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

2005－SE－149（4）
2005／7／28

X.500ディレクトリのためのモデリング言語の提案

五月女健治[†]　近藤誠一[††]

X.500ディレクトリサービスが、企業の情報システムを構築する道具のひとつとして位置付けられ、利用され始めている。しかし、ディレクトリの設計においては標準的な設計モデルがなく、ディレクトリ構造を例示する程度の不完全な図を示すのが一般的で、ディレクトリ設計のための手法が求められている。本稿では、X.500ディレクトリにおけるディレクトリ情報ツリーの設計を目的とするUMLをベースに拡張したモデリング言語と、そのモデルに基づいてディレクトリ情報ツリーを実装する方法を提案した。

Proposal of the modeling language for X.500 Directory

KENJI SAOTOME,[†]　SEIICHI KONDO[††]

X.500 Directory Service begins to be used as a tool for the development of Enterprise Information System. Nevertheless, there are not the standards of the model for the design of the Directory and generally incomplete diagrams of the Directory have been illustrated. The methods to design the Directory are expected. Then, we proposed the modeling language extending UML for the design of Directory Information Tree(DIT), and the method to implement the DIT as a target based on the model described by the language.

1. はじめに

近年、ディレクトリサービスの標準化が進み、企業の情報システムのデータを管理することを利用目的としたディレクトリサービス製品がリリースされるようになり、利用され始めている。しかし、ディレクトリのデータ構造を設計するための標準的な設計モデルがないために、階層構造を例示する程度の不完全な図を示すのが通例となっているようである。そのため、従来のディレクトリの構築方法では、ディレクトリ設計者とそれを利用するプログラム開発者やディレクトリ情報の利用者間での円滑な意思疎通を実現できず、プログラム開発者や利用者にとって所望のディレクトリを得ることが困難となり、稼動間際に問題が発生するプロジェクトも見受けられる。本稿では、広く使用されているUMLモデルをベースとして、ディレクトリサービスのデータ構造を一意に表現できるディレクトリサービスに特化したディレクトリモデル言語を提案する。

[†]法政大学大学院イノベーション・マネージメント研究科
Hosei Business School of Innovation Management
[††]三菱電機株式会社 情報技術総合研究所
Mitsubishi Electric Corporation,
Information Technology R&D Center

本稿で対象とするディレクトリとは、ITU-T(International Telecommunication Union -Telecommunication Standardization Sector) X.500シリーズ勧告で定義されている構造を持ち、IETF（Internet Engineering Task Force）で定義されている LDAP（Lightweight Directory Access Protocol）のインタフェースを有するものとする。また、提案するディレクトリモデル言語は、OMG(Object Management Group)で標準化されたUML(Unified Modeling Language)をベースとする。

2. ディレクトリモデル言語の定義仕様

2.1. 概要

ディレクトリエントリの関係を UML で示すことを考えるとき、図-1 のような UML における典型的な２つのクラスの 1 対多の関連は、図-2 のようなDIT（Directory Information Tree）に対応すると考えると、UMLによるモデル化が妥当であることが分かる。そこで、図-3 のようなクラスとその関連に対しては、図-4 のような対応が考えられる。しかし、ディレクトリは階層ツリー構造で、ディレクトリエントリは２つの上位エントリを持つことができないため、DITだけで図-3 のようなクラスとその関連を表現することはで

きない。これらのクラスのエントリを関連付けるいくつかの別の手段を利用することによって、UMLをベースにしたモデリングにより適切なディレクトリの設計・構築を実施できるディレクトリモデル言語を提案する。

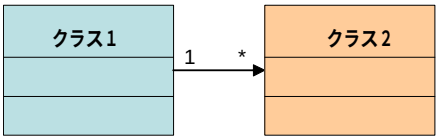

図-1　クラスの関係(1)

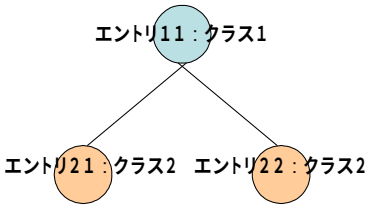

図-2　図-1 に対応する DIT

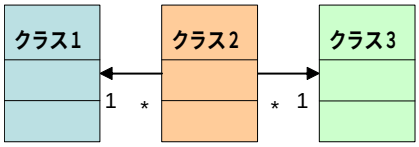

図-3　クラスの関係(2)

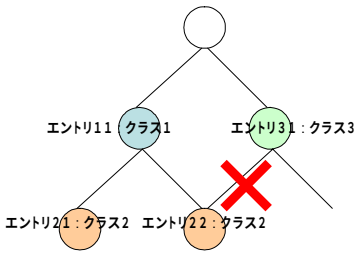

図-4　図-3 に対応する DIT

2.2. ディレクトリモデル言語の構文仕様

(1)ディレクトリモデル言語で記述するモデルは、UML のクラス図によって表現する。
(2)クラス(class)には、クラス名およびディレクトリエントリを定義することを示すステレオタイプ<<LDAPEntity>>を指定する。クラス名は、当モデル内で一意でなければならない。
(3)クラスには、{LDAPObjectClass}タグ付き値を指定する。{LDAPObjectClass}タグ付き値には、1つ以上の文字列を指定する。
(4)クラスは、複数の属性(attribute)をもつ。
(5)属性は、属性名からなる。指定する属性名は、当クラス内で一意でなければならない。
(6)属性には、ステレオタイプ<<LDAPRDN>>を指定することができる。クラスの中で、1つ以上の属性に対して、ステレオタイプ<<LDAPRDN>>を指定しなければならない。
(7)クラス間の関連(association)を指定することができ、関連に対するステレオタイプ、多重度(multiplicity)、方向および役割(role)に対するステレオタイプとタグ付き値を指定できる。あるクラスにおいて、矢印の始点となる関連が複数あるとき、それらの関連の他方のクラス側に指定された役割は互いに一意でなければならない。
(8)関連の多重度は、以下のどれかを指定する。*は、0以上を意味する。

- 1
- *

(9)関連の方向は、双方向または片方向を指定する。
(10)関連の方向で示した矢印の終点に、当クラスに対する役割を指定する。
(11)関連には、以下の種類のうちの1つのステレオタイプを指定しなければならない。

- <<LDAPDIT>>
- <<LDAPDN>>
- <<LDAPQuery>>
- <<LDAPAttr>>

(12)役割には、<<LDAPUpper>>ステレオタイプを指定することができる。
(13)役割には、以下のタグ付き値を指定できる。

- {LDAPKey}

{LDAPKey}タグ付き値には、指定した側のクラスの属性名を指定する。

2.3. ディレクトリモデル言語の意味仕様

ディレクトリモデル言語の意味仕様を、対応する DIT と関連付けて記述する。

2.3.1. クラス

クラスは、DIT を表現することを示すために<<LDAPEntity>>ステレオタイプを指定する。
クラスには、そのクラスに属するディレクトリエントリを構成するオブジェクトクラスを{LDAPObjectClass}タグ付き値で記述する。{LDAPObjectClass}タグ付き値には、複数のオブジェクトクラスを指定可能とする。ただし、top オブジェクトクラスは、すべてのディレクトリエントリに存在するため省略する。
クラスに定義される属性の属性名は、{LDAPObjectClass}タグ付き値で示したオブジェクトクラスのどれかに属するディレクトリ属性型を指定する。なお、クラスに定義される属性の属性型は指定しない。これは、ディレクトリ属性

型によって属性構文が特定できるためである。

<<LDAPRDN>>ステレオタイプは、ディレクトリエントリの RDN（Relative Distingushed Name）となる属性型であることを示す。

図-5 は、ディレクトリモデル言語で記述されたクラスの例である。

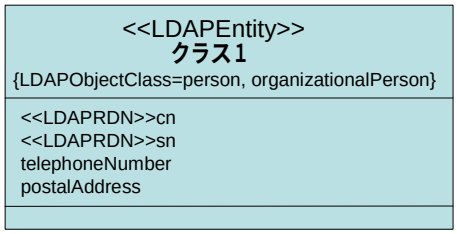

図-5　クラス

2.3.2. クラスの関連

2.3.2.1. 概要

関連は、ディレクトリエントリ間の関係を示すために使用する。関連の矢印は、始点のクラスに属するディレクトリエントリから、終点のクラスに属するディレクトリエントリへの参照関係が確立していることを示す。関連の実現方法を表現するために、UML における関連に対して、<<LDAPDIT>>、<<LDAPDN>>、<<LDAPQuery>>または<<LDAPAttr>>ステレオタイプのいずれかを指定する。

2.3.2.2. <<LDAPDIT>>による関連

<<LDAPDIT>>ステレオタイプは、その関連を DIT により実現することを示す。関連する２つのクラスにおいて、役割に<<LDAPUpper>>ステレオタイプを指定したクラスに属するディレクトリエントリが、他方のクラスに属するディレクトリエントリの直接上位エントリとなるように配置することによって関連を実現する。関連を DIT により実現するため、以下の規則を適用する。

