リアルタイム組み込みソフトウェアの用語の木

小川清† 斉藤直希† 吉川直邦‡伊藤正樹‡ 後田直樹‡ 藩建華‡

†名古屋市工業研〒456-0058 名古屋市熱田区六番 3-4-41
‡大同工業大〒457-8530 名古屋市南区滝春町１０－３
E-mail: †{ogawa.kiyoshi,saito.naoki}@nmiri.city.Nagoya.jp ‡{d99e124,d99e171,d99d016,d99d023}@daido-it.ac.jp

あらまし　μITRON、TOPPERSを例にリアルタイム組み込みソフトウェア開発のための用語の木を作成した.
キーワード　μITRON，TOPPERS，リアルタイムOS，組み込みソフトウェア、用語の木

Thesaurus on Real Time Embedded software

OGAWA Kiyoshi,SAITO Naoki† YOSHIKAWA Naokuni,ITO Masaki,USHIRODA Naoki, HAN Kenka‡

† Nagoya Municipal Industrial Research Institute 3-4-41 Rokuban, Atsuyta-ku, Nagoya, 456-0058 Japan
‡ Daido Institute of Technology　10-3 takiharu-cho, minami-ku, 457-8530 Nagoya Japan
E-mail: †{ogawa.kiyoshi,saito.naoki}@nmiri.city.Nagoya.jp ‡{d99e124,d99e171,d99d016,d99d023}@daido-it.ac.jp

Abstract　Thesaurus on Real Time Embedded software using microITRON and TOPPERS.
Keyword　microITRON, TOPPERS, Real-time OS, Embedded software, Thesaurus

1. はじめに

組込みソフトウェアの開発者で、PCの汎用OSに馴染んでから組込み開発に従事することになる人たちが増えている。PC（パソコン）は汎用OSを組み込んで販売している場合が多い。そのため、汎用OS上でのソフトウェア技術者は、ハードウェアの知識、場合によってはOS、コンパイラの知識なしに、ソフトウェア開発を行うことがある。

また、大量に普及しているコンピュータ関連の雑誌・書籍では、略語,カタカナ語が多用され、固有名詞と標準用語とが混在し、同一文書内で、用語の示す範囲が文脈によって異なることがある。ソフトウェア技術者教育で、OS、コンパイラ言語の知識を教えずに、OS、コンパイラの利用者としての教育が行われている場合がある。ソフトウェア技術者で、開発環境、開発対象の違いにより、用語の意味が異なる場合がある。

名古屋市工業研究所では、大同工業大学との大学生に対する教育と、中小企業との技術者の教育を取り組んでおり、組み込みソフトウェア技術者教育において、用語の木を作成しながら、用語における課題を検討してきた。意味ネットワークは、それぞれの立場によって、異なり、用意に木を構成することはできない。しかし、リアルタイムOSを中心に、リアルタイムOSの開発者とリアルタイムOSの利用者（応用ソフトの開発者）に絞って、用語の木を作成することを検討する。

特定の開発目的のためには、リアルタイムOSを用いない場合もある。その際にも、リアルタイムOSで検討されている特性、機能には利用可能が概念も多い。

本研究では、リアルタイムOSを前提に、組み込み開発を行う際に必要な用語の木の作り方をμITRONとTOPPERSの文書、プログラムを基に、考慮する点を明確にした。

2. リアルタイムOS

組込みシステムで、リアルタイムOSを利用するには、メモリ容量が制約となることがある。ソフトウェア技術者が組み込みシステムを開発する場合に、リアルタイムOSの機能が一つの境界線になりうる。リアルタイムOSを利用しない場合でも、リアルタイムOSで実装している技術の部分的な利用が可能な場合があるからである。

リアルタイムOSの仕様としては、μITRON[5]がよく参照されており、そのオープンソースとしてTOPPERS/JSPカーネル[6]がある。TOPPERS/JSPカーネルは，豊橋技術科学大学組込みリアルタイムシステム研究室を中心とするTOPPERSプロジェクトにおいて開発したμITRON4.0仕様のStandardプロファイルに準拠したリアルタイムカーネルである.

