UMLによる大学組織モデル化と規程文自動生成

UMLによる大学組織モデル化と規程文自動生成

持　尾　弘　司

Modeling University Structure with UML and Generating Official Regulations Automatically

Hiroshi MOCHIO

概要：本論文では，大学組織の一貫性・整合性を確立するために，モデル化言語UMLを用いて組織をモデル化し，そのモデルから組織を定義する一連の規程文を自動生成する方法を提案する。モデル化により一貫性・整合性の検証が容易となり，検証後のモデルを元に大学規程文を自動生成することにより，明確で矛盾のない組織定義が可能となる。

１．はじめに

今日，あらゆる分野で，組織は複雑化する現実への迅速な対応を求められている。学校組織もその例外ではなく，特に大学は，その性質上保守的にならざるを得ない義務教育，あるいはそれに準ずる感のある高等学校とは異なり，世情変化への対応を迫られる機会が多い。実際，伝統的な大学の運営形態は一部の超難関校を除いて不可能となっており，大多数の大学が頻繁な組織改革を行って要求に応えようと努力している。

その際問題となるのは，組織の一貫性や整合性である。一般に組織改革にあたっては，既存組織を一旦解体の上再構築することなど不可能であるから，現存形態を手直ししながら新しい要求に応えるべく改革して行くしかない。ところがこの手法は，しばしば重大な矛盾や陥穽を組織に持ち込むことになる。

なぜなら，ごく小規模な組織を除いて組織全体を見渡すことが極めて困難だからである。改変あるいは新規導入される規則や部署と全体との関係を見定めるのが容易ではない。規則を記述する条文が整備されるのは当然としても，それだけでは局所的検証はともかく大局的な検証は不可能である。結果はつぎはぎだらけ穴だらけの規則となり，規則を調べる度に堂々巡りに陥ったり，唐突な断定に悩まされることになる。

大学組織では問題はより深刻である。そもそも，大学組織は構造上の「捻れ」とでも言うべきものを内包しており，組織定義が複雑で分かりにくいものになりやすい。具体的には，教学組織と事務組織の相互関係がそれで，教授会に代表される教学側組織と事務側組織の関係は，時に大幅な矛盾を抱え込むことがある。しかも，現実に矛盾が生じていることを認識しているにもかかわらず，解決の糸口がつかめないという事態にしばしば遭遇する。すべては，組織全体を見渡す「俯瞰」の欠如が原因である。

他方，コンピュータの普及に伴い肥大化・複雑化したソフトウェア開発に目を向けてみると，似たような状況にあることが見て取れる。上記組織俯瞰の欠如は，ここではソフトウェア俯瞰の欠如となる。ソフトウェアは人や組織などに代わってコンピュータが行う処理を記述したプログラムおよびその付随物から構成されるから，現実の人や組織が孕む矛盾をそのまま取り込んでしまうことがある。加えて，設計やコーディング作業そのものの誤りによって，一貫性や整合性の不備を生じることもある。

コンピュータの一般利用が進むにつれ，どうにか遣繰りしてこのような問題を凌いできたソフトウェア開発の現場ではあるが，1990年代に入るとついに越え難い壁にぶつかることになる。膨大なコスト（この場合は人力）を掛けてもソフトウェアの品質が向上しない，つまり内部の矛盾を除去できない事態に見舞われたのである。この場合，とりあえず生産性を無視しても事態は一向改善しなかったという点が重要である。すなわち，何か本質的な方法論の変革が必要となったのであった。

今日「ソフトウェアの危機」と呼ばれる事態を収拾へと導いたのは，「オブ

ジェクト指向」へのパラダイムシフトである。「オブジェクト指向」を一言で説明するならば，「すべての対象を『物（オブジェクト)』と考え，オブジェクト間のコミュニケーションとそれに対する反応を記述する方法論」とでもなるだろうか[1)]。抽象的な処理概念を擬人化した手法で捉える点が非常に興味深い。言うまでもないが，これによって開発者はなじみ深い日常的対人関係を連想しながら，プログラムを把握することが可能となる。

ただし，それは局所的な見通しがよくなることを意味するだけで，一気に全体理解までが可能になる訳ではない。そこで登場するのが，オブジェクト指向パラダイムに従って対象をモデル化し表現するモデル化言語である[2),3),4)]。モデル化言語を用いて対象を抽象化し明示的に表現することにより，局所的な理解はもとより，全体を俯瞰，把握することができるようになる。