- 多重度は、１対１または１対*でなければならない。
- <<LDAPUpper>>ステレオタイプを指定したクラスに指定する多重度は1でなければならない。ただし、関連の多重度が１対*の場合は、<<LDAPUpper>>ステレオタイプは、多重度として１を指定した側に指定したものとみなし、省略できる。
- １つのクラスに複数の<<LDAPDIT>>ステレオタイプによる関連があるとき、そのクラスは２つ以上の<<LDAPUpper>>ステレオタイプを役割としてもつクラスと関連を持つことはできない。

図-6 は、<<LDAPDIT>>ステレオタイプを使用した UML モデルであり、図-7 は、図-6 に対応する DIT の概略を示す。DIT による関連の実現は、ディレクトリとしては最も自然な構造である。また、企業の組織を表現する場合など、自身のクラスと関連をもつ再帰的関連は、ディレクトリの DIT によって表現できる典型である。図-8 は再帰的関連を表す UML モデルであり、図-9 は図-8 に対応する DIT の概略を示す。

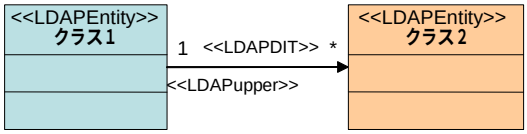

図-6　<<LDAPDIT>>によるクラス図

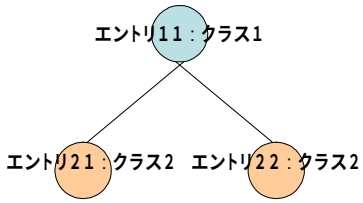

図-7　図-6 に対応する DIT

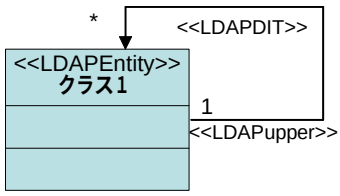

図-8　<<LDAPDIT>>による再帰的関連

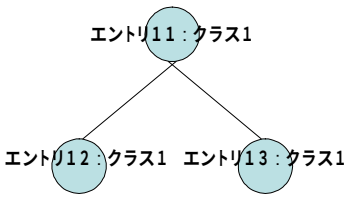

図-9　図-8 に対応する DIT

2.3.2.3. <<LDAPDN>>による関連

<<LDAPDN>>ステレオタイプは、その関連を DN（Distingushed Name）により実現することを示す。矢印の始点となるクラスに属するディレクトリエントリが、その矢印の終点となる他方のクラスに属するディレクトリエントリの DN をもつことによって関連を実現する。関連するエントリが複数あるとき、すなわち他方のクラスの多重度が*のときは、このディレクトリ属性型は複数の属性値をもつ。この関連の実現方法は、ディレクトリサービスの静的グループで、グループとそのメンバーの関係を表現するときに使用されている

方法と類似する。図-10 は<<LDAPDN>>ステレオタイプを使用した UML モデルであり、図-11 は図-10 に対応する DIT の概略を示す。なお、図-11 の点線は、関連するディレクトリエントリを示すためのもので、実際の DIT を構成するための表記ではない。

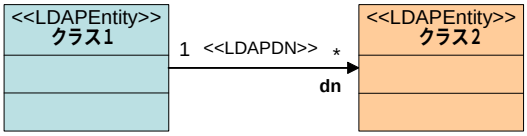

図-10　<<LDAPDN>>によるクラス図

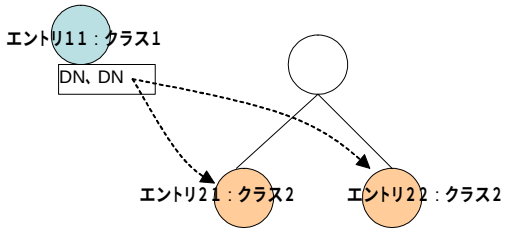

図-11　図-10 に対応する DIT

2.3.2.4. <<LDAPQuery>>による関連

<<LDAPQuery>>ステレオタイプは、その関連を任意の検索条件の記述を利用して実現することを示す。矢印の終点となる他方のクラス側に属するディレクトリエントリを得るための検索条件によって関連を実現する。

この関連の実現方法は、いくつかのディレクトリサービス製品で採用されている動的グループを実現するときに、グループとそのメンバーの関係を表現するときに使用されている方法と類似する。なお、動的グループにおける検索条件は、LDAP URL 形式で指定する。

