B-020

大規模システム開発のための階層的モデル化技法
Layered Modeling Method for Developing Large-Scale Systems

山田隆弘†
Takahiro Yamada

1. はじめに

システム（ここでは、ソフトウェアを中心とし、場合によってはハードウェアも含むシステムを意味するものとする）の開発を行うときに Unified Modeling Language (UML)[1][2]等のモデル言語を用いてシステムの構造や振る舞いを記述する試みが増えている[3][4]。UML のようなモデル言語は、図の記法や構成法を定めているだけであるが、これをシステム開発の方法論と結びつけ、システム開発のどのような局面で UML のどの記法をどのように使うべきかを定めた方法論として Unified Process (UP)[5][6]が知られている。UML は多くの記法を定めているので、UML の記法の統一的な使い方を定めない限り、システム毎に UML の記法の使い方が異なってしまい、複数のシステム間でのモデルの比較、統合、共有が困難になる。様々なシステムで UP を採用すれば、どのシステムでも同じことは同様の記法でモデル化されることになり、複数のシステム間でのモデルの比較、統合、共有が容易に行えるようになる。

ところで、UP は小規模のシステム開発には適しているが、後述するように大規模システムの開発には必ずしも適していないように思える。本論文では、UP を多少改変することにより、大規模システムの開発に適したモデル化技法の提案を行う。

2. 大規模システム開発におけるモデル化技法の問題点

大規模システムを開発する場合は、設計・製造・試験を容易にするために一つのシステムを複数のサブシステムに分割し、個々のサブシステムはそれぞれ独立に開発される。この場合、まずシステムに対する要求（システムは何を行うべきか）に基づいて、そのシステムが含むべきサブシステムが設定され、それぞれのサブシステムに対する要求（サブシステムは何を行うべきか）とサブシステム間のインタフェースとが規定される。各サブシステムは、そのサブシステムに対する要求と他のサブシステムとのインタフェースを満たすように開発される。システムの規模が非常に大きな場合は、それぞれのサブシステムがさらに複数のコンポーネントに分割され、コンポーネント毎に独立に開発される。このような分割の階層の数はシステム規模によって異なるが、システム分割の階層の一例を図１に示す。例えば、人工衛星の場合は、図１のような３階層に分けて開発するのが普通である。（注：本論文の階層は、システム～サブシステム～コンポーネントというような包含関係の階層であり、アプリケーション～ミドルウェア～オペレーティングシステムというような機能の上下関係の階層とは異なっていることに注意されたい。）

UML や UP においても上に示した階層的システム開発に適用できる概念が存在している。例えば、システムに対する要求は UML のユースケース図とユースケース定義（文章として記述される）として表される。ユースケースは、システムがアクタ（システム利用者や外部システム）に対して何を行うかを表すものである。また、システムとアクタの間の相互作用を表すために UML のシーケンス図が使用され、これはユースケース実現と呼ばれる。これに対して、サブシステムやコンポーネントに対する要求は、UP や UML では主にインタフェースという概念を用いて表される。インタフェースとは、何を行うかをオペレーションの集合として定義したものである。ユースケースとインタフェースとでは概念が異なっているが、それらの間の関係についての厳密な規定は存在しない。システムもサブシステムもコンポーネントも、分割のレベルが異なっているだけであるのに、それらの要求の規定方法が異なるのは不便である。なぜならば、サブシステムも、それを開発するときには一つのシステムとして開発されるし、また、一つのシステムが他のシステムと組み合わされ、さらに大きなシステムを構成することもあり得るからである。

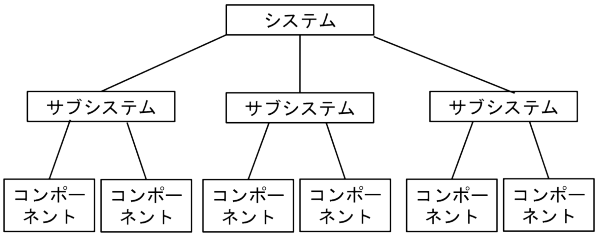

図１　システム分割の階層の例

そこで、本論文では、システムもサブシステムもコンポーネントも同様な手法でモデル化でき、それらを自由に組み合わせてシステムのモデルを構成できるようなモデル化技法を提案する。

3. 大規模システムのモデル化技法

ここで提案するモデル化技法においては、システムもサブシステムもコンポーネントも同様に扱い、それらを総称してシステム要素と呼ぶ。

システム要素に対する要求（システム要素は何を行うべきか）は、そのシステム要素が提供すべきインタフェースとして規定される。このインタフェースは、UML で規定しているインタフェースの概念と同一であり、そのインタフェースを通じて使用できるオペレーションの集合を規定したものである。また、各々のオペレーションが何を行うべきかを規定するために、オペレーション毎にオペレーションコントラクト（[4]の第 11 章参照）も規定する。オペレーションコントラクトは、そのオペレーションが実行されたときに何が起こるかをドメインモデルの要素を使って表したものである。ドメインモデル（[4]の第 9 章参照）は、

