通信ソフトウェアの仕様記述の理解性向上に関する一考察

4E－8

宇都宮 栄二　新田 文雄　齋藤 博徳　伊藤 篤
国際電信電話株式会社 研究所

1. はじめに

一般に、プログラムと同様に通信ソフトウェア仕様も機能に関した階層化(階層分割)を行なうことにより、その理解性の向上が図られるとされている。たとえば、ITU-T の仕様記述言語 SDL[1]では、System、Block、Process、Substructure などの機能単位で仕様を記述する階層分割が提案されている[2]。しかし、意味のない階層化や階層数の増加により、階層分割しない場合に比べ、むしろ、階層分割した仕様の理解性が低くなってしまうという問題がある。本稿では、SDL を用い記述された通信ソフトウェア仕様の理解性の向上を目的とした階層的な仕様記述法を提案し、その方法の利点および問題点について考察した結果を述べる。

2. 仕様と階層分割

2.1 仕様記述

通信ソフトウェア仕様は、(複数の)機能ブロック、機能ブロック間および機能ブロックと環境(外界)との間で授受される信号および信号ルートから構成される静的仕様要素と、信号の受信等による機能ブロック内の状態遷移の定義から構成される動的仕様要素により記述される。

2.2 階層分割の目的

通信ソフトウェア仕様を階層分割する目的は、複雑な通信ソフトウェア仕様を単純な機能ブロックに分けることによりその複雑さを取り除き、仕様の理解性の向上を図ること、および、独立した小さな機能ブロックに分割することにより機能ブロック単位の仕様記述とこれに基づくプログラム作成等の作業を並行して行なうことを可能とすることである。

2.3 SDL による階層分割

SDL における階層は、主に、仕様記述対象の全体を表わす system、system を静的な機能で分割した単位である block、block を分割した実際の動作単位である process から構成される。また、より詳細な分割を行なうために、substructure により block を分割することができる。階層分割した SDL による通信ソフトウェア仕様の例を、図 1 に示す。図 1 の例では、通信ソフトウェア仕様を階層分割し、動的仕様要素である process は最下位ブロックにのみ存在する。その他のブロックには静的仕様要素のみが定義されている。

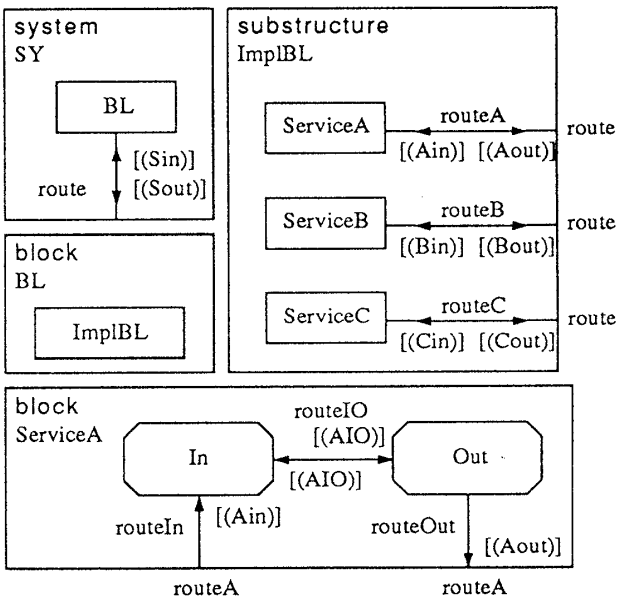

図 1: SDL による仕様階層化の例

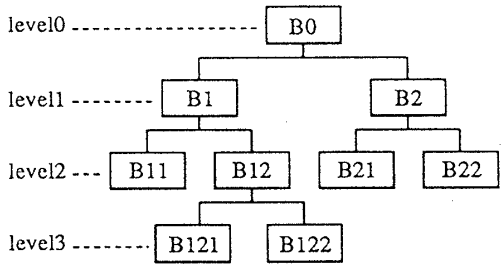

図 2: ブロックツリーダイアグラムによる階層表示例

3. 階層分割の問題点

通信ソフトウェア仕様を階層分割した場合、その目的に反して理解性が低くなる場合がある。

図 2 に、階層分割の例を、ブロックツリーダイアグラムで示す。図 2 において、動的仕様要素が定義されているブロックは、level2 の B11, B21, B22 および level3 の B121, B122 のみである。その他のブロックは静的仕様要素のみ定義されている。したがって、level1 のブロック B1 における動作を知りたい場合、ブロック B11,