さらに，モデル化言語の利用はソフトウェア開発に止まるものではない。モデル化言語は「現実」を対象としてモデル化し記述するものであるから，ソフトウェア開発の如何に係らず，社会や組織のモデル化に利用可能である。ここに，一般組織の記述・分析（ビジネスモデリング）用途が生じることとなる[5)]。

本稿は，モデル化言語を用いて大学組織をモデル化，記述，検証し，さらにモデルを元に規程文まで自動生成する方法について提案する。これにより，一般的な法人組織よりもさらに複雑で混乱しやすい要素を内包する大学組織の改編整備に貢献することができる。

モデル化言語としては，UML を用いる。UML（Unified Modeling Language）は，それまで乱立気味であったモデル化言語群の利点を統合して90年代後半に誕生した，今日最も有望視される言語である。実際の開発現場でも既に多数の導入実績がある[2),3),4)]。

以下，大学組織の問題点を提示し，UML によるモデル化と検証，および規程文自動生成の概略について述べた後，本学の一部組織を例として具体的に提案内容を検討する。

２．大学組織の問題点

前述の通り，一般に組織の全体把握には困難が伴う。ある程度以上の規模を持つ組織であるならば事情はほぼ同じである。加えて，本学のような大学組織では，その組織構造上の特質から，特に一貫性・整合性の維持は容易ではない。

本学の組織を例に取ると，教授会選出による部長職の職権と各事務部局の長たる課長職の職権，あるいは事務局のトップである事務長と学部の長である学部長の関係等，分かりにくい関係が散見される。また，人事権が一般法人のように一元化されておらず，教育職員と事務職員では採用審査の手順も全く異なる。さらには，設備や予算の管轄権限が学園本部組織との間で複雑に絡み合っており，実際の権限執行にあたってはしばしば混乱をきたすことがある。

このように，元々困難な一貫性・整合性の維持は，組織の特殊性からより一層覚束なくなっている。そもそも，個別の委員会や日常的な職務の定義も含め，組織のすべては「学園例規集」に明記される建前になってはいるものの，実際には記述漏れおよび術語の不統一があったり，記述内容に矛盾があったりで，一貫性・整合性とはほど遠い状態である。

現状の由来は，一つには，本学が短大を出発点として順次拡大，大学体制にまで発展して来た事にある。その間，組織展開に伴って例規集の記述も拡大，複雑化したが，一貫性・整合性の統合的な検証手段を持ち得ぬまま例規集への規程文追加作業が行われた結果，今日の惨憺たる状況に至ったものと推察される。

いずれにせよ，大学組織は一般法人組織以上に一貫性・整合性検証・維持のための合理的な手段を必要とする。さらには，組織を記述し定義する矛盾のない規程文を過不足なく生成する仕組みが強く望まれる。

３．モデル化言語 UML とそのビジネスモデリングへの拡張

UML（Unified Modeling Language）は，1997年11月に OMG（Object

Management Group）によって標準化された，モデル化言語である[2),3),4)]。その基本はダイアグラムによる表現である。UML は九種類のダイアグラムから構成されているが，その内「クラス図」「ユースケース図」「シーケンス図」が多用される。

オブジェクト指向では，物（オブジェクト）の集合（クラス）を基本単位と考えるが，クラス図はこのクラス間の関係を示す。ビジネスモデリングでは，クラスは組織，情報，文書，製品等を表すと考えられる。一方，ユースケース図は利用者の視点に立ってシステムの機能を記述するもので，平文で定義され

