メタデータを用いたデータアクセス制御ミドルウェアの開発

Development of the Data Access Control Middleware Using Metadata

相馬 仁志♠ 市川 範子♥ 近藤 誠一♠
松田 昇平♥ 松岡 恭正♥

Hitoshi SOHMA Noriko ICHIKAWA
Seiichi KONDO Shohei MATSUDA
Yasumasa MATSUOKA

本稿では，情報系システム構築に必要なデータベースアクセスのセキュリティ確保を目的として，ユーザ権限に応じたデータベースの行単位・列単位でのアクセス制御機能について報告する．特徴として，組織変更や人事異動に連動してデータへのアクセス権の設定が可能であること，アクセス条件の指定による行単位・列単位のきめ細かなアクセス制御が可能なことが挙げられる．実現方式として，データベースのスキーマ情報に，アクセス権情報や論理情報などを付加することによりメタデータを構築し，ユーザ情報である人事情報や組織情報とともにディレクトリサーバで管理する．これにより，シングルサインオン等の個人認証との連携を強化する．

This letter reports the access control function in the row-level and column-level of the database according to user authority in order to secure the database access.

As features, this function enables the organization change and staff reassignment reflects automatically to the right to access data and enables fine-grained access control of the row-level and column-level by specifying access conditions.

As a realization system, by adding right-to-access information, logic information, etc. to the schema information on a database, metadata is built and it is managed by the directory server with the personnel information and organization information. This strengthens cooperation with authentication for single sign-on, etc.

1. はじめに

本稿では，アプリケーションセキュリティ，特に企業や団体における情報資産に対する利用者のアクセスの観点から，ユーザ認証と組合せた形でのデータベースへのアクセス制御について論じる．

2. 現状の課題と開発の目的

2.1 現状の課題

情報漏洩や情報の改ざん，誤操作や誤解によるデータの流出，消失を防止するために，企業や組織の情報資産を保護する必要がある．この手段の一つとして，情報が格納されているデータベースへのアクセス制御を実施する方法がある．例えば，二人の異なるユーザが，あるデータベースの同じテーブルを参照する場合，そのアクセスしているユーザによって，参照できるレコードやカラムを制限するというものである．

通常，利用しているデータベース製品にアクセス制御機能がないため，データベースそのものにアクセス制御のための情報を付加したり，データベースへアクセスするアプリケーションで対応したりする．たとえば，本来，顧客に関する情報のみが格納されている顧客情報データベースに対して，アクセスしてよいユーザ用のカラムを付加したり，また，現在利用しているデータベースの仕様に合わせてアプリケーションを開発したりする．

しかし，データベースの仕様変更や，システムによってはデータベース製品を変更する必要に迫られる場合があり，次のような課題がある．

- データベースの仕様変更に合わせて，アクセス制御のための情報を再度追加，変更しなければならない
- データベース製品の変更に伴い，システム毎にアプリケーションを修正しなければならない

2.2 開発の目的

これらの課題を解決するためには，データベース製品や仕様に依存することなく，汎用的なアクセス制御の仕組みが必要となる．

我々は，この解決策の一つとして，既存のデータベースに一切，手を加えることなく，メタデータを用いたアクセス制御により，これを解決しようと試みた．すなわち，既存データベースのスキーマや内容は変更することなく，またデータベース製品のバージョンや適用製品が変わっても，アプリケーションは変更する必要がないようにするための仕組みを開発する．

3. 開発機能

一般にユーザ毎にアクセス制御を実現するためには，まず，データにアクセスするユーザの特定を行う必要がある．ユーザの特定はシステムへのログインを許可するためのユーザ認証として行う．

ユーザの認証には，当社の開発物である統合認証システム[1][2]を用いた．統合認証システムは，個人や組織，コンテンツをそれぞれ別のディレクトリツリーで管理することにより，人事異動や組織変更に柔軟に対応できるようにした認証システムである．これを用いることによりユーザ管理やユーザ認証，コンテンツ制御機能と連携したアクセス制御機能を提供する．

3.1 概要

統合認証システムが持つディレクトリサーバ（LDAP サーバ）上に，個人，組織，コンテンツと同様にメタデータを持ち，これらを連動させたアクセス制御を行う機能を提供する．