---

"A Study on Understandability of Specifications for Telecommunication Software"
Eiji UTSUNOMIYA, Fumio NITTA, Hironori SAITO and Atsushi ITO
KDD R & D Laboratories

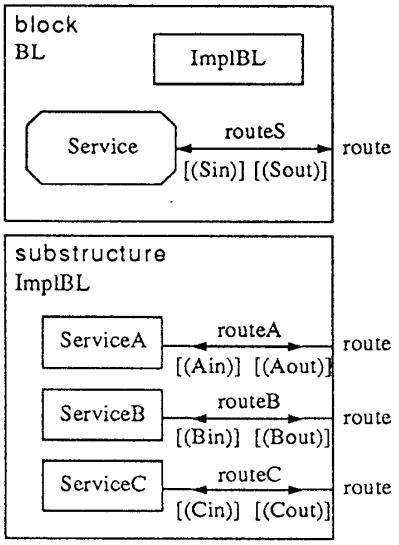

図 3: 中間ブロックへの動的仕様要素記述例

B121, B122 に定義されている動的仕様要素を参照する必要がある。

したがって、一般に、階層分割された仕様では、中間ブロック (末端ではないブロックのこと。図 2 において、静的仕様要素のみが定義されている B0, B1, B2, B12) における動作概要を得るために、その中間ブロックに関係する動的仕様要素が定義された複数の最下位ブロックをすべて参照・解析する必要があり、かなりの労力を必要とする。このような階層分割においては、多階層化される通信ソフトウェア仕様の理解性は、かなり低くなると考えられる。

4. 階層分割における理解性の向上

前章で述べた理解性に関する問題点を解決するために、各中間ブロックにおいても、そのブロックに応じた詳細度の動作を表す動的仕様要素を記述することを提案する。

すなわち、図 2 の例において、動的仕様要素は、最下位のブロックである B11, B21, B22, B121, B122 にのみ定義されているが、最下位でないブロック B0, B1, B2, B12 にも動的仕様要素を定義するということを意味する。

この記述方法を実際の SDL に適応した例を、図 3 に示す。従来の方法では、block BL には、substructure ImplBL のみが定義されるが、ここでは、同時に、このブロックにおける動的仕様を process Service として記述する。具体的には、このブロックの process Service では、substructure ImplBL と外界との信号の授受を中心に動作が記述されることになる。すなわち、ブロックに定義されている process は、そのブロックで必要な詳細度の動作を表し、substructure はその process の動作を、より詳細なレベルで定義する。

このように、ブロック (block) に動的仕様要素 (process) と静的仕様要素 (substructure) を同時に定義することは、ITU-T 勧告上の規定によれば、SDL の本来の記述手法ではない。しかし、シンタックス上では記述可能である。また、SDL を対象にした記述支援ツールにおいても一般的になんら問題はない。

5. 考察

前章で提案した記述法によると、動的仕様要素がすべてのブロックに記述されているため、特に必要のない限り下位ブロックへの参照が少くなり、理解性が向上する。

しかし、この記述法の問題点として、異なるブロックの動的仕様要素間の意味上の整合性を保持しなければならない。これら動的仕様要素間の整合性を保証するためには、最下位の動的仕様要素を合成した動的仕様と中間ブロックにおける動的仕様の等価性を判定する必要があり、これを高速・効率的に機械処理する技術の実現は、現状では困難である。このため、本仕様記述法には、以下のような制約がある。

- 動的仕様要素は、そのブロックが分割されるまで (つまり、そのブロックが中間ブロックとなるまで)、形式的な仕様として取り扱われる。しかし、中間ブロックより下位のブロックに動的仕様が記述された時点で、中間ブロックに記述された動的仕様要素は、形式的な仕様として扱うことができなくなる。

しかし、このような制約下でも、中間ブロックに記述された動的仕様要素はインフォーマルな記述として、理解性の向上には十分有用であると考えられる。

6. おわりに

本稿では、通信ソフトウェア仕様を階層的に作成するときの理解性に関する問題点を述べた。これをもとに仕様における階層の意味を示し、仕様の理解性を向上させるためには、最下位ブロックのみでなく、すべてのブロックに、動的仕様要素 (状態遷移) の定義が必要であることを示した。最後に日頃御指導頂く KDD 研究所 浦野所長、真家次長、若原交換グループリーダに感謝します。

参考文献

[1] ITU-T: 'Recommendation Z.100, Specification and Description Language (SDL)', 1988.

[2] F. Belina, D. Hogrefe, A. Sarma: 'SDL with APPLICATIONS from PROTOCOL SPECIFICATION', Prentice Hall International, 1991.