操作オントロジがある場合，この Action Ontology が「What Action」と同時に「What Action regardless of Where」に属することはない．

図 5 における Action_Ontology 候補の抽出モデルは次のようになる．まず，タグ A に記述してある「What Action」とタグ B に記述してある「Where Action」の積集合を取る．同様にタグ A に記述してある「Where Action」とタグ B に記述してある「What Action」の積集合を取る．この結果，同一の Action Ontology があれば，それがこの二枚のタグ検知列に関する Action_Ontology 候補であると絞り込む．この分類により，図 2 の処理 4 による利用者が実際に行った Action_EE の認識精度を高めることができる．なお,「What Action regardless of Where」は他のタグと積集合を取って共通する Action_Ontology 候補 ID を探す必要はなく，タグ検知列のどれかに存在すればそれだけで Action_Ontology 候補となり得る．また，現実では二枚以上のタグが同時に検知される場合もある．その場合はタグ検知列の中から 2 枚ずつの組合せを網羅的に作り，上述した方法によって Action_Ontology 候補の絞り込みを行っていく．

4. 実 装 環 境

3 章の設計に基づき実装を行った．実装にあたり，使用したサーバ・クライント PC 及び RIFD タグ・リーダを表 1 に示す．

4.1 電子タグ付き空間の構築

電子タグ付き空間では，利用者が無意識に行動した場合でもタグの検知漏れを可能な限り防ぐことが必要である．そこで，実験環境では一つの物体に対し複数枚のタグを貼付し，特に重要な物体や場所には大量のタグを貼付した．これにより，利用者の自然な振る舞いからタグを可能な限り検知できるような電子タグ付き空間を構築した．本研究で用いた電子タグ付き空間では，4,410 枚のタグが空間内のオブジェクトに貼付され，DB によって実空間の事物と関連付けられている．

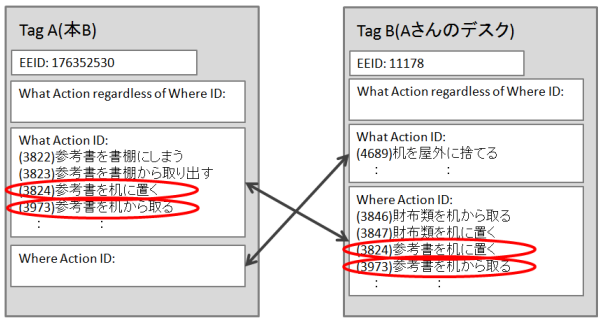


図 5　RFID メモリを利用した Action_Ontology 候補の取得モデル

表 1　実 験 環 境

Server	Dell Poweredge T410	
	OS	Microsoft Windows Server 2008 R2
	CPU	Intel Xeon E5530 x2
	Memory	16GB
	DBMS	Microsoft SQL Server 2008
Client	OS	Microsoft Windows 7 Professional x64
開発環境	Microsoft Visual Studio 2008	
RFID タグ	OMRON 社製 V730S-D13P01, V730S-D13P02	
RFID リーダ	FUJITSU 社製 F3972-T110	(有線タイプ:USB)
	WELCAT 社製 WIT-150-T	(無線タイプ:Bluetooth+SPP)

4.2 RFID メモリのデータ構造

RFID タグは表 1 で示したタグを利用した．タグの規格は ISO15693 であり，メモリ容量は 112Byte である．3 章の設計に基づき，RFID メモリに書込むデータ構造を図 6 に示す．まず，処理 2 の解を保持する領域として，そのタグと関連付けられた物体 EEID をそのまま記録する領域 (4Byte) を割り当てた．次に，処理 3 の解を保持する領域として，100Byte の領域を割り当てた．この領域には，Action_Ontology 候補群が記録される．ここで，そもそも Action_Ontology 候補の ID 空間は決して大きくない (すなわち一つの物体に対して行うことができる動作の種類は多くない) ことから一つの ID につき 2Byte を割り当て，最大で 50 個まで書込むこととした．Action_Ontology 候補は 3.5 節で述べたように,「What Action」「What Action regardless of Where」「Where Action」の三つに別けて RFID メモリに記述する必要がある．ヘッダ部分に含まれる Action_Ontology Counter は，タグに書込まれた Action_Ontology 候補を上述の 3 つに分類して各々の個数を記述する為に利用する．同じくヘッダ部分の Version はタグが書込まれた時期を示しており，タグに書込みが行われた時期を特定することを可能にする．限られた記憶領域を有効に活用できるよう以上の情報は全て符号無し Byte 型整数で書込むこととした．これらのデータ構造は限られたタグの記憶領域を最大限活用することを目指している．