① ディレクトリスキーマ定義：ディレクトリサーバ上でメタデータを保持するためのディレクトリスキーマの定義を行う．具体的には，ディレクトリ構造の決定とそれに沿ったオブジェクトクラス定義と属性定義を行う．

② データアクセス API：ユーザ毎にデータへのアクセスを制御するためのデータアクセス API（Java クラス）の提供を行う．この API を用いることにより，開発者はデ

♠ 正会員 三菱電機(株)情報技術総合研究所 {sohma, seiichi}@isl.melco.co.jp
♥ 非会員 同上 {noko, mazda, yymatsu}@isl.melco.co.jp

ータへのアクセス権を意識することなく，セキュリティレベルが統一されたアプリケーションを効率良く開発することができる．

③ 管理ツール：ディレクトリサーバ上でメタデータおよびアクセス権限を登録・変更・削除する機能を有する管理ツールの提供を行う．これにより，データベースのスキーマ情報を自動で収集したり，意味情報やアクセス権情報の設定・変更を実施したりすることが可能となる．

3.2 メタデータの定義

メタデータとして，データベースのスキーマ情報のほかに，意味情報やアクセス権情報を扱う．意味情報とはテーブルやカラムに対するコメントであり，また，アクセス権情報はユーザが対象とするテーブルやカラムにアクセス可能かどうか表す情報である．このメタデータをディレクトリサーバ上でユーザ情報とともに一元管理し，データベースへのアクセス制御情報として利用する．

3.3 機能構成図

統合認証システムと連携したデータアクセス制御の全体機能ブロック構成を図 1 に示す．この図の開発部分に記された数字①～③が上記の概要で示した開発項目に対応する．図の既存部分が統合認証システムの機能を表す．

また，統合認証システムが持つ個人，組織，コンテンツおよび役割情報のディレクトリ構成を図 2 に示す．

4. 実現方式

メタデータを用いたデータベースのテーブル単位，テーブルの列単位・行単位のアクセス制御方式について示す．

4.1 メタデータ構造

ディレクトリ上のメタデータの構造について示す．

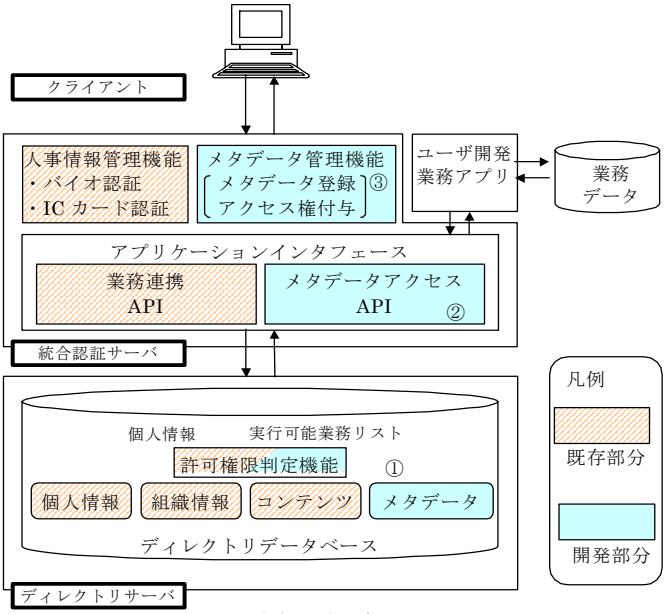

図 1 全体機能ブロック図
Fig.1 Function block of system

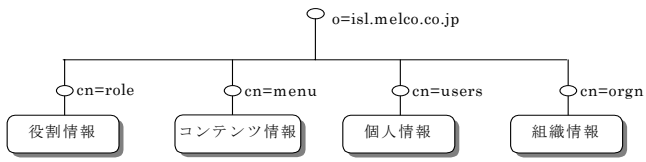

図 2 統合認証システムのディレクトリ構成
Fig.2 Directory Schema of Integrated Authentication System

4.1.1 属性定義

アクセス制御を行う上で必要となる属性として，LDAP に準拠するように，属性のオブジェクト識別子，属性名，属性の型を表 1 のように定義する．これらは独自に拡張した属性であり，オブジェクト識別子は，当社の企業識別子を用いる．また，型（SYNTAX）は RFC2252[3]で定義されているものである．

4.1.2 オブジェクトクラス定義

属性定義と同様に，メタデータをディレクトリ上で管理するために必要となるオブジェクトクラスとして，クラスのオブジェクト識別子，クラス名，必須属性，任意属性を表 2 のように定義する．

4.1.3 ディレクトリ構成

ディレクトリ構成を図 3 に示す．メタデータを格納するディレクトリとして，cn=metadata を作成し，その下にメタデータおよび関連する情報を入れる．

例えば，表 3 に示すようなデータベース INFODB にあるテーブル SALES の場合，カラム NO は dn:meCname=NO, meTname=SALES,meDname=DBINFO,cn=Tokyo,cn=physical, cn=metadata, o=isl.melco.co.jp と表される．

また，表 1 と表 2 に示すように，各エントリの属性として，テーブルの場合には，テーブルの論理名や所在など，カラムの場合には，データ型やキータイプなどが割り当てられる．この属性の一つとしてアクセス制御情報がある．

表 1 メタデータの属性定義（抜粋）
Table1 Attribute definition of metadata

OID	属性名（NAME）	型（SYNTAX）		内容
1101	meCode	15	Directory String	識別コード
1102	mePhysicalName	15	Directory String	物理名
1103	meLogicalName	15	Directory String	論理名
1104	meResourceName	15	Directory String	リソースに関する情報
1105	meAttribute	15	Directory String	属性に関する情報
1106	meDescription	15	Directory String	コメント
1107	meVersion	15	Directory String	バージョン情報
1108	meDataType	15	Directory String	データ型
1109	meDataLength	15	Directory String	データの長さ
1110	meDataKeyType	15	Directory String	データのキータイプ
1111	meDataNullType	15	Directory String	NULL許可
1112	meSemantic	15	Directory String	意味情報
1114	meDname	15	Directory String	データベース名
1115	meTname	15	Directory String	テーブル名
1116	meCname	15	Directory String	カラム名
1120	meDataAce	15	Directory String	アクセス権限
1121	meColumnDN	12	DN	カラム情報へのリンクDN
1123	meSemanticDN	12	DN	意味情報へのリンクDN
1124	meHostname	15	Directory String	DBが稼動しているサーバ名
1125	meDBProductName	15	Directory String	DBの製品名
1126	meDBProductVersion	15	Directory String	DBの製品バージョン
1127	meRolCode	15	Directory String	役割を表すコード

表 2 メタデータのオブジェクトクラス定義（抜粋）
Table2 Object definition of metadata

OID	クラス名	必須属性(MUST)	任意属性(MAY)	説明
1101	melSite	objectclass cn*	meDescription meDataAce	システムを表すクラス
1102	melDB	objectclass meDname*	meDataAce mePhysicalName meLogicalName ...	データベースを表すクラス
1103	melTable	objectclass meTname*	meDataAce mePhysicalName meLogicalName meAttribute ...	テーブルを表すクラス
1104	melColumn	objectclass meCname*	meDataAce meDataType meDataLength meDataKeyType ...	カラムを表すクラス
1105	melSemantic	objectclass meCode*	cn meLogicalName meSemantic meColumnDN	意味情報を表すクラス
1106	melRoleAttribute	objectclass meCode* meRoleCode meDname ...		行制御を行うための役割属性を表すクラス

4.2 アクセス制御方式

前節で示したテーブルやカラムの属性である meDataAce がそのオブジェクト（テーブル単位，カラム単位）にアクセス可能かどうかを表す．また，melRoleAttribute クラスがテーブルの行単位でのアクセス制御情報を表す．これらの情報を用いてアクセス制御を行う．

4.2.1 役割との関係

ここでの役割とは，アクセス制御を行うためにユーザに与えられた権限の単位である．例えば，R01 という役割をユーザ A に与える．そして，SALES テーブルにアクセス可能な役割は R01 であると設定する．これにより，ユーザ A は SALES テーブルにアクセスすることができる．すなわち，ログインしたユーザ情報を基に，ディレクトリ上からそのユーザの役割を取得し，メタデータには，各データ項目に対してアクセスできる役割を割り当てる．

図 4 に示すように，ユーザ A はログイン時に，ユーザ情報から役割 R01 を取得する．その役割情報を基に，メタデータで管理されているアクセス情報に従い，アクセス制御を行う．この場合，ユーザ A は metadata1 にはアクセスできるが，metadata2 にはアクセスできない．

4.2.2 列制御方式

テーブル単位やテーブルの列（カラム）単位での制御方式は，各テーブルオブジェクトやカラムオブジェクトの属性である meDataAce 属性に役割を設定することで実現する．

例えば，表 4 のような顧客情報テーブル CUSTOMER テーブルを例に考える．この場合のメタデータは，表 5 に示すようになる．この列制御情報（meDataAce）として，各カラムにアクセスできる役割を設定する．ここでは，INCOME と BALANCE というカラムは R03 という役割の権限を持った利用者しかアクセスができない．他のカラムについては，制限なくアクセス可能であるので，ANY が設定されている．

4.2.3 行制御方式

行制御は，行単位アクセス制御オブジェクトである melRoleAttribute の属性によって制御を行う．

行単位のアクセス制御は，テーブルのどのカラムをキーにして，その行にアクセスしてよいかどうか判断する必要がある．したがって，対象とするテーブルとキーになるカラム，そのカラムの値がどのような場合にアクセスしてよいか，および役割を与える．

例えば，営業 1 課の yamada，2 課の tanaka，3 課の sato が，表 3 の営業売上のテーブル SALES にアクセスする場合を考える．SALES テーブルには，各課の売上が格納されているが，それぞれのユーザは他の課の売上を見てはいけない．

この場合，SECTION カラムをキーとして，また，そのときの値がそのユーザの所属（例えば，yamada はディレクトリで管理されているユーザ情報より自動的に所属が営業 1 課であることがわかる）に等しいときにアクセス可能となるので，表 6 に示すコード S0001 のように属性を設定する．また，各ユーザには役割として R01 を与えることにより，行単位でのアクセスが可能となる．

4.3 アクセス制御の実装

このアクセス制御方式を実装するために Java によるアクセス制御 API の開発を行った．DB へアクセスするための標準的な java.sql クラスでは，DriverManager, Connection, Statement, ResultSet などを用いるが，これらと同様に，SdacManager, SdacConnection, SdacStatement, SdacResultSet などを作成した．

図 5 に示すように，これらを用いることにより，開発者はセキュリティを意識することなく，アプリケーションを今までと同様に作成することができる．たとえば，

```
SdacStatement stmt = conn.createStatement();
SdacResultSet rs = stmt.executeQuery("SELECT * FROM CUSTOMR);
```

とすることにより，内部的には実行ユーザの権限に応じて，

```
SELECT id, name, address, birthday, salesman FROM CUSTOMER WHERE SALESMAN = 'suzuki'
```

というような SQL 文が実行されている．

表 3 営業売上テーブル SALES の例
Table3 Example: SALES table

NO	SECTION	YEARMONTH	VOLUME
001	営業1課	2003/1	2000
002	営業1課	2003/2	1000
004	営業2課	2003/1	3000
005	営業2課	2003/2	2000
007	営業3課	2003/1	1500

表 4 顧客情報テーブル CUSTOMER の例
Table4 Example: CUSTOMER table

ID	NAME	ADDRESS	BIRTHDAY	INCOME	BALANCE	SALESMAN
12301	山田太郎	港区1-1	1953/12/24	10000	3000	yamada
12302	山田高志	港区2-1	1941/10/11	12000	1000	tanaka
12303	加藤花子	北区3-2	1978/11/15	8000	20000	yamada

表 5 顧客情報テーブル CUSTOMER のメタデータの例
Table5 Example: metadata of CUSTOMER table

エントリ	属性				
カラム名	データ型	キータイプ	論理名	コメント	列制御情報
ID	文字列型	主キー	顧客番号	xxxx	ANY
NAME	文字列型	−	氏名	xxxx	ANY
ADDRESS	文字列型	−	住所	xxxx	ANY
BIRTHDAY	日付型	−	誕生日	xxxx	ANY
INCOME	数値型	−	収入	xxxx	R03
BALANCE	数値型	−	残高預金	xxxx	R03
SALESMAN	文字列型	−	担当営業マン	xxxx	ANY

表 6 行単位アクセス制御オブジェクトの例
Table6 Example: row-level access control object

エントリ	属性				
コード	テーブル名	カラム名	役割	参照先情報	演算条件
S0001	SALES	SECTION	R01	所属	等しい
S0002	CUSTOMER	SALESMAN	R02	ユーザID	等しい

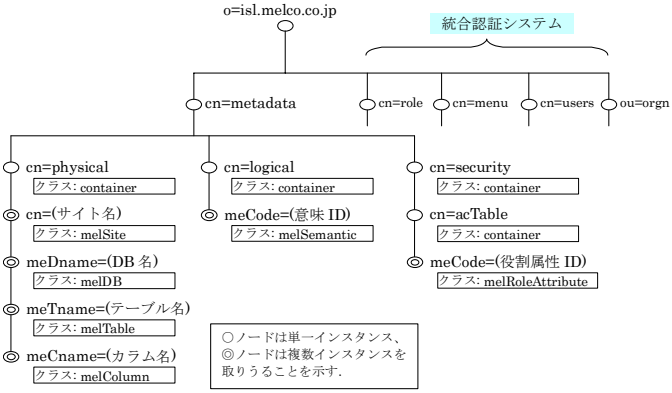

図 3 メタデータを追加したディレクトリ構成
Fig.3 Metadata added to directory schema

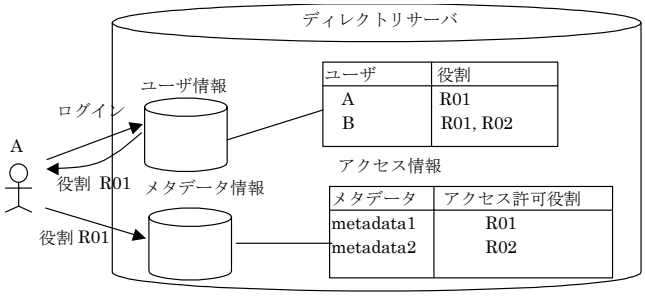

図 4 役割の使い方の概念図
Fig.4 Role usage

結果として図 6 に示すように，ユーザの権限により参照できるテーブルの範囲を制御することが可能となる．

5. 特徴と効果

データベースに関するアクセス制御の実体部分をディレクトリサーバとして，データベースの外に出すことにより，次のような特徴がある．

- 汎用性：データベース製品に依存することなく，JDBC 準拠であれば，汎用的にデータベースのアクセス制御をすることが可能
- セキュリティの一貫性：アクセス性御情報をメタデータとして一元管理することにより，統一されたセキュリティポリシーの基で一貫したアクセス制御が可能
- カスタマイズ性：LDAP サーバで管理することにより，JNDI などの標準インタフェースが利用でき，よりカスタマイズ性が向上

また，本方式により次の効果を上げることができる．

- テーブル単位やテーブルの行列単位でのアクセス制御が可能
- 管理ツールにより，メタデータの管理やアクセス権の設定が容易
- 人事異動等による会社組織の変更に対して，システムとして影響が少ない
- セキュリティを意識することなく，効率良くアプリケーションを開発することが可能

さらに，データベースのスキーマ情報をメタデータとして活用する場合に，論理的な意味情報などを付加することにより，次のようなことが可能となる．

- テーブルやデータの所在を意識することなくアクセスすることが可能

■通常のコーディング例

```
Connection conn =
DriverMDriverManager.getConnection
    ("jdbc:oracle:thin:@host:1521:orcl",
    "scott",
    "tiger");
// SQL 文の定義
String sql = "select * from customer";
Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while(rs.next()) {
        System.out.print(rs.getString(1));
}
```