名古屋市工業研究所の研修では、2002年度から組み込みソフトウェア研修を実施しており、大同工業大学の学生への研修としてTOPPERS/JSPカーネルを取り組んでいる。企業向けの組み込みソフトウェア研修の一貫として、TOPPERS/JSPカーネルの普及の切り口と

して、μITRON 文書と合わせて用語の木を作成しながら利用することを検討している。

3 組み込み開発における分類

組込み技術者教育における用語の課題を、開発における立場の違いによって分類するとともに、開発の対象、システムの特性、規模、開発で用いる言語・環境ごとに分類を作成した。

2.1 対象システムの範囲

組込みシステムで、焦点を当てるシステムの範囲が、どこまでかによって、検討する事項が異なる。例えば、組み込むシステムの機械装置との比率、電源を含める必要があるかどうか、ノイズ・発熱を考慮する必要があるかどうか。ここでは、５つに分類した。開発ソフトウェアは,ENV2 開発言語と関連がある。

TRG.1 電源
TRG2 対象装置
TRG3 コンピュータ周辺回路
TRG4 開発環境、開発用ソフトウェア
TRG5 実行ソフトウェア

電源には、TRG1.1 商用電源、TRG1.2 電池、TRG1.3 発電機という細分類を付与することができる。対象装置は、例えば、TRG2.1 家庭電化製品、TRG2.2 通信機器, TRG2.3 事務機器, TRG2.4 輸送機械、TRG2.5 工作機械、TRG2.6 建機 TRG2.7 化学機器、TRG2.8 医療機器、TRG2.9 食品機械、TRG2.10 繊維機械、TRG11 映像機器、TRG2.12 音響機器のように、細分類ができる。電子機器は、複合化または単機能化により、製品分類が数年で変化する場合があるため、どのような分類がよいかは、システムの特性と合わせて検討するとよい。

2.2 システムの特性

システムの特性は、ISO/IEC 9126 の品質特性に基づいて分類した。リアルタイム性は、効率性の１つであるが、時間は他の単位との違いから、リアルタイム性を取り上げた。

CHR1 機能性
CHR2 信頼性
CHR3 使用性
CHR4 リアルタイム性
CHR5 資源効率性
CHR6 保守性
CHR7 移植性
CHR8 利用時の品質

他の副特性もすべて分類するとよい。例えば、利用時の品質には、CHR8.1 有効性、CHR8.2 生産性、CHR8.3 安全性、CHR8.4 満足性という副特性を細分類にしてもよい。各特性、副特性には、具体的な指標を考え、細々分類とするとよい。リアルタイム性では、必要とする時間の単位と、OS を利用するかどうかの区分が考えられる。

2.3 開発規模

開発対象が、単品生産か、大量生産か。開発要員が、一人での開発か、１００人以上での開発か。開発環境を専用に購入できる予算規模か、他のシステムの開発環境を流用するか。開発規模、期間、費用によって、リアルタイム OS を利用するかどうかの判断をすることがあるかもしれない。

SCL1 開発対象の生産個数
SCL.2 開発要員数
SCL.3 開発期間
SCL4 開発費用

開発期間には、SCL2.1 計画期間、SCL2.2 試験期間、ＳCL2.3 開発期間を含む。試験期間が開発期間よりも若い番号にしたのは、開発の入力の試験が、対象の開発、試験に先立つと考えたからである。

2.4 開発環境

開発環境は、試験環境との関係、実行環境との関係を明確にするとよい。

ENV1 開発用 OS
ENV2 開発用言語
ENV3 仕様記述言語
ENV4 開発環境
ENV5 試験環境
ENV6 実行環境

例えば、仕様記述言語には、ENV2.1 英語、ENV2.2 日本語、ENV2.3　仕様記述専門用語を細分類する。

4　用語の木

４．１　略号

略語には同一の表記で、複数のフルスペルに対応しているものがある。類似の表記で、同一の分野で異なる意味の略号もある。略号は、何度も出てくる用語を短縮し、場所と時間を節約することが目的である。また、数字のコードに比べて、意味が取りやすい利点もある。

本分類でも略号を用いているのは、数字のコードは意味がわかりにくいためである。英語の略号よりも、漢字によるコードも有用かもしれない。例えば、TRG：対象、CHR：特性、SCL：規模、ENV：環境という具合である。場合によっては、漢字による分類記号の作成を検討する。

4．2 カタカナ語

カタカナ語では、文脈に応じた使い分けがされている場合には、厳密な意味を持つ場合と、意味の重複した用語が存在する場合がある。これらの用語を構造的に定義することにより、相互の意思疎通を円滑にすることができる。カタカナ語の利点は、英語による技術を、早期に取り入れることが可能なことである。従来利用していた体系とは切り離して考察できることである。欠点としては、意味の重複・欠落、処理、管理の重複・欠落が生じたり、誤解を生むところがある。[7]