5. 実装環境に関する予備調査

5.1 実空間情報の取得による DB サーバ負荷

本節では，Action_EE 登録トランザクションから生じる DB サーバ負荷の内訳について調べた．その為に，次の三つのケースにおける DB サーバ負荷を実験から求めた．

Action_EE 登録 SELECT+INSERT クエリ：図 2 の処理 1〜5 で生じる全ての DB アクセスを実行．

Action_EE 登録 SELECT クエリ：図 2 の処理 2・3 による DB アクセス (Action_EE 登録 SELECT クエリ) のみ実行．

Action_EE 登録 INSERT クエリ：図 2 の処理 1・5 による DB アクセス (Action_EE 登録 INSERT クエリ) のみ実行．

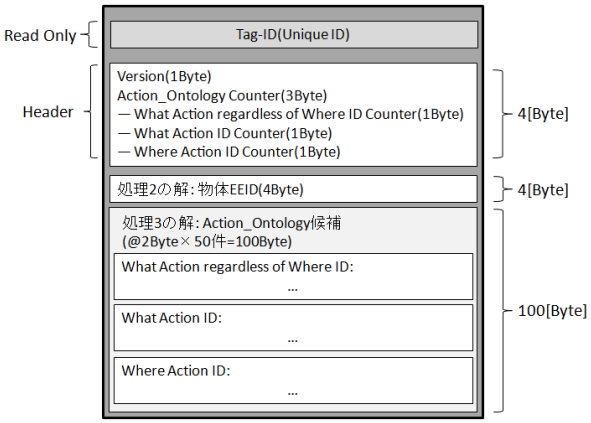


図 6　RFID メモリにおけるデータ構造

表 2　実空間情報の取得による DB サーバ負荷の内訳

発行クエリ	平均 CPU 使用率 [%]
Action_EE 登録 SELECT+INSERT クエリ	40.7
Action_EE 登録 SELECT クエリ	36.0
Action_EE 登録 INSERT クエリ	3.4

表 3　RFID リーダによる RFID メモリのアクセス時間

	アクセス時間 [s]	
機種	F3972-T110	WIT-150-T
Inventory	0.227	0.123
Inventory+Read	0.424	0.246
Inventory+Write	2.280	0.587

実験では表 1 の有線タイプのリーダを 10 台用い，リード間隔を最小の約 0.2[sec] にした上でタグ検知をトリガに各クエリを発行した．この時，常にタグが検知されクエリが発行され続けるようにした．これにより，常にタグ検知が起こり得る環境を再現した．この実験を各ケース毎に行い，DB サーバの平均 CPU 使用率を測定した．　なお，サーバは表 1 に示したものを利用した．結果を表 2 に示す．表 2 から図 2 の処理 2 及び 3 で生じる Action_EE 登録 SELECT クエリによるサーバ負荷の割合が大きいことが分かる．本研究では，これの削減を通して実空間情報の取得時に生じる図 2 の一連の処理による DB サーバ負荷を低減することを目標にする．

5.2　RFID メモリへのアクセス時間

本節では，実環境における RFID メモリの書込み及び読取りに影響を与えるであろう各処理の完了時間について述べる．ISO15693 規格における RFID リーダライタの RFID メモリへの書込み手順を図 7 に示す．まず，リーダは検知範囲内のタグを検出する為の Inventory コマンドを定期的に発行する．範囲内にタグがあれば，タグの応答から範囲内に存在するタグ全ての UID を得る．次に Inventory コマンドの応答から検知範囲内にタグの存在が確認できれば，範囲内のタグ一枚ずつに UID を指定した Write コマンドを発行して RFID メモリに書込みを行う．Inventory コマンド発行の結果，検知範囲内にタグが存在しなければ，Write コマンドは発行されない．リーダは上述した一連の処理により検知範囲に入ったタグの記憶領域に書込みを行っていく．RFID メモリの読取りも同様に，一度 Inventory コマンドを送り検知範囲内の UID を取得する．その後，UID を指定して各タグに Read コマンドを発行することで RFID メモリの読出しを行う．表 3 に各コマンドの処理完了時間を示す．なお，表 3 の Inventory+Write 及び Inventory+Read コマンド処理時間は UID を読取った上で RFID メモリ 112Byte の書込み及び読取りに要する時間である．表 3 に示した処理時間の差が 5.3 節で述べるタグ書込みや 6.2 節で述べる実環境における RFID メモリの読出しに影響を与える．