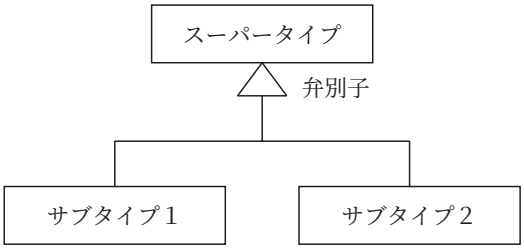

図１－１　クラス図（汎化）

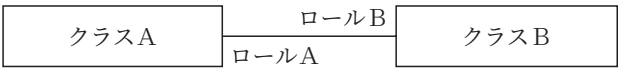

図１－２　クラス図（関連）

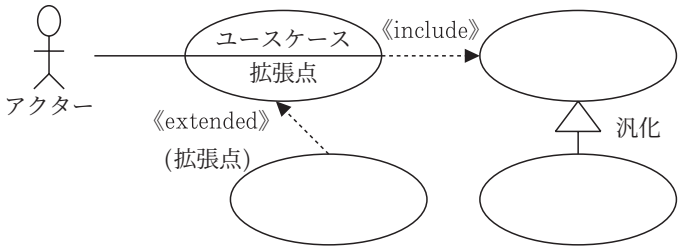

図１－３　ユースケース図

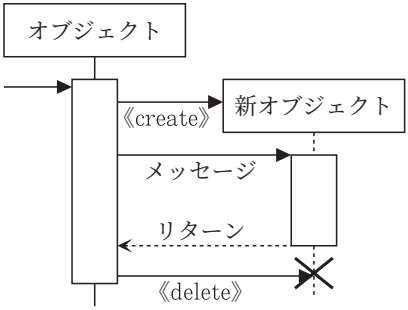

図1－4　シーケンス図

るという特徴がある。シーケンス図はオブジェクト間でやり取りされるメッセージを時系列で表したものである。

さて，ビジネスモデリングを行うためには，以下のような概念が必要となる[5)]。

1．**リソース**：人，情報，材料，製品など，そのビジネスで使用または生産されるもの。リソースはプロセスによって操作される。
2．**プロセス**：ビジネスにおける操作・処理のこと。プロセスはルールによって決定され，リソースの状態に作用する。
3．**ゴール**：ビジネスが達成しようとしている目標。ゴールはサブゴールに分解され，プロセスによって達成される。
4．**ルール**：ビジネスを定義あるいは制約する言明。

ビジネスモデリングとはこれら概念の相互関係を明確にすることに他ならない。

ところで，UML によるモデル化の本質は「抽象化」と「可視化」にあるが，これらビジネスモデリングに特有の概念を UML で可視化するには若干の拡張が必要である。UML は元々拡張が容易な仕様になっており，新しい図や要素を簡単に加えることができる。ビジネスモデリングのための拡張は Eriksson-

Penker によって行われた。

さらに，ルールの記述には制約を定義する言語が必要となる。OCL（Object Constraint Language）は1995年に IBM のグループによって開発された形式言語で，version1.1より正式に UML の一部となった。OCL の記述力は強力で，どんな制約も明確に定義することができる。

４．ビジネスパターンと規程文の自動生成

ビジネスモデリングにおいては，多くの記述が既存のものと同型になる。ここで同型とは，具体的オブジェクトは異なるがクラス構造やその関係が同じであることを意味する。このような記述の型はパターンと呼ばれ，ビジネスモデルの大部分はパターンの組み合わせで記述可能である[6),7)]。パターンの構造はクラス図を用いて定義され，その振る舞いはシーケンス図によって記述される。

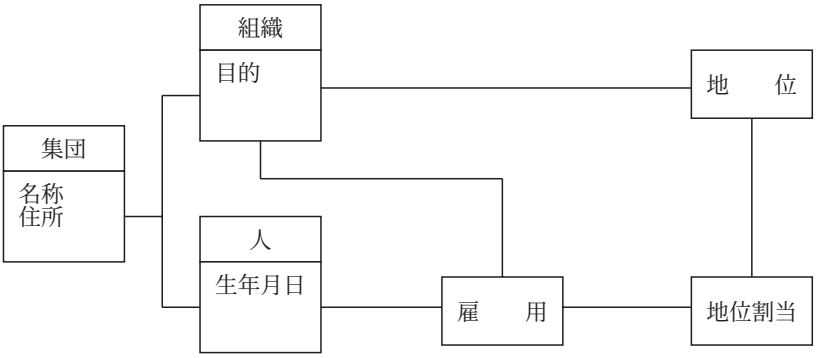

図２　雇用パターンの構造を示すクラス図

例えば図２は David C. Hay[7)]による「雇用パターン(Employment Pattern)」の構造を示すクラス図である。「集団」は人と組織の両方を記述する抽象クラスであり，住所や名前などの属性で拡張可能である。また，「組織」と「人」は集団のサブクラスであって，その間の関係が「雇用」クラスである。雇用には開始日，終了日，雇用形式などの属性がある。「地位」は組織によって定義

され，給与や業務命令といった属性を持ち，「地位の割当」は人と地位の関係を定義する。

雇用パターンはある人とその雇用主の関係を構造化する。これを用いたモデル化により組織の雇用に関するすべての情報を遺漏なく記述することができ，さらにその一貫性・整合性の検討を行うことが可能となる。例えば，ある地位の任免権限が重複する場合は，複数の地位割当オブジェクトからリンクが生じることになるが，これはプログラムによって自動検出可能である。

