

欧米における実時間システムの研究動向

Research Trends on Real-Time Systems in Europe and USA

阿江 忠 (広島大学)

Tadashi Ae, Hiroshima University

概要

欧米における実時間システムの研究動向について、この10年をグローバルに紹介し、今後を考えてみたい。

Abstract

Getting back to ten years of researches on Real-Time Systems in Europe and USA, I will discuss the future of this research field.

1. まえがき

実時間システムについて、筆者の理解している範囲であるが、欧米におけるこの10年の研究動向について紹介し、今後を考えてみたい。できるだけ最近の話題も含ませたいが、個々のテーマはその方面の専門家にお任せし、ややグローバルな話をしたい。

話の都合上、個人的な事情が随所に入ることをお許し願いたい。

大雑把な表現をすると、この10年のはじめの数年(1985年頃まで)はハードのウエイトが結構あったが、それ以降はソフトを中心とする研究に移っている。

この点でも、「ハードウェアの復権」をもちろむ筆者は端の方からものを眺めていることになり、やや肩身が狭いが、概略現状への推移を述べてみたい。

2. アメリカにおける研究

--- Symposium, Workshop などからみて ---

もっとも歴史のあるのはIEEEの Real-Time Systems Symposium である。このシンポジウムが発足したのは1981年で、前身のワークショップ時代は Data Acquisition and Control と称していた。名前からわかるように、コンピュータという色彩より、コント

ローラとかセンサの話が中心でハードやシステムの話はあってもソフトの話は少なかった。この傾向は「実時間シンポジウム」になっても、しばらく続いた。筆者が出席した初めはこの時期(1982、1983年)であった。個人的には、1978年、1982年それぞれにAKOVST(5CPU)、UNIP(32CPU)というマルチマイクロプロセッサを完成させていたが、いずれも8ビットCPUをベースにしていたのでISCA(International Symposium on Computer Architecture)では肩身が狭いと考えた。また、当時は産業用ロボットのブームでプリンタポートにつなげる(デモ兼教育用)のものが市販されたりした。このような状況から、コントローラ指向のマルチマイクロプロセッサの論文は、ハードウェア[1]、ソフトウェア[2]はともに受理された。ただし、今思えば1983年頃から変化の兆しがあった。この年のWashington D.C.のシンポジウムでJ. Stankovicが(確か特別セッションかあるいはバンケットトークのどちらかで)米国政府(とくにNASA)の実時間システムへの力の入れ方の話をしており、大学などへの研究援助資金についての質問が出たりした。1984-1986年は諸般の事情で出席できなかった(毎年12月初めに催されるのは都合がいいようであるが、まずい面もある)。久々に参加出来たのは1987年であったが、様子が一变していた。この間に Real-Time Operating Systems のWorkshopが出来たり、OSやソフトを中心とするシンポジウムになっていた(したがって、メンバーもがらっと入れ替わってしまった)。また、このことは論文の質にもかかわっており、俗にいう一流シンポジウムの条件である採択率1/3に近づいたのもこの時期である。その影響あってか

Data Acquisition and Control の時代にあった計測、制御への応用みたいな話がなくなり、Advanced Computer System に関する話でないと採択されにくくなっている（このような傾向にはヨーロッパ側からは批判もあるが、日本からの参加を増やそうとするとき考慮すべきことであろう）。この Real-Time Systems Symposium のおよその傾向の変化を参考 1 に示す。

「実時間システム」という用語自体は古いし、あらゆる分野に関連している。つまり、その分野での仕事の特徴として実時間性があれば、(多少の誇張も含めて)「リアルタイムXXX」という表現を用いて特徴をアピールすることに役立ってきた(たとえば、リアルタイムUNIX、真に実時間性があるか否かは別にして)。したがって、Real-Time Systems Symposium 以外の場所でも実時間システムに関する論文は結構発表された(し、今後も発表されるであろう)。ただし、それは散見される傾向が強いから、筆者の知るかぎりという条件つきで少し述べたい。

IEEE Computer Society のワークショップとして Languages for Automation (略称 LFA) が発足したのが 1983 年頃である。もっとも、Factory Automation のみならず当時の流行であった Office Automation も含む広い範囲でのソフトウェアを対象としていた(LFA では Japan Session が設けられたこともある)。この LFA の中にも実時間システム関係の論文が見られる。この LFA はその後 LFA と改称されている。蛇足であるが、LFA の中心メンバーである T. Ichikawa と S-K. Chang が 1984 年に Visual Languages のワークショップを発足させた(第一回の 1984 年と第八回の 1991 年の 2 回は日本で開催された)。

その他、もっと大きな国際会議の一セッションとしても実時間システムはよくとりあつかわれるが、これについては割愛する。

3. ