

OSインタフェース標準化の動向

松田 晃一

NTT 電気通信研究所

プログラムを作り捨てにせず、自由に流通し再利用出来るようにすることは、強い社会的要請である。このためには、プログラミング言語の標準化を進めることが先ず必要であるが、それだけでは不十分でありOSインタフェースの標準化ができて初めて一層高い流通性を確保することができる。

本稿では、このような立場から、現在内外で進められているOSインタフェース標準化の動向を概括する。IEEEで作業中のUNIX標準およびマイクロプロセッサ用OSのMOSI、DoDにおけるAda開発環境標準化のためのCAIS、さらに日本のTRONを取り上げ、各々の状況を紹介する。

RECENT ACTIVITIES ON STANDARDIZATION FOR OPERATING SYSTEMS

K o i c h i M A T U D A

NTT Electrical Communication Laboratory

This paper describes recent activities on standardization for operating systems.

Major items included in the paper are:

- (1) MOSI (Microprocessor operating system interface) , developed by IEEE Task P-855,
- (2) POSI; standard for UNIX, by IEEE Task P1003 and /usr/group,
- (3) CAIS for ADA programming environment , by DoD KIT and (4) TRON project.

1. はじめに

プログラムを作り捨てにせず、自由に流通し再利用可能とすることは、強い社会的要請である。しかし、現実にはこれが中々困難であって、同機種の中でのソフト流通でさえもそれ程容易ではなく、まして異なる機種間では相当な労力を必要とする状況にある。

ソフトウェアの流通性を阻害する技術的要因は、①言語仕様 ②OS仕様 ③ハード仕様 が機種毎に異なることにある。言語仕様が異なれば、その言語で書かれたプログラムは、流通性は無いのは当然であるが、ファイル/レコードの形式、オーバーレイ制御方式の違い等のOS仕様差やデータ形式や演算精度などプロセッサアーキテクチャの差が流通性を阻害するケースも現実には発生する。

この様なOS機能やハード仕様の差は、高級言語によって仮想化するのが基本であり言語の標準化がまず重要であることは言を待たない。しかし、それでも言語仕様上に直接現れるOSの機能や、言語仕様に規定されていないOSの機能を直接呼び出す場合については、OS仕様差が流通の阻害要因となる。このため、言語の標準化と平行して、少なくともプログラム流通に必要な範囲においてOSインタフェース標準化も進めていくことが重要である。

本稿では、この様な立場から、現在内外で進められているOSインタフェース標準化の動向を概括する。

2. 内外の標準化動向

(1) MOSI (Microprocessor OS Interface)

マイクロプロセッサ上に作られる各種プログラムを機種間で流通させるために必要なOSインタフェースを定めることを目的に、1980年9月にIEEE P-855WGが発足し検討を開始した。メンバはモトローラ、TI、インテル等の半導体メーカーが中心である。

1984年6月にはドラフト6.2を作成、これを改版(6.2A)した後、Technical Committee on Microprocessors and Microcomputers によって投票が行われ、その答申に基づいて1985年3月IEEEのStandard board がこれをTrial-use Standard Specification として承認した。

さらに、この内容はIECの本部に送付され1985年5月に開催されたIEC SC47B カナダモントリオール会議で、MOSIの標準化を新規検討項目として投票にかけることが承認された。(ただし、現在のところ作業は遅れている模様)

これに対応して、IEC SC47B 国内対策委員会では下部にWG-Aを新設し(1985年10月第1回会合)、国内意見の取りまとめがおこなわれている。

(2) UNIX

UNIXの標準化には2つの動きがある。

その1つは、UNIXのユーザ団体である/usr/groupが着手し、IEEEに引き継がれた標準化活動であり、もう一方はATT自身が行っている標準化である。

/usr/groupは、1981年7月に標準化委員会を設立し、1981年12月にボストンで第一回の会合を開催して以来1984年1月に委員会原案を採択、1984年7月の投票を経て、1984年11月に“1984 /usr/group Standard”として標準案を刊行した。

その後、IEEEのP1003作業グループがこの仕事を引き継ぎ、ANSIのX3J11 C Language CommitteeにおけるC言語規格との調整を図って1985年11月に