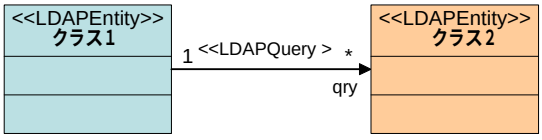

図-12 <<LDAPQuery>>によるクラス図

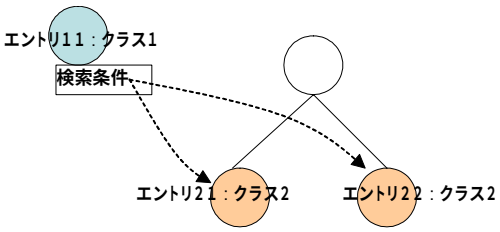

図-13　図-12 に対応する DIT

図-12 は、<<LDAPQuery>>ステレオタイプを使用した UML モデルであり、図-13 は図-12 に対応する DIT の例である。

2.3.2.5. <<LDAPAttr>>による関連

<<LDAPAttr>>ステレオタイプは、その関連を任意のディレクトリ属性型の値を利用して実現することを示す。関連には、矢印の終点となる他方のクラス側の役割に{LDAPKey}タグ付き値を指定する。{LDAPKey}タグ付き値には、矢印の終点となる他方のクラス側の属性の中で、関連に使用する属性名を指定する。ディレクトリエントリ内のあるディレクトリ属性型に保持された属性値と、他方のクラスに属するディレクトリエントリ内の{LDAPKey}タグ付き値で指定したディレクト属性型が保持する値が一致することによって関連するとみなす。この関連の実現方法は、リレーショナルデータベースにおける主キーと外部キーの関係と類似する。図-14 は<<LDAPAttr>>ステレオタイプを使用した UML モデルであり、図-15 は図-14 に対応する DIT の例である。

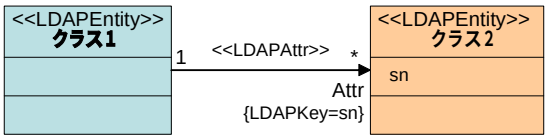

図-14 <<LDAPAttr>>によるクラス図

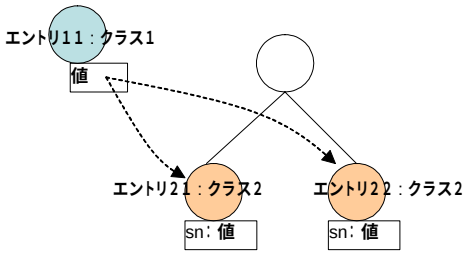

図-15　図-14 に対応する DIT

3. 実装例

ディレクトリモデル言語により、UML モデルから DIT を実装する方法の 1 例を示す。

3.1. ディレクトリエントリ

関連情報を除いたあるクラスに属するディレクトリエントリは、クラスに記述した{LDAPObjectClass}タグ付き値のオブジェクトクラス、属性と同じ名前のディレクトリ属性型および属するクラスを示す情報から構成する。属するクラスを示す情報は、DIT でどのクラスに属するエントリかを明確にする。特に、<<LDAPQuery>>および<<LDAPAttr>>ステレオタイプの関連において、関連する他方のクラスに属するディレクトリエントリを取得するとき、その検索操作をクラス内に限定するために有用である。図-16 は、図

-5 で定義したクラスのディレクトリエントリの内容を、LDIF（LDAP Data Exchange Format）形式で示したものである。図中、クラス情報を示すディレクトリ属性型を UMLEntryClass、その属性型をもつオブジェクトクラスを UMLEntry としている。

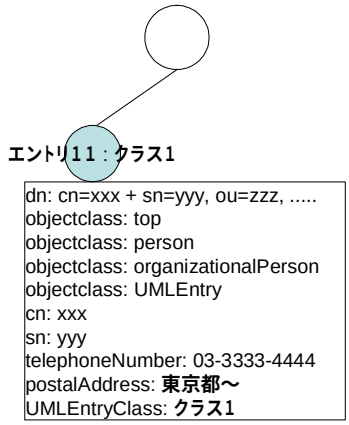

図-16　ディレクトリエントリの実装例

3.2. ベースエントリ

当モデルで示すクラス群の最上位エントリとなるモデルベースエントリを定義する。モデルベースエントリは、モデル全体をひとつの DIT とするためのルートとしての役割を持つ。また、クラスに属するすべてのディレクトリエントリを下位にもつクラスベースエントリを定義する。クラスベースエントリは、クラスに関連する情報を保持する役割と同時に、<<LDAPQuery>>および<<LDAPAttr>>ステレオタイプの関連において、ディレクトリエントリの検索における検索ベースとしての役割をもつ。モデルベースエントリ、クラスベースエントリおよびディレクトリエントリによる DIT は以下のように決める。