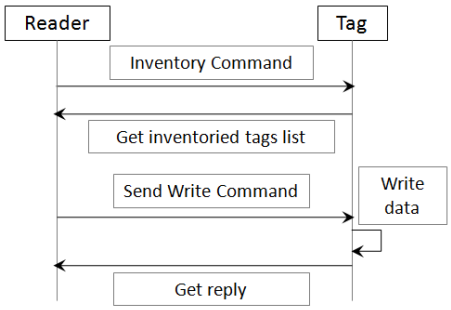


図 7　RFID メモリの書込み手順

5.3　電子タグ付き空間で利用されるタグへの書込み

実際の電子タグ付き空間に提案手法を適用する場合，環境内で利用される全てのタグに対する書込みタグ枚数の割合が実験結果に影響を与える．本節では，CONSENT を適用した電子タグ付き空間で利用されているタグへの書込み方法について述べる．既に利用されているタグが数百から数千枚になる場合，その全てに対し意図的に書込みを行うことは非常にコストがかかる．一方，適用環境である CONSENT ではリーダを利用者の利き手首に装着し，一つの物体に複数枚のタグを貼付している．これにより，利用者の自然な振る舞いの中でリーダがタグを検知し，3.3 節で述べたデータの蓄積を行っている [9]．そこで，本研究ではタグ書込みにおける前提条件として，日常生活で利用者が装備したリーダによるタグ検知をトリガに書込みを行うことにする．この方法ならば，利用者に負担をかけずに環境内のタグに書込みを行えると考えられる．但し，タグへの書込みはタグ ID や RFID メモリの読出しと比べると多少の時間を要する (表 3 を参照)．つまり，リーダの検知範囲内にタグが存在する時間が表 3 の Inventory+Write コマンドによるアクセス時間より短い場合は書込みに失敗してしまう．

そこで，利用者が日常生活を送る中でタグに書込みを行っていく場合の書込み成功率を実験から求めた．ここで，書込み成功率とはタグへの Write コマンド発行回数に対し書込みを完遂できた Write コマンドの割合と定義する．実験の前提条件として利用者には自然な振る舞いをしてもらった．実験環境では約 4,000 枚に及ぶタグが物体等に貼付されている．実験参加述べ人数は 10 人，期間は 5 日間で行った．この実験を表 1 で示した RFID リーダの有線タイプ，無線タイプのそれぞれで行った．結果を表 4 に示す．表 4 から，利用者の自然な振る舞いから検知したタグに対して 7 割以上の確率で書込みが行えていることが分かる．すなわち，日常生活の自然な振る舞いからでもタグへの書込みは十分に可能だと言える．

この結果から，実環境に提案手法を適用する為の環境構築として，利用者の自然な振る舞いを前提条件としたタグ検知をトリガにする書込みを用いた．これにより，環境内の利用タグ 4,410 枚のうち利用者が触れる機会の多い物体や場所に貼付されたタグを中心に 741 枚のタグに書込みが行われた．環境内には書込みがされているタグと書込みがされていないタグの両方が混在することになった．構築した電子タグ付き空間内における書込みタグの分布をバーチャル空間を用いて表現したものを図 8 に示す．図 8 は書込みの行われていないタグが貼付された物体を薄く表示することで，書込みタグが示す物体を強調表示したものである．図 8 から，タグへの書込みは一定の場所に集中したりせず，利用者が頻繁に利用する机や本棚，またその近辺の物品に対して行われていることが分かる．

5.4　提案手法において Action_EE 登録 SELECT クエリが発行されるケース

提案手法では基本的に Action_EE 登録 SELECT クエリは発

表 4　利用者の自然な振る舞いからのタグ書込み成功率

使用 RFID リーダ	Write コマンド発行数 [回]	書込み成功率 [%]
FUJITSU 社製 F3972-T110	5,689	71.6
WELCAT 社製 WIT-150-T	6,623	73.9