さて，ここでモデルから規程文を自動生成する仕組みについて考えてみることにする。ビジネスモデルの大部分は上記のようなパターンの組み合わせで記述できるのであった。であれば，パターン（＝決まった型）に対応する規程文テンプレートを準備することで，モデルから組織定義の規程文を生成できるはずである。具体的には，規程文テンプレートの該当部分にインスタンス（オブジェクトの実体）や属性値を当てはめるだけでよい。

UML のダイアグラムはコンピュータ上で文字記号表現（例えばラベル付きグラフ）することが可能だから，このような対応関係を文字記号の対応関係として予めコンピュータに蓄積しておけば，規程文は自動生成されることになる。必要な操作は，インターフェイスに従ってダイアグラムを作成することだけである。

例えば雇用パターンの場合，対応する規程文テンプレートは以下のようなものになるであろう。

> [住所：集団] にある [名称：集団] は，[目的：組織] のために [生年月日：人] 生まれの [名前：人] を [職名：地位] として [開始日：雇用] より [終了日：雇用] まで雇用する。

5．モデルの実例，および対応する規程文

UML によるビジネスモデル記述の具体例を見てみよう。次に示すのは雇用

パターンを用いた，本学学長の選出規程モデル化（オブジェクト図）である。

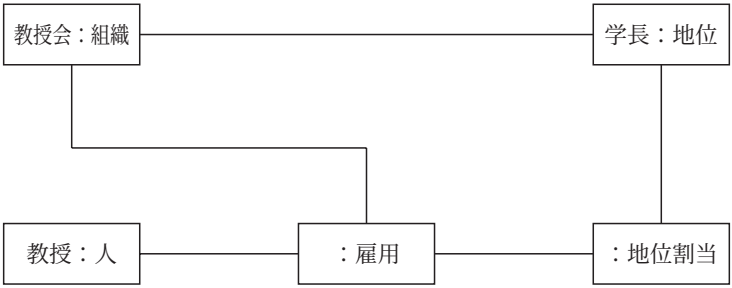

図３　学長選出規程のモデル化

これに対応して自動生成されるべき規程文は，例えば

> [教授会（名称：組織)] は [大学運営（目的：組織)] のために [本学専任教授または助教授（資格：人)] を [学長（職名：地位)] として [選挙（方法：地位の割当)] により指名する。任期は [三年（任期：雇用)] とする。

となる。

実際には，教授会の人事権限は複数多岐に渡るから，このようなオブジェクト図が複数必要である。したがって，モデルデータ内部では，モデル化が進むにつれて「教授会オブジェクト」から様々な「地位オブジェクト」へリンクが伸びて行くことになる。前述の通り，一貫性・整合性の検証は，このリンクを辿ることが基本である。

６．おわりに

本稿は，モデル化言語 UML を用いたビジネスモデリング手法を大学組織に適用し，一貫性・整合性のある組織定義とそれを明記する規程文の自動生成を行う方法について基本的な枠組みを提示した。

今後は，まず実際の例規集に基づいてモデルを記述することが必要である。

また，UML モデルから規程文を自動生成するプログラムの実装を行いたい。さらに，UML モデルを元に実動モデルプログラムを作り，対象とした大学組織の一貫性・整合性を検証する手法（モデルチェッキング）も試みてみたい。

他方，本稿提案のモデルを反映した規程文自動生成は，ソフトウェア開発の現場に応用することができる。ソフトウェア開発工程において UML によるモデル化は上流工程に属するが，例えば仕様策定に際して UML 記述と自然言語記述が並記されることは，関係者全員による確実な情報共有の観点からも望ましい。UML モデルの記述のみで自然言語による仕様文書までが作成できれば，常にそのような望ましい状態を維持することが可能となる。

参 考 文 献

1）Graham, Ian. “Object-Oriented Methods. Reading.” Reading, MA: Addison-Wesley, 1994.

2）Eriksson, Hans-Erik and Magnus Penker. “UML Toolkit.” John Wiley & Sons, Inc., 1996.

3）Booch, Grady, James Rumbaugh, and Ivar Jacobson. “The Unified Modeling Language Users Guide.” Reading, MA: Addison-Wesley, 1998.

4）Fowler, Martin. “UML Distilled, 2nd Edition.” Addison-Wesley, 1994.

5）Eriksson, Hans-Erik and Magnus Penker. “Bussiness Modeling With UML.” John Wiley & Sons, Inc., 2000.

6）Fowler, Martin. “Analysis Patterns: Reusable Object Models.” Addison-Wesley, 1997.

7）Hay, C David. “Data Model Patterns: Conventions of Thought.” Dorset House, 1996.