カタカナ語には、英語のフルスペルを付け、英語、日本語での上下関係の違いを付記するとよい。

4.3 検討方法

本研究では、用語の木の表記方法の検討を行い、あいまいな知識しか持っていない段階から順に作成できるよう検討し提案する。英語文献、日本語文献、プログラムの３つを対象として、それぞれ用語の木を作成する。

4.4 日本語の用語の木

日本語の場合には、単語の切り出しが難しい。例えば、電子ファイルであっても、簡単な方法が利用しずらい。テキスト処理として、漢字、ひらがな、カタカナ、アルファベットの区切りで切り出して一次資料とするか、英語で作成した辞書を翻訳して、見だし語とする。

4.5 プログラムの用語の木

プログラム中の関数、変数の関係を明確にするために、英語で用いた方法と同様に、単語の切り出しをする。関数、変数、定数に記号を用いている場合には、記号を含めておくとよいかどうかを検討する。用語の分析の段階では、記号をすべて削除した上で、命名規則の妥当性を確認するのに用いることができる。プログラムの理解のためには、関数、変数、定数の記号を含めた用語の木を作成する。

5 英語の用語の木の作成

5.1 単語切り出し文献の特定

標準文書、仕様定義文書から単語を切り出す。用語体系が違う文書の場合には、独立に切り出し、それぞれを比較するようにする。

5.2 英単語の切り出し（抽出）

全単語の集計表を作成する。大文字小文字を変換し、記号を削除する。AWK のスクリプトで単語の出現回数を計算する。複数形、過去形、形容詞の副詞形、動詞の名詞形を集約するプログラムを使って、集約していた。そのため１度しか出現しないものには、変化形が含まれている。それ以外では、図、表、事例に登場するだけの用語の可能性がある。また、定義のみで使っている場合もある。仕様書の性格からすれば、定義とその利用の２度以上出現する用語が多いはずである。ITRON 仕様書の例では、3600 語（関数、変数、符号、固有名詞を含む）のうち３分の１が出現頻度１回であるが、その半分以上が、固有名詞か符号類である。

自然言語における出現頻度１回の用語プログラミング言語では、構文解析で、左辺値としてしか出現しない、定義されているが使われていない、定義されずに使われているという警告に相当する可能性を持っている。

5.3 略号のフルスペル付与

略号は、そのフルスペルが何かを調べる。英日の機械翻訳ソフトにかけ、単語の切り出し方、用語の文脈依存性を検討する。文脈によっては略号として用いているものと、フルスペルとして用いているもの、プログラム上の定数、変数、関数として用いているものがある。

5.4 STOP WORD の除去

A, the,のように、英文でよく出現する用語は、stop word として用語から削除する。

表１　Stop word の例

単語	頻度	単語	頻度
the	8033	in	1674
is	2611	and	1243
of	2290	for	962
to	1930	be	949
A	1899	from	783

表１は、ITRON 仕様書における Stop Word の多い方から１０語である。文書中に出現する約 3600 語のうち、この１０語だけで、文書中の約 99000 語（関数、変数、符号を含む）の２２％を占めている。

何が STOP　WORD かは、文書の性格による。例えば、in がアセンブラで命令語である場合には、STOP

WORD として除去しない方がよい。頻度を利用する場合には、頻度の計算には、grep(global regular expression print)で用いている KWIC(Key Word In Context)の方法により、文脈を判定するとよい。

5.5 固有名詞、規格用語かどうかの判断

専門用語かどうかを判断する前に、登録商標をはじめとする固有名詞、定義のある規格用語かどうかを判定し、引用を付与する。固有名詞は、定義があいまいな場合は、混乱の原因になる可能性があるが、具体性がある場合には、理解の助けとなる。各概念を、固有名詞で例示するとよい。単語によっては、規格用語と一般用語が混在している場合もある。

組込みシステムに関連する製品、サービスは、多岐に渡っている。そのため、固有名詞で内容を定義していないものがあったり、定義している内容を調べることが困難なものがある。商品名の場合には、定義の変更を明確にせずに、内容を変えている場合がある。

5.6 専門用語かどうかの判定

Stop word だけではなく、一般用語と考えられるものは、用語辞書から除外する。専門用語かどうかの判定は文脈に依存する。専門性は、2 で分類した、対象（TRG）、システムの特性（CHR）、規模（SCL）、開発環境（ENV）に関する分類を付与する。

例えば、STATE は、専門用語として状態を表すこともあるが、一般名詞として current state のように、現在の状態を記述する文章中に表れることもある。動詞として記述するという一般用語として用いている場合もある。

5.7 熟語

用語の木は、単語単位ではなく、熟語単位とすることが必要である。重要な熟語は、目次に掲載されているものである。目次から、熟語を切り出すとよい。よい文献は、索引にも重要な熟語が存在している。共出現率が高い用語は、熟語を形成する可能性が高いため、文中、文節中の共出現頻度を計算して、熟語の可能性を検討することもできる。単語に専門的な意味を付加している場合には、その単語も見だし語(header)とするとよい。

State
 Dispatch pending state
 Dispatch disable state

5.8 用語の包含関係

包含関係には、無関係、同等関係、上下関係、並列関係の 4 つを考える。

無関係　A and B = ϕ
同等関係（同義語）　A and B = B,
　　A not B = ϕ
上下関係　A and B = B,
　　A not B = not ϕ
並列関係　A and B = not ϕ,
　　A not B = not ϕ,
　　B not A = not ϕ

例えば、下位語（Narrower Term）は NT とする。

Dispatch pending state
NT　Dispatching disable state
NT　CPU locked state
NT　Execution with higher precedence

上下関係では、Dispatch pending state は、ディスパッチする側の状態で、dispatching disable state は、ディスパッチされる側の状態である。

抽出してきた内容にあった意味を調べる、類語・関連語を調べ、わかりにくい用語は図にしてまとめるとよい。

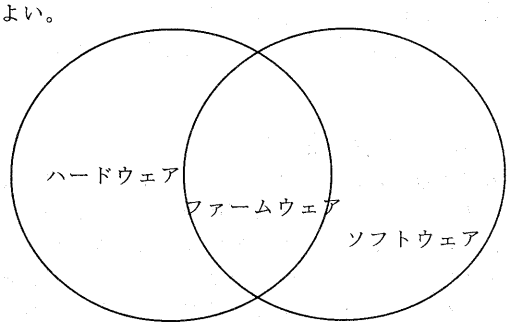

図 1、ハード、ファーム、ソフトの関係

6 編集方法

2 つの概念の共通部分をどのように編集するとよいかを検討した。図 1 の概念を出発点として具体例で説明する。

6.1 関連語として定義

何か関係のある語は、関連語（RT: Related Term）として分類しておけばよいので、用語の意味がよくわからない段階でも、分類可能である。RT が膨大になる可能性があるため、RT が何語以上になったら、再分類するという規則を作っておくとよい。

ハードウェア
 RT ファームウェア、ソフトウェア
ファームウェア
 RT ソフトウェア、ハードウェア
ハードウェア
 RT ソフトウェア、ファームウェア

6.2 両方の下位語として定義

上下関係が明確であることが理解できれば、下位語（NT: Narrower Term）として整理する。

上位語(BT: Broader Term)を記載するかどうかは、全体の用語の量による。

ハードウェア
  NT ファームウェア
ソフトウェア
  NT ファームウェア
ファームウェア
  BT ハードウェア
  BT ソフトウェア

固有名詞は、それを表す一般名詞の下位語に位置付ける。

リアルタイム OS
NT μITRON
MITRON
NT TOPPERS

6.3 AND による表記

各概念に、共通部分との AND を表記して、掲載する。利点としては、上位概念のどういう特性を持った部分集合であるかがわかる点である。

ハードウェア
  AND ソフトウェア ファームウェア
ソフトウェア
  AND ハードウェア ファームウェア

電子回路のハードウェアを実現するソフトウェアには、CAD がある。回路の作り方のソフトウェアはハードウェアのためのソフトウェアであって、ハードウェア AND ソフトウェアという定義の仕方も可能である。

6.4 is-a 関係の定義

２つの概念関係のうち、上下関係を is-a 関係として表記する。逆の関係は a-si で表記する。

ハードウェア
a-si ファームウェア
ファームウェア
  is-a ハードウェア

同義語は、is-a, a-si の併記で表現できる。

6.5 has-a 関係の定義

２つの概念関係のうち、構成要素の関係を has-a 関係で表記する。逆の関係は a-sah で表記する。

ソフトウェア
 a-sah ファームウェア
ファームウェア
 has-a ソフトウェア

ファームウェアは、ソフトウェアを含んではいるが、実体としてはハードウェアの一部であることを表現できる。

6.6 use-a 関係の定義

is-a, has-a 関係だけでは、ソフトウェア、システムの関係を記述するのに十分ではない。例えば、OS の開発で C コンパイラの技術を利用している場合は次のようになる。

OS
  use-a C コンパイラ
C コンパイラ
  a-esu OS

２つの関数が、相互に呼び合っている場合には、相互に use-a, a-esu を付与する。

Use, UF(use for)は、シソーラスにおいて、参照用の略号に使われている。Use-a, a-esu と混用の可能性がある。前述の用語頻度一覧では、is（２位）, has（４５位）, used(９２位)とよく用いられている用語であるためこれを採用した。

6.7 事例と検討

１）サービスコールを例に、is-a, has-a, use-a 関係を含む用語定義を確認した。実際には、そのサービスコールを開発する人と、そのサービスコールを利用する人で、視点が異なる可能性がある。

サービスコール
  a-si ソフトウェア構成要素

a-si カーネル（システムコール）
has-a 型、順番、名前、データ型（引数、戻り値）
use-a 関数呼び出し（C）
use-a マクロ

型
a-si 自動割付け対応整数値パラメータ
a-si 自動割付け非対応整数値パラメータ
a-si プリプロセッサ定数式パラメータ
a-si 一般定数式パラメータ

2) 表2は、出現頻度の高い用語を、3.1から3.4までで定義した分類に基づいて値を代入してみた。この値が正しいかどうかよりも、その値を代入した評価の視点の違いから、複数人による開発の視点の違い、役割による視点の違いを確認することが可能である。

表2　頻出用語と分類との関係

用語	頻度	対象	特性	規模	環境
task	1930	4,5	1,2,3,4,5,6,7,8	2	1,4,5,6
state	1047	1-5	2,4,5,6,7,8	1	1,3,4,5,6
service	1007	5	1-6,8	1,2	1-6
error	688	1-5	1,2,4,7,8	2	1-6
call	628	4,5	1-8	2	1,2,4,5,6

3)　デバッグのための機能

デバッグのための機能と、マルチタスクを実現するための機能を、どこで分類するとよいか。デバッグでは、ICE を使う場合と、シリアルで監視する場合とでは対応できる場合が異なる。

OS とデバイスドライバの関係は、両方ともソフトウェアの下位語として定義可能である。パッケージとして販売されている OS には、デバイスドライバからいくつかの応用ソフトウェアまで同梱されていることがある。そのため、購入者にとっての OS とは、OS の箱に入ったものすべてである。

リアルタイム OS でも、応用ソフトウェアの開発者は、OS、デバイスドライバのすべてが同梱されているハードウェアの購入者には、これらを区分せずにソフトウェアが開発可能かもしれない。

7　まとめ

μITRON に出てくる用語と、TOPPERS/JSP カーネルに出てくる用語、関数、変数を抽出し用語辞典を作成することにより、組み込みソフトウェア開発における用語を分析・分類した。

文書によっては、記号の利用の仕方、変数名の利用の仕方で、切り出し易い文書と、切り出しにくい文書が存在している。いずれにしても、専門家が、その用語を評価し、どのような分類になるかを定義するか、関係者が用語を分類することにより、相互の理解の相違を確認するための手段として用いることができることが分かった。

今後の課題

具体的な作業をしながらの理解であるため、現在やっている作業に偏った検討になっている可能性がある。

ソフトウェアを開発しながらシソーラスを作成する教育を行う場合には、OS とデバイスドライバ、アプリケーションソフトウェアの境界とその設計方法、実現方法について理解していることが重要である。

ファームウェア関連、割込み（例外）関連の用語が、どのような分類にするとよいか、１つのシステムに、ファームウェアが２つ以上存在する場合をはじめ、複雑な場合と単純な場合の違いを検討したい。

現在、TOPPERS/JSP カーネルのソースコードを直接利用しながら、μITRON の機能の理解を進めている。具体的には、無線通信機能を持った端末の開発を想定し、実装方法を検討している。通信関連、試験関連の仕様についても、順次対象とし、μITRON、TOPPERS 全体の概念構造と、具体的な開発対象の生産性、品質の向上を図りたい。

「参考文献」

[1] データベース検索技術者試験・傾向と対策，小川清・安藤彰敏編著,紀伊国屋書店

[2]言語教育モデルによるプログラミング教育ツールの設計、青木賢太郎、小川清、廣瀬哲也、言語処理学会年次大会, 1997

[3]　データベース検索技術シソーラスの作成、小川清、山崎貴和、ドクメンテーションシンポジウム、1995

[4] 組み込みソフトウェア技術者教育の課題と方向性、小川清、渡部謹二、斉藤直希、電気関係学会東海支部,2002

[5] μITRON4.0 仕様、ITRON プロジェクト, 1999

[6] TOPPERS, http://www.ertl.ics.tut.ac.jp/TOPPERS/

[7]分かりにくい外来語を分かりやすくするための言葉遣いの工夫についての提案（中間報告）http://www.kokken.go.jp/public/gairaigo/gai_tyukan001.pdf