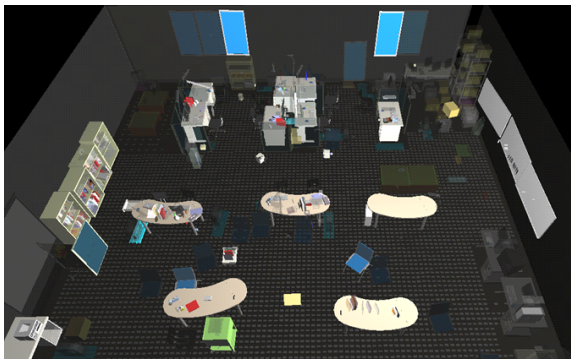


図 8　環境内における書込みタグの分布

生せず，Action_EE 登録 INSERT クエリのみが DB アクセスとして生じる．しかし，提案手法においても Action_EE 登録 SELECT クエリが DB サーバに対して発行されるケースがある．それは，タグは検知したが RFID メモリが読出せなかった場合 (表 3 を参照) や，検知タグに書込みが行われていなかった場合である．特に，5.3 節による書込み方法では環境内に書込みタグとそうでないタグが混在し易く，書込みが行われていないタグを検知する可能性も高い．ここで，提案手法を適用しても Action_EE 登録 SELECT クエリが発行されるケースを以下の二つにわけて定義する．

Read Error：表 3 の Inventory+Read コマンドによるアクセス時間よりもタグが検知範囲内に存在する時間が短く RFID メモリを読取る十分な時間がなかった場合である．なお，図 2 における処理 4 の結果，利用者が行った Action_EE は無かったと判断されたものも含まれる．

Non Deployment：検知したタグの RFID メモリに書込みがされていない場合である．

6.　評　　価

6.1　シミュレーションによるサーバ負荷低減手法の評価

Action_EE 登録トランザクションによる DB サーバへの負荷に着目して提案手法を評価する為シミュレーションによる実験を行った．電子タグ付き空間では利用者の増加や環境に埋め込まれるタグ枚数の増加に伴い，各利用者が装着したリーダから大量のタグ検知が生じる．そこで，複数のリーダから多量のタグ検知が生じた場合を想定し，複数のクライアントから Action_EE 登録トランザクションを DB サーバに対して送り続けることでサーバに過剰負荷をかけた．この状況下でサーバが一分間に処理できたトランザクション件数とその間の平均 CPU 使用率及びサーバの瞬間消費電力を従来手法と提案手法のそれぞれで測定した．電力測定には日置電機社製クランプオンパワーハイテスタ 3168 及びクランプオンセンサ (AC 100A 定格) 9298 を，なお，サーバは表 1 に示したものを用いた．シミュレーションに使用したクライアント数は 30 とした．シミュレーションによる Action_EE 登録トランザクション件数と DB サーバにおける平均 CPU 使用率の結果を図 9 に，Action_EE 登録トランザクション件数と DB サーバにおける平均瞬間消費電力の結果を図 10 に示す．図 9，図 10 から，従来手法に比べ提案手法では同じ Action_EE 登録トランザクション件数を低い CPU 使用率と低い瞬間消費電力で処理できていることがわかる．また，図 9 において従来手法では一分間の Action_EE 登録トランザクション処理が約 7,000 件で飽和してしまうが，提案手法では約 20,000 件の処理が可能となっている．すなわち，提案手法によってサーバの処理能力の限界を引き延ばすことができ，貧弱なサーバでもユビキタス環境 DB として十分に稼働させることが可能であると言える．これらのことから，提案手法では Action_EE 登録トランザクションによる DB サーバの負荷を低減できていることが確認された．

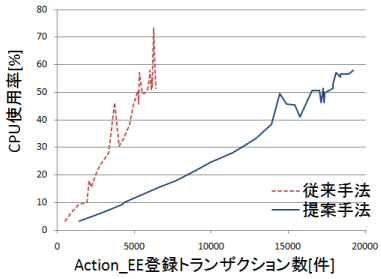


図 9　Action_EE 登録トランザクション件数とサーバ CPU 使用率の関係

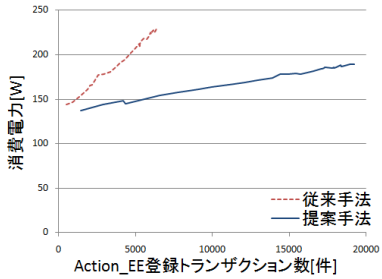


図 10　Action_EE 登録トランザクション件数とサーバ消費電力の関係

6.2　実環境におけるサーバ負荷低減手法の評価

実環境に提案手法を適用することで，提案手法の有効性及び実現性の評価を行う．4.1 節において実装し 5.3 節による書込みを行った電子タグ付き空間で，実生活に提案手法を適用して実験を行った．実験期間は従来手法と提案手法をそれぞれ 15 日間ずつとした．また，期間中は最も人が活動する時間帯である 10 時から 17 時においてサーバが処理した Action_EE 登録トランザクション件数と 7 時間の累計サーバ消費電力を毎日測定した．実験参加人数は 10 人であった．従来手法では，3.4 節の定義に基づき，Action_EE 登録 SELECT クエリが DB サーバに対して発行され，検知物体に対し実際に何らかの Action_EE が行われた場合はその Action_Ontology 候補を Action_EE として DB に蓄積する．一方，提案手法では Action_EE 登録 INSERT クエリのみとなる．例外として 5.4 節で定義したケースにおいては Action_EE 登録 SELECT クエリが DB サーバに対して発行されることになる．

まず，サーバが処理した Action_EE 登録トランザクション件数と消費電力の関係を図 11 に示す．各プロット点は測定日ごとの Action_EE 登録トランザション処理件数と 7 時間の累計サーバ消費電力を表している．図 11 におけるプロット点の分布状態から従来手法では Action_EE 登録トランザクションの処理件数の増加に伴いサーバ消費電力も増加傾向にあるが，提案手法では処理件数が変動しても消費電力は大きく変化していないことが分かる．このことから，提案手法ではタグ検知をトリガとする DB サーバの負荷を抑えられていると考えられる．

次に，図 12 では提案手法適用期間における発生 Action_EE 数の分布を実験環境を表したバーチャル空間を用いて表現した．図 12 における棒グラフは各々Action_EE の対象となったオブジェクトの座標から出ており，棒グラフの高さが発生した Action_EE 数を表している．各棒グラフ下部の濃い赤の部分は提案手法が適用されず，図 2 の処理 1 から 5 で生じる DB アク

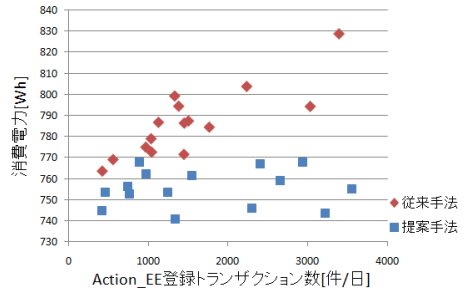


図 11　実生活における一日あたりのトランザクション件数と消費電力

セスを全て実行して処理した Action_EE 登録トランザクション件数を示す．各棒グラフ上部の淡いピンクの部分は RFID メモリを読み出すことで提案手法が適用され，Action_EE 登録 INSERT クエリによる DB アクセスだけで処理できた Action_EE 登録トランザクション件数を示す．図 12 から，環境内の多様な場所で起こるタグ検知によって生じる DB アクセスの低減が可能であることが言える．なお，同じオブジェクトから赤とピンクの棒グラフが伸びている原因にはまず Read Error による Action_EE 登録 SELECT クエリの発行がある．また，この理由以外にも Non Deployment による Action_EE 登録 SELECT クエリの発行が考えられる．これは一つのオブジェクトに対し複数枚のタグが貼付されていることに関係しており，同一のオブジェクトに対して書込みタグとそうでないタグが混在する場合がある為である．

最後に，提案手法適用期間中に発生した Action_EE 登録トランザクション件数のうち，どの程度が RFID メモリを利用して処理できたのか定量的に評価した．Action_EE 登録トランザクションをどのように実行したのかその内訳を図 13 に示す．RFID メモリを利用できず，Action_EE 登録 SELECT クエリを発行した場合は，その理由も合わせて示した．Reduced Select Query は提案手法によって削減することができた Action_EE 登録 SELECT クエリの件数である．図 13 から期間中に発生した Action_EE 登録トランザクションの約 70 %が提案手法により Action_EE 登録 SELECT クエリを発行せずに処理できたことがわかる．この結果から提案手法が実生活においても有効であることが示せた．また，Non Deployment 及び Read Error は DB サーバに Action_EE 登録 SELECT クエリを発行して処理した Action_EE 登録トランザクション件数を表している．しかし，Non Deployment における Action_EE 登録 SELECT クエリ発行は書込みが行われていないことに起因している．すなわち，タグ書込み環境が整っていない為に生じている部分である．本実験において環境内で利用されるタグおよそ 4,000 枚のうち，書込みが行われていたタグは約 700 枚であった．従って，Non Deployment が占める割合は環境内のタグ書込みが進むに連れ，Reduced Select Query となる余地があると考えられる．よって，環境内で利用されるタグの中でも利用者が日常的に触れるような一部のタグに書込みを行った場合でも，提案手法は十分に機能していると言える．以上の結果から，提案手法の有用性及び実現性を示すことができた．