- クラスに関連がないか、または関連があっても他方のクラス側の役割に<<LDAPupper>>ステレオタイプをもつ<<LDAPDIT>>ステレオタイプの関連が含まれていないとき、モデルベースエントリの直接下位にエントリを作成し、それを当クラスに対するクラスベースエントリとする。当クラスに属するディレクトリエントリは、当クラスベースエントリの直接下位となるよう配置する。
- クラスに関連があって、再帰的な関連を除く、他方のクラス側の役割に<<LDAPUpper>>ステレオタイプをもつ<<LDAPDIT>>ステレオタイプの関連が含まれている（たかだか一組のみ）とき、モデルベースエントリの直接下位にエントリを作成し、それを当クラスに対するクラスベースエントリとする。このクラスベースエントリは、この関連における他方のクラスのクラスベースエントリの別名エントリとする。他方のクラスで、クラスベースエントリが決まっていないときは、他方のクラスベースエントリを決めて、そのクラスベースエントリの別名エントリとする。このとき、他方のクラスベースエントリが別名エントリのときは、この別名エントリが指し示すクラスベースエントリを、当該クラスに対する別名エントリが指し示すエントリとする。

クラスの関連が<<LDAPDIT>>ステレオタイプによる再帰的な関連でないとき、当クラスに属するディレクトリエントリは、他方のクラスに属するディレクトリエントリの直接下位となるよう配置する。クラスの関連が<<LDAPDIT>>ステレオタイプによる再帰的な関連をもつとき、当クラスに属するディレクトリエントリのサブツリーを、他方のクラスに属するディレクトリエントリの直接下位となるよう配置する。

図-17 のモデルの例を、上述のように構築したときの DIT を図-18 に示す。また、図-19 は、<<LDAPDIT>>ステレオタイプとの関連がないときのクラスベースエントリと、<<LDAPDIT>>ステレオタイプとの関連があるときに別名エントリとなるクラスベースエントリの実装例を示す。この図では、クラスベースエントリは、クラス名を保持する UMLClassName ディレクトリ属性型からなる UMLClass オブジェクトクラスをもつ。このような実装を行うと、モデルベースエントリの DN を既知のものとすると、クラス名からクラスベースエントリの DN を一意に決定できる。たとえば、モデルベースエントリの DN を、“dc=xxx, dc=com”とすると、クラス 1 のクラスベースエントリは、”UMLClassName=クラス 1，dc=xxx，dc=com”のように一意に決定できる。