†宇宙航空研究開発機構宇宙科学研究所, JAXA/ISAS

その分野の基本概念を UML のクラス図で表したものである。さらに、これらのオペレーションの使い方を示すために、シーケンス図を使用する（これは、[4]の第 10 章で述べられているシステムシーケンス図に相当する）。

この技法を用いてシステムに対する要求を UML で表した例を図２に示す。この図は、システム S はインタフェース IA と IB を提供すべきことを示している。これらのインタフェースは、UML の規則に従ってオペレーションの集合として定義される。さらに、これらのオペレーションは、オペレーションコントラクトとシーケンス図とによって規定される。インタフェース定義とシーケンス図の例を図３に示す。

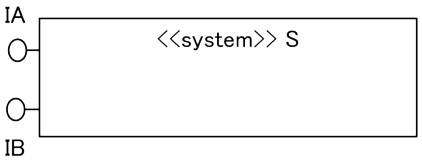

図２　システム要求の記述例

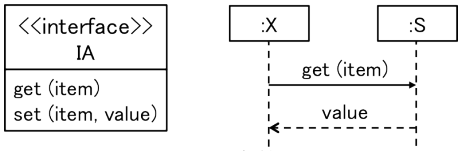

図３　インタフェース定義とシーケンス図の例

上記のシステム S をサブシステムに分割した例を図４に示す。この図は、システム S が３つのサブシステム A、B、C に分割されることを示し、さらにサブシステム間のインタフェースについて以下のことを示している。

・サブシステム A はインタフェース IA を提供し、インタフェース IC1 を使用する

・サブシステム B はインタフェース IB を提供し、インタフェース IC2 を使用する

・サブシステム C はインタフェース IC1 と IC2 を提供する。

サブシステムをコンポーネントに分割する場合も、これと同一の方法が適用される。

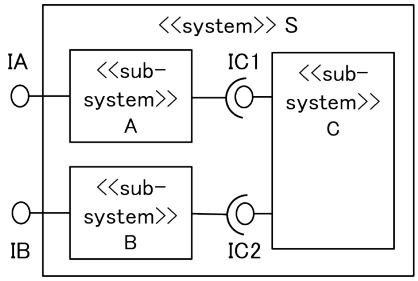

図４　サブシステム要求の記述例

4. ユースケースの行方

本技法と UP との大きな違いは、本技法ではユースケースを用いないことである。しかし、インタフェースを用いてユースケースと等価な情報を記述することができる。それを示したものが図５である。この図において、X はシステム S の利用者（人あるいは外部システム）であり、この図は、システム S が提供するインタフェース IA と IB を利用者 X が使用することを表している。上で述べたインタフェース定義とオペレーションコントラクトとシーケンス図とを組み合わせることによってユースケースおよびユースケース実現と等価な情報を表すことができる。ユースケースが既に存在している場合は、ユースケースをインタフェースに変換しても良い。

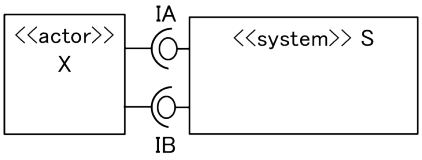

図５　ユースケースと等価なインタフェスの表現

5. 本技法の特徴

本技法の特徴は、システムもサブシステムもコンポーネントもシステム要素として全く同じ方法によってモデル化できることである。具体的には、システム要素に対する要求は、そのシステム要素が提供すべきインタフェース（オペレーションの集合）として規定される。また、ユースケースを廃することにより、システム要求とサブシステム要求との間の関係を厳密に表すことができる（図４参照）。これにより、システム要素（これにはシステム全体も含む）のモデルの統合や共有や再利用が容易に行えるようになる。インタフェース定義やオペレーションコントラクトは、システム要素のブラックボックス試験に対する要求としても使用できる。

インタフェースの構成要素として定義されたオペレーションを実現するための方法は、従来の UP とほぼ同様であるが、これについても今後詳細な検討を行いたい。

6. おわりに

本論文では、大規模システムの開発に適した階層的なモデル化技法の提案を行った。本論文では、システム要素に対する要求（何を行うべきか）を規定する方法に焦点を当てたが、今後は本技法をシステム開発全般に適用するためにさらに詳細な検討を行う予定である。

参考文献

[1] Rumbaugh, J., Jacobson, I., Booch, G.: *The Unified Modeling Language Reference Manual*, Second Edition, Addison-Wesley (2004).

[2] Booch, G., Rumbaugh, J., Jacobson, I.: *The Unified Modeling Language User Guide*, Second Edition, Addison-Wesley (2005).

[3] Blaha, M., Rumbaugh, J.: *Object-Oriented Modeling and Design with UML*, Second Edition, Prentice Hall (2005).

[4] Larman, C.: *Applying UML and Patterns*, Third Edition, Prentice Hall (2004).

[5] Jacobson, I., Booch, G., Rumbaugh, J.: *Unified Software Development Process*, Addison-Wesley (1999).

[6] Arlow, J., Neustadt, I.: *UML 2 and the Unified Process*, Second Edition, Addison-Wesley (2005).