7.　まとめと今後の課題

本稿では，電子タグ付き空間において RFID リーダによるタグ検知の度に生じる問合せの解を，予め RFID メモリに記述することによって DB サーバの負荷を低減させる手法を提案した．RFID メモリへのアクセスや書込みに関する問題及び実空間情報の取得にかかるサーバ負荷を明らかにした上で提案手法を設計・実装した．そして提案手法に対し CPU 使用率とサーバ消費電力の観点からサーバ負荷の評価を行い，その有用性を及び

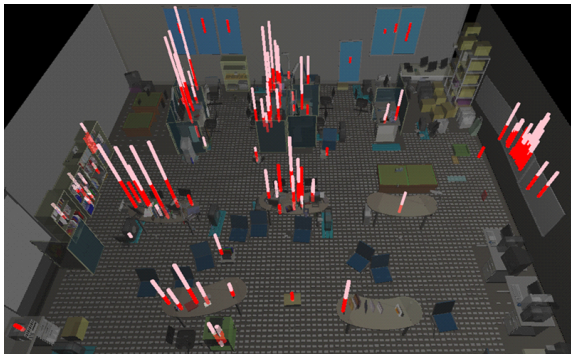


図 12　環境内における Action_EE 数の分布

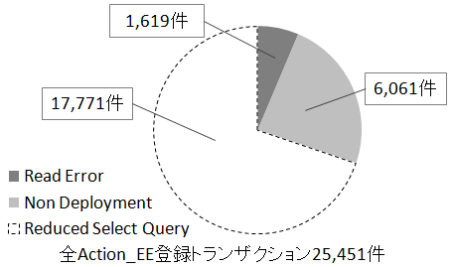


図 13　Action_EE 登録トランザクション累計発生件数の内訳

実現性を示した．今後は，書込みを行った RFID メモリの更新に関する手法の考案やタグの書込み枚数が提案手法のキャッシュヒット率に与える影響の評価を行っていく必要があると考えられる．

謝辞　本研究の一部は文部科学省科学研究費補助金 (若手研究 (B)：課題番号 21700102, および特定領域研究 (情報爆発 IT 基盤)：課題番号 21013023) の助成を受けて行った。

文　献

[1] 楓仁志，山原裕之，野口豊司，島田幸廣，島川博光，接触物体から個人の行動を認識するための確率的手法，情報処理学会論文誌，Vol. 48, No. 3, pp.1479-1490, 2007.

[2] 合田和生，QU Wenyu，喜連川優，複数問合せ処理を意識したディスクストレージ省電力化に関する一考察，Proc. of Data EngineeringWorkshop,DEWS2008 D5-2, 2008. 3.

[3] Qingbo Zhu and Yuanyuan Zhou, Power Aware Storage Cache Management, IEEE Trans. Computers, Vol. 54, No. 5, pp.587-602, 2005.

[4] Marco Mamei, Renzo Quaglieri and Franco Zambonelli, Making Tuple Spaces Physical with RFID Tags, Proc. of the 2006 ACM Symp. on Applied Computing, pp.434-439, 2006

[5] Marco Mamei and Franco Zambonelli, Pervasive pheromone-based interaction with RFID tags, ACM Trans. Autonomous and Adaptive Systems, Vol. 2, No. 2, Article 4, 2007

[6] Wooseok Ryu and Bonghee Hong, A Reprocessing Model Based on Continuous Queries for Writing Data to RFID Tag Memory, LNCS 5463, Proc. of Database Systems for Advanced Applications Conf, pp.201-214, 2009

[7] EPCglobal:EPC Tag Data Standards Version1.5, EPCglobal Ratified Specification, 2010

[8] Beth Bacheldor, Taiwan's Chang-Gung Hospital Uses HF RFID to Track Surgery, RFID Journal, 2007. 1.

[9] 清水隆司, 古賀浩史, 富井尚志, 大量の RFID データを扱う概念共有環境 CONSENT の運用による実用性の評価, 日本データベース学会論文誌, Vol. 8, No. 1, pp.41-46, 2009. 6.

[10] 猿田芳郎, 富井尚志, 加速度センサと RFID を用いたユビキタス環境での利用者コンテキスト推定手法, 日本データベース学会 Letters, Vol. 6, No. 3, pp.13-16, 2007. 12.