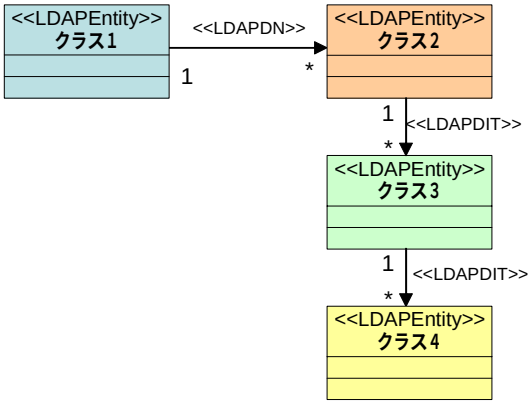

図-17　ディレクトリモデル言語で記述したクラス

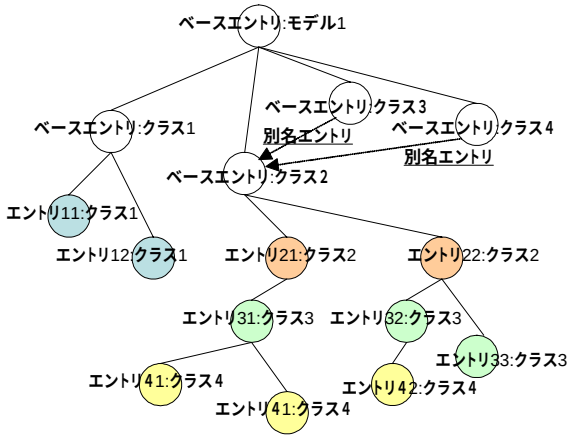

図-18　図-17 に対応する DIT

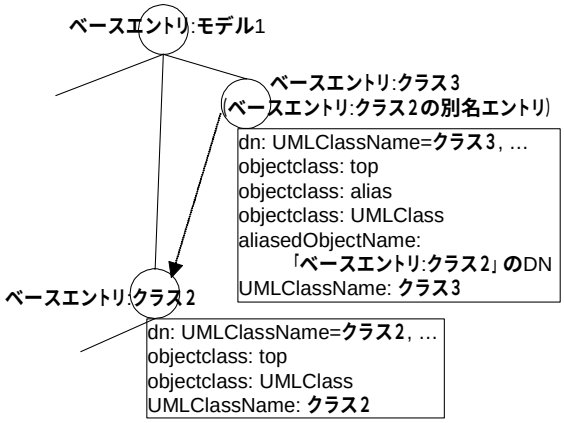

図-19　ベースエントリの実装例

3.3. <<LDAPDIT>>による関連

<<LDAPDIT>>ステレオタイプの関連は DIT により表現できるため、図 5 のディレクトリエントリだけで実現できる。図-20 は、図 6 のクラスとその関連を上述のように実装したときの DIT の例である。図中の各エントリは、図 5 のディレクトリエントリと同一の属性型をもつ。

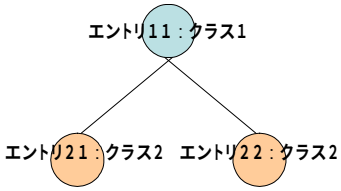

図-20　<<LDAPDIT>>の関連の実装例

3.4. <<LDAPDN>>による関連

<<LDAPDN>>ステレオタイプの関連は、以下のように実装する。UML モデルにおける矢印の始点のクラスに属するディレクトリエントリに、関連情報を格納する関連情報エントリを下位エントリとして作成する。この関連情報エントリは、UMLAssocName ディレクトリ属性型と UMLAssocDNs ディレクトリ属性型をもつ UMLDN オブジェクトクラスからなる。UMLAssocName ディレクトリ属性型は、当ディレクトリエントリの RDN を形成するディレクトリ属性型とし、UML モデルの役割を値としてもつ。UMLAssocDNs ディレクトリ属性型は、関連するディレクトリエントリの DN を 1 つ以上値としてもつ。図-21 は、図 10 のクラスとその関連を上述のように実装したときの DIT の例である。図中のエントリ xx は図 5 のディレクトリエントリと同一のディレクトリ属性型をもち、ベースエントリは、図 19 のディレクトリエントリと同一のディレクトリ属性型をもつ。また、エントリ xx の右下に記述した“cn=X11”などは、当エントリの RDN を示す。

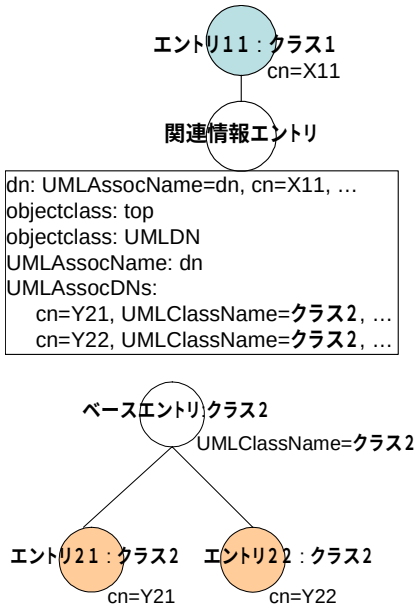

図-21　<<LDAPDN>>の関連の実装例

3.5. <<LDAPQuery>>による関連

<<LDAPQuery>>ステレオタイプの関連は、以下のように実装する。UML モデルにおける矢印の始点のクラスに属するディレクトリエントリに、関連情報を格納する関連情報エントリを下位エントリとして作成する。この関連情報エントリは、UMLAssocName ディレクトリ属性型と UMLAssocQuery ディレクトリ属性型をもつ UMLQuery オブジェクトクラスからなる。

UMLAssocName ディレクトリ属性型は、当ディレクトリエントリの RDN を形成するディレクトリ属性型とし、UML モデルの役割を値としてもつ。UMLAssocQuery ディレクトリ属性型は、関連するディレクトリエントリを検索する時の検索条件を値としてもつ。図-22 は、図-12 のクラスとそ

の関連を上述のように実装したときの DIT の例である。図中の UMLAssocQuery ディレクトリ属性型の属性値である検索条件は、いくつかのディレクトリサービス製品で採用されている動的グループと同じ LDAP URL 形式で指定している。この LDAP URL 形式の検索条件は、以下の LDAP の検索操作を行う。

・検索ベース
　　クラス 2 のベースエントリ
・検索スコープ
　　sub（検索ベースの下位エントリすべて）
・検索フィルター
　　クラス 2 に属するディレクトリエントリ、かつ mail 属性の属性値が ”*@y.ac.jp ”
　　（部分文字列、*は任意の文字列）

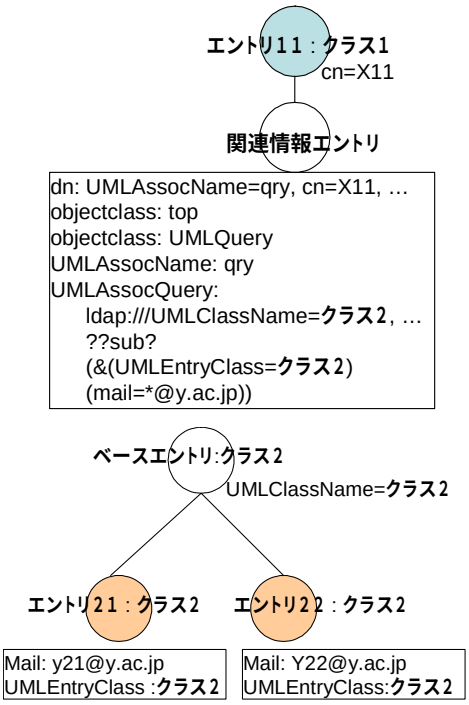

図-22 <<LDAPQuery>>の関連の実装例

3.6. << LDAPAttr>>による関連

<< LDAPAttr>>ステレオタイプの関連は、以下のように実装する。UML モデルにおける矢印の始点のクラスに属するディレクトリエントリに、関連情報を格納する関連情報エントリを下位エントリとして作成する。この関連情報エントリは、UMLAssocName ディレクトリ属性型、UMLAssocClass ディレクトリ属性型、UMLAssocKey ディレクトリ属性型および UMLAssocValues ディレクトリ属性型をもつ UMLAttr オブジェクトクラスからなる。UMLAssocName ディレクトリ属性型は、当ディレクトリエントリの RDN を形成するディレクトリ属性型とし、UML モデルの役割を値としてもつ。UMLAssocClass ディレクトリ属性型、UMLAssocKey ディレクトリ属性型および UMLAssocValues ディレクトリ属性型は、関連するディレクトリエントリを LDAP の検索操作を行うときの検索条件を形成するために使用する。これらの値で形成される検索条件は以下のようである。

・検索ベース
　　MLAssocClass ディレクトリ属性型の属性値をクラス名とするクラスのベースエントリ
・検索スコープ
　　sub（検索ベースの下位エントリすべて）
・検索フィルター
　　UMLAssocClass の属性値をクラス名とするクラスに属し、かつ UMLAssocKey の属性値で示した名前を持つディレクトリ属性型が UMLAssocValues の属性値がもつ値と一致するディレクトリエントリ

図-23 は、図-14 のクラスとその関連を、上述のように実装したときの DIT の例である。この例で形成される LDAP 検索条件は、以下のようになる。

ldap:///UMLClassName=クラス 2,･･･??sub
?(&(UMLEntryClass=クラス 2)(sn=zzzzzz))

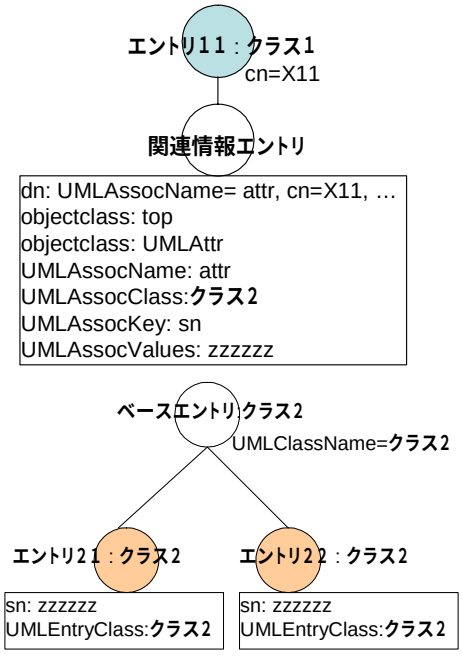

図-23 <<LDAPAttr>>の関連の実装例

4. ケーススタディ

すでに構築済みのディレクトリサービスを利用したシステムを取り上げ、本稿で示した UML モデリング言語で記述できることの検証、および従来慣習的に記述していた図との記述能力の比較を行う。対象のシステムとして、認証・認可機能を備えたワークフローシステム T を取り上げる。

図-24 は、当ワークフローシステムの DIT の設計・保守を目的に記述したツリー構造表現であり、図-25 は、当 DIT を本稿のモデリング言語で記述したモデルである。今回の検証で、当ワークフローシステムの DIT を本稿のモデリング言語で記述できることが確認できた。従来の記述方法と比較して、本稿で示した UML モデリング方法は、以下のような長所があることがわかる。