■本方式を用いたコーディング例

```
SdacManager sm = new SdacManager();
SdacConnection conn = sm.getConnection
    ("jdbc:oracle:thin:@host:1521:orcl",
    "scott");

// SQL 文の定義
String sql = "select * from customer";
SdacStatement stmt = conn.createStatement();
stmt.setSecurityRole(roleList);
SdacResultSet rs = stmt.executeQuery(sql);

while(rs.next()) {
        System.out.print(rs.getString(1));
}
```

図 5 Java コーディング例
Fig.5 Example of java source code

■権限が高い山田部長が顧客テーブルを参照した場合

ユーザ:I開部 山田 部長(B) (yamada)

id	name	address	birthday	job	income	balance	salesman
12301	山田太郎	千代田区 1-1	1953/12/24	会社員	10000	3000	83001
12302	加藤花子	千代田区 1-2	1978/11/15	自営業	8000	20000	83001
12303	田中一郎	千代田区 1-3	1945/10/30	公務員	8000	4000	83002

ページが表示されました　イントラネット

■権限が低い鈴木担当が顧客テーブルを参照した場合

ユーザ:I開2 鈴木 担当 (suzuki)

id	name	address	birthday	job	income	balance	salesman
12301	山田太郎	千代田区 1-1	1953/12/24	会社員	*	*	83001
12302	加藤花子	千代田区 1-2	1978/11/15	自営業	*	*	83001

ページが表示されました　イントラネット

図 6 ユーザ権限に応じたアクセス制御
Fig.6 Example of access control according to user authority

- 対象とするテーブルの目的や各データの意味を把握することが可能

6. まとめ

本稿では，企業に蓄積されたデータベース資産に対して，ユーザの権限によりアクセス制御を行うことでセキュリティを確保する方式について述べた．すなわち，情報系システム構築に必要なデータベースアクセスのセキュリティ確保を目的として，ユーザ権限に応じたデータベースの行・列単位でのアクセス制御機能について報告した．これにより，

- シングルサインオンを実現する認証システムとの連携により，組織変更や人事異動に連動してデータへのアクセス権の設定が可能であること
- アクセス条件の指定による行単位・列単位のアクセス制御が可能なこと

などを挙げた．

実現方式としては，データベースのスキーマ情報にアクセス権情報や論理情報などを付加してメタデータとして，ユーザ情報である人事・組織情報とともにディレクトリサーバで管理することにより，統合認証との連携の強化を図った．

機能として，メタデータ管理機能，セキュアデータアクセス API を提供し，統一されたセキュリティポリシーの下，アプリケーション開発の効率化，セキュリティ維持のための管理運用負荷の軽減をねらった．

今後の課題としては，不正アクセスやインシデント発生時の事後否認を防止，抑制したいという要求に対する監査証跡機能の検討やユーザ認証機能として他の製品や機能を組合せた場合の対応などが挙げられる．

[文献]

[1] 菊竹秀夫，荻野義一，松岡恭正，虎渡昌史，五月女健治:”金融情報システム向けセキュア情報活用ソリューション”，三菱電機技報 Vol77No4,2003 年 4 月号 (2003).

[2] 石井洋 他:”ディレクトリを用いた標準企業ポータル/認証システム”，情報処理学会 第 64 回全国大会 (2002).

[3] M. Wahl, A. Coulbeck, T. Howes, S. Kille: ”Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions”, (RFC2252).

相馬 仁志　Hitoshi SOHMA
三菱電機株式会社 情報技術総合研究所勤務．異種分散システム連携技術の研究・開発に従事．情報処理学会正会員．日本データベース学会正会員．

市川 範子　Noriko ICHIKAWA
三菱電機株式会社 情報技術総合研究所勤務．異種分散システム連携技術の研究・開発に従事．

近藤 誠一　Seiichi KONDO
三菱電機株式会社 情報技術総合研究所勤務．システム間連携，統合データベースに関する研究・開発に従事．情報処理学会正会員．日本データベース学会正会員．

松田 昇平　Shohei MATSUDA
三菱電機株式会社 情報技術総合研究所勤務．Web システムにおけるトランザクション技術の研究・開発に従事．情報処理学会正会員．

松岡 恭正　Yasumasa MATSUOKA
三菱電機株式会社 情報技術総合研究所勤務．高信頼 Web システム技術の研究・開発に従事．