ドラフトP1003/D6が作成され、これをTrial-Use-Documentとして採用するか否かの投票が行われることとなった。

一方、ATTは独自にUNIXの標準化に力をいれはじめ、SYSTEM V Interface Definition (1985. 1 1版、1985. 10 2版)を刊行すると共に、検定プログラム(SYSTEM V Verification System)を販売するなどを行っている。このなかで、/usr/groupの標準仕様やBSD版など普及度の高いUNIXの持つ機能を採用するなど、広く受け入れられる標準案の制定を狙っている様である。

国内においては、これらのUNIX仕様の標準化と直接対応した動きは見られないが、UNIXへの日本語導入に際しての標準案作りがおこなわれた。即ち、「UNIXシステムが持つべき日本語機能」に関する調査検討を目的とした日本語UNIXシステム諮問委員会が1984年8月に発足し、1985年3月には「UNIXシステム日本語機能提案書」がまとめられ、文字集合、文字コード、システムコール、言語プロセッサ、コマンド等の日本語に関する仕様案が作成された。

(3) CAIS (Common APSE Interface Set)

米国の国防総省(DOD)が進めているプロジェクトであり、Adaプログラミング用の各種ツール類の機種間での可搬性をもたせることをねらいに、これらプログラムとOSとの標準インタフェースを定めている。

CAISの制定作業は、DODの下機関であるKIT(KAPSE Interface Team)が中心になって次のスケジュールで進められている。

1983. 9. 30 Draft Version 1.1完成

1985. 1. Military Standard 1完成

1987. 1. Military Standard 2完成予定

本内容が、標準として提案される予定はいまのところ無い様であるが、DODの標準として、納入品はすべてCAIS上に構築させることをDODは意図しているようである。DODがAdaプログラム発注の大きな割合を占めることを考慮するとその影響は大きく、CAISが実質的な標準となる可能性を持っており、OS標準化にあたっては、その動きに充分注意を払う必要がある。

なお、国内においては1985年度より、電子協ソフトウェア委員会配下の言語標準化専門委員会においてCAISに関する調査が開始されている。

(4) TRON (The Real time Operating system Nucleus)

TRONプロジェクトは、1984年6月から東大を中心に開始された日本のプロジェクトである。そのターゲットは32ビットのVLSI向きマイクロプロセッサ、OS、マンマシンインタフェースなど多岐にわたるアーキテクチャを体系的に構築することにある。このアーキテクチャ体系の1つの特徴として計算機の構造に対する階層構造をきちんと定めて、その階層インタフェースの定義をしっかりとさせることにより、与えられたアーキテクチャ体系を複数の人が自由にインプリメントしても互換性が保てるということを重視して行っている点がある。この点から、OSインタフェースの標準化に関連した動きとしても、注目しておく必要がある。

TRONのうち、制御用リアルタイムOS(ITRON: Industrial TRON)の仕様はすでに定められ、それに基づいた製品も出荷されている。さらに、マンマシンインタフェースを考慮したOS(BTRON: Business TRON)の仕様も1986年の第4四半期に発表される予定である。

(5) 関連の動向—OS I 関連

コンピュータネットワークの中で、情報処理機構（ジョブ実行システム）、情報蓄積機構（データベースなど）、入出力機構（端末）などを共通の資源として自由に利用することをねらいに、OS I の議論の重点が高位レイヤに移ってきている。これに伴ない、OS インタフェースとの関係が密接になりつつある。

これらの議論は、ISO TC 97/SC 21 を中心に、活発な取り組みが行われているが、特にOS インタフェースに関連が深いと思われるトピックを拾うと、GKS (WG 2)、データベース言語 (WG 3)、OS I 管理 (WG 4)、ファイル転送・アクセス・管理、仮想端末、ジョブ転送・操作、OS コマンド・応答言語 (OSCRL) (以上 WG 5)、CASE (WG 6) などが挙げられる。

これらの各項目とOS インタフェース標準化との関係については別稿にゆずるが、OS コマンド・応答言語 (OSCRL) について、以下に若干付記する。

OSCRLとはOperating System Command and Response Languageの略で、OSを端末などから利用するためのコマンド言語とその応答を標準化しようとするものである。これは、CODASYLにおいて検討が進められてきたCOSCL (Common Operating System Command Language:1982) をベースに、ワーキングドラフトが作られ、1986年9月にDP化することを目標に作業が進められている。

一般にOSの提供するサービスを受ける場合、システムコールの様にプログラムを通じてOSを呼び出す方法と、人間が直接コマンドを用いて呼び出す方法があるが、OSCRLは現在のところ後者の方法の標準化として位置づけられている。しかし、OSの動向としては両者を区別するのではなく、いずれの方法であっても同一のインタフェースでOSのサービスを受けられるようにする方向にあり、この点からOSCRLの標準化は、OSインタフェースの標準化と密接な関係をもってくると思われる。