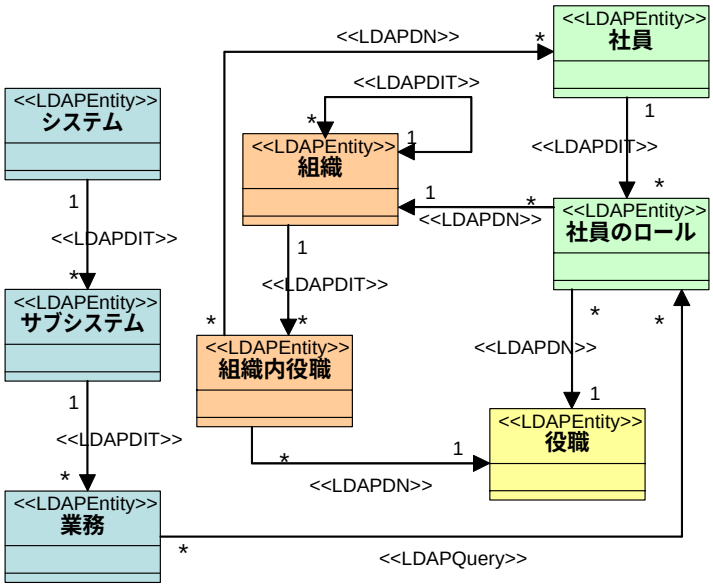

図-25　本稿モデル言語による記述例

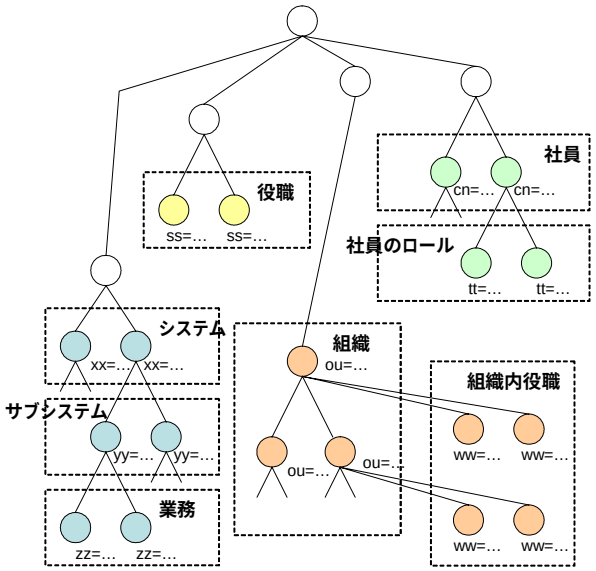

図-24　従来の DIT の記述例

(1) 本稿の UML モデリング言語で記述したモデルは、従来の慣習的な方法に比較して、ノード数が少ない。従来の慣習的な方法では、DIT による 1 対多の関連を明示的に表現するために、2 以上の下位エントリを記述することが多い。したがって、エントリの数が増大し、情報が冗長となる傾向がある。

(2)従来の慣習的な方法では、DIT 以外の関連を記述する場合、全体として煩雑な図になる傾向がある。そのため、図-24 のように、DIT 以外の関連を記述しないことが多く、重要な論理的な関連の情報が欠落する傾向がある。

5. おわりに

今後は、ディレクトリサービスを利用する新規プロジェクトにおいて、本稿で示したモデリング言語を適用し、設計効率の評価を実施する必要がある。

また、当モデリング言語で記述したモデルを入力とし、実装案に基づいた DIT の構築・検索を支援するアプリケーションを出力とするシステムを設計・試作する。その過程で課題の抽出とその解決を行い、実現可能性を検証する。さらに、試作成果をもとに、当モデリング言語の正当性の検証およびその生産性の評価を実施する。

参　考　文　献

[1]ITU-T　URL : http://www.itu.int/ITU-T/
[2]IETF　URL : http://www.ietf.org/
[3]UML　URL : http://www.uml.org/
[4]Tim Howes : “Understanding and Deploying Ldap Directory Services”, Macmillan Technical Pub, 2003
[5]河津 正人 他 : “LDAP ハンドブック”, ソフトリサーチセンター, 2002
[6]稲地 稔 : “OpenLDAP 入門”, 技術評論社, 2003
[7]グラディ・ブーチ (著), 羽生田 栄一 (翻訳) : “UML ユーザーガイド”, ピアソン・エデュケーション, 1999