3. MOS I

3.1 考え方

OSのシステムコールの全てを標準化するのではなく、①広く理解されており、②多数のOSで提供され、③流通プログラムにとって必要または有用と思われる一般的な機能を対象として、標準化を行っている。従って、未だ充分受け入れられていない機能やシステム資源へのアクセスが不要な機能、システムに固有な機能などは対象外となっている。

また、アプリケーションプログラムとOSの間のパラメータの引渡方法のマシンレベルでの規定も意図していない。プログラムからの呼び出しインタフェースは、規格書の付録としてPascal, C, FORTRAN, COBOL, Ada, PL/Mとの結合方法をリコメンドするに止められている。

インプリメントの方法は、当然規定されていないが、この仕様に従って新規に作成するまたは既存のOSを修正する方法の他に、既存OSの上にインタフェース変換のための層を追加する方法でも実現できることを想定している。

3.2 規定の概要

(1) ケーバビリティおよびモジュール構成

OSの機能を7つのカテゴリに分類し、それぞれをケーバビリティと呼ぶ。あるケーバビリティに属するシステムコールを、互いに関連あるものを集めてグルーピングしたものをケーバビリティモジュールと呼ぶ。

インプリメンテーションにおいては、このケーバビリティモジュールを単位として必要なものを選択することが許される。

MOS Iでは、幾つかの応用分野を想定し、各分野毎の標準的なモジュール選択を

Recommended configuration として提示している。具体的には、①リアルタイムサポート②簡単なソフト開発サポート③複雑なソフト開発サポート④フルサポートが挙げられている。表1の左側にはモジュールの一覧を、右側にはRecommended configuration を示している。

(2) システムコール

システムコール毎に、期待される動作と入出力パラメタの種類、タイプおよび意味、生じ得るエラーの種類が規定されている。表1の各機能に対応して80個弱のシステムコールが規定されている。

3.3 実施例

米国西バージニア大学において、CP/M-80の上にインタフェース部分を作成した例がある。その他に、現在インプリメントの途上にあるものが数例ある模様であるが、いずれも実験レベルで、実用レベルのものはまだない。

4. ITRON

4.1 考え方

ITRONは、複数のマイクロプロセッサに共通な標準仕様を規定することと、リアルタイム向けにプロセッサの特性を活かして効率向上を図るインプリメントとの調和を図るため、いわゆるバーチャルマシ的なアプローチは採っていない。

すなわち、1ワードのビット数や割り込み処理、例外処理といったマシン従属の部分と、システムコールの機能といったマシン独立の部分を明確に分離し、できるだけプロセッサ間での相違が吸収出来るように作られている。そして、現在広く使われているマイクロプロセッサ(M68000系およびインテルiAPX86系)については、マシン依存部分の標準的なインプリメント方法も示されている。(ITRON/68K, ITRON/86)

また、ITRONの仕様としては、各システムコールの名前や機能、パラメタの種類等を規定するに止め、パラメタの渡し方やビット数、システムコールの起動法などはインプリメント毎に決めるようになっている。

ITRONでは標準の高級言語としてC言語を採用しており、C言語からITRONのシステムコールを呼び出す方法(言語結合)について、標準的なものが規定されている。このような規定法は、先に述べたMOSIと同様の考え方である。

4.2 規定の概要

ITRONは大きく2つの部分、ITRONチップ核と外核に分かれている。

チップ核は、リアルタイムOSの中核となるもので、タスク管理、同期通信機能、割り込み処理、例外処理、メモリ管理、時間管理に関するシステムコール群からなっている。

また、外核には拡張システムコール、入出力、メモリマネージメントユニット付きシステム用のシステムコール、ファイルシステムに関するシステムコール群が含まれている。

4.3 実現例

ITRONの仕様に基づく製品は、日本電気、日立製作所、富士通、三菱電機がNEC Vシリーズ、MC68000, I80286, NS32000用にインプリメントしており、既に商品化されているものもある。

5. UNIX

UNIXが様々な機種に移植されるにともなって、仕様にすこしずつズレが出来はじめ、同じUNIXといってもプログラムの流通性が保たれなくなる危険性がでてきた。これ

を避けるため、ソース・レベルでの互換性を確保し、再コンパイルで機種間を移行できる事をねらいとした標準化である。

すでに製品があって、それらを整理するかたちで後から標準仕様が決まっていくという点では、前に述べたMOS IやTRONとは異なった性格をもっている。

/usr/group の標準は、A T Tのドキュメントをもとに、・機種依存、ハード依存な仕様を除去、・表現の厳密化、一般化（システム定数など）、・機能の追加（ファイル/レコード・ロック機能）を図って案が作られている。 規定の範囲としては、

- ・ System Call : OSの機能を直接呼ぶ為のPrimitive の規定
- ・ Subroutine and Libraries : OS の機能の補助的役割を果たすものとして組み込みライブラリの形で提供される機能の規定

- ・ Miscellinous Data Areas and File Formats : OSの運用管理上必要な各種データおよびファイルの規定。データ交換を狙いとしたMedia Formatの規定

なお、次の様な項目は、当面の対象から除かれている。

- ・ User Interface(Shell and associated commands) ・ Network Protocols
- ・ Graphic Interface ・ Data Base Management System Interface
- ・ Record I/O considerations ・ Object or Binary Code Portability

6. C A I S

6. 1 考え方

A d a総合プログラミング環境は、S t o n e m a nと呼ばれる要求仕様書において、その機能条件が規定されており、それによれば図1に示す3つの階層からなっている。

C A I Sは、この中のKAPSEのインタフェース標準案を指している。

(1) KAPSE (Kernel Ada Programming Support Environment)

プログラミング環境が、機種に依存しない様に、各種のツールと計算機とのインタフェースを標準化し、以下のMAPSEやAPSEの可搬性を保証するためのものである。

(2) MAPSE (Minimal Ada Programming Support Environment)

A d aプログラムの開発に必要な最小限のツールセットを指す。たとえば、テキストエディタ、コンパイラ、インタプリタ、リンカ、ローダなどが含まれる。

(3) APSE (Ada Programming Support Environment)

個々のプロジェクトの目的に応じたツールを追加することによって、MAPSEを拡張したもので、たとえばA d aプログラム専用の特別なシンタックスエディタや文書化システム、プロジェクト管理システム等が含まれる。

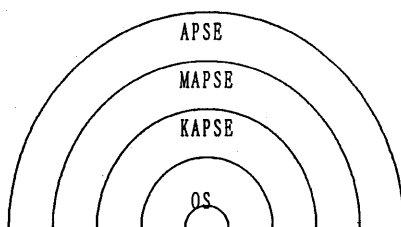


図1 KAPSE/MAPSE/APSE

表 2 C A I Sの規定概要

分類	内 容
ノード管理	Node Management, Attributes, Access-Control, Structural-Node
プロセス管理	Process-Control
入出力	Direct I/O, Sequential I/O, Text I/O, I/O Control, Page-terminal Scroll-terminal, Form-terminal, Magnetic-tape, File-Import-Export
ユーティリティ	List-Utilities

CAISは、OSインタフェースそのものの規定を意図するものではなく、システム毎のOSを活かし、その上に乗せる一種のインタフェースライブラリとして実現されることを意図している。実際に、規定はAdaのパッケージの形式で与えられている。

なお、CAISでは、OSに相当する機能の他に、プログラム開発のライフサイクル中に発生する情報を貯蔵し、一元的に管理する一種のデータベース機能も持っている。

6.2 規定の概要

最新のCAIS仕様 (Military Standard 1,1985.1) では、表2に示す項目について規定されている。

6.3 実現例

現在のCAIS仕様をそのまま実現した例はまだない。

これに相当するものとしては、ALS (Ada Language System)におけるKAPSEがある。ALSは、米国SofTech社が、米陸軍用に開発しているAda開発環境であり、VAX/VMSをホストOSとしてその上に仮想的なOSインタフェースをインプリメントしている。同様な例として米国Intermetrics社が、米空軍用に開発しているAda開発環境AIE (Ada Integrated Environment) のKAPSEがあったが、現在はACS (Ada Compilation System)と名前を替え、コンパイラ開発に縮退した様である。

7. むすび

ここまで紹介してきた各種の標準化の動きについて、若干の意見を述べる。

(1) MOSI

今後規定する予定となっている機能(データ管理、デバイスI/O、等)を追加するとともに、通信機能や共用メモリに関する機能などの追加が必要であろう。

また、マイクロプロセッサの利用という観点からは、割り込みの機能が規定されていない現状では効用が半減してしまう可能性があること(プロセッサ依存性が高いので標準化とのトレードオフになるが)、規定のなかに抽象度が低すぎる機能があり(たとえばタイマ管理)既存OSの上にインタフェース変換のプログラムを作るような方法では実現できそうもないこと、さらに日本語の扱いなども気になるところではある。

全体として良く整理されているが、今後実用レベルでの実施例を積み重ねて改善を図っていく必要がある。

(2) ITRON

製品としてのインプリメントの裏付けもありしっかりした仕様になっている。しかし、対象が組み込み型のリアルタイムOSに絞られているために規定されている機能が限定されている。

ファイル管理や通信機能、アプリケーションプログラムの制御機能、マンマシンインタフェース制御などは、BTRONと呼ばれるOSで仕様が規定されるようで、その公開がまたれるところである。さらに、BTRONでは日本語に関する考慮がなされておりその点でも期待される。

(3) CAIS

機能面では、端末への入出力機能が非常に豊富であること、ノード管理の規定があること等に特徴がある。

しかし、標準化の対象がAda記述プログラムだけで、しかもソフト開発ツールの可搬性を確保するために限定されている。従って、この規定だけではOSインタフェースの標準にはならない。

全体として、相当機能が豊富であり、母体として選択するOSの機能にもよるがCAISフルバックを実現するには、相当の量のソフトが必要となることが予想される。

本稿で報告したいいくつかの標準化の動きは、対象とする計算機の規模や、対象分野、目的、前提とする環境条件等がそれぞれに異なっており、同列に論ずることは出来ない。逆に、OSの標準化を進めるには、まずこれらの条件をはっきりさせ、ターゲットをある程度絞って議論する必要性を示している。

OSの標準化は、技術的な困難さと同時に、企業の戦略や利用者側の思惑などが複雑に絡み合った非常に難しい問題である。しかし、一方でプログラムを作り捨てにせず、自由に流通し再利用可能とすることは、強い社会的要請であることは間違い無く、この実現にむけて、すこしずつでもこの様な標準化の努力を重ねていく必要がある。

謝辞 本稿の内容は、情報処理学会規格委員会のOSインタフェース専門委員会での議論に依るところが大きい。高橋茂委員長、棟上昭男幹事をはじめ委員各位に謝意を表す。

参考文献

- 1) IEEE Trial-use Standard Specification for Microprocessor Operating Systems Interfaces, (1985)
- 2) J.D.Mooney: The MOSI Standard for operating system interfaces implementation and use, Technical Report 85-1, (1985)
- 3) 石畑清他: 最近のADAの動向, 情報処理, No.3, Vol127, pp229 (1986)
- 4) Military Standard Common APSE Interface Set(CAIS), proposed MIL-STD-CAIS, (1985)
- 5) 坂村健: リアルタイム・オペレーティングシステムITRON, 日本ロボット学会誌, No5, Vol13, (1985)
- 6) 情報処理学会 計算機アーキテクチャ研究会資料 61 —1 ~6(1986)
- 7) IEEE, Portable Operating System Environment P1003/D6, (1985)
- 8) /usr/group: PROPOSED STANDARD 1984/usr/group Standard, (1984)
- 9) ATT:UNIX SYSTEM V Interface Definition, (1985)

表1 MOSIの
ケバビリティ
およびモジュール

CAPABILITY	CAPABILITY MODULE					APPLICATION CLASS		
	1	2	3	4	5	Real-time support	Simple development support	Complex development support
MEMORY MANAGEMENT	Simple	Multiple				1	1	1
TIME MANAGEMENT & SCHEDULING	Simple Delay	Timer Management	Time of Day			1 & 2	---	1 & 3
DATA TRANSFER	Direct I/O	Synchronous File I/O	Asynchronous File I/O			2 & 3	2	2 & 3
PROCESS MANAGEMENT	Single Process	Multiple Process				1 & 2	1	1 & 2
PROCESS SYNCHRONIZATION & COMMUNICATION	Semaphores	Multiple Semaphores	Messages	Messages with Responses		1 & 3	---	1, 2 & 3
ENVIRONMENT	System Characteristics	Command String Acquisition	Direct Argument Acquisition	Overlay Management	Standard I/O	1	1, 3 & 4	1, 3 & 4
EXCEPTION HANDLING	Default Handler	Single Exception Handler				1 & 2	1 & 2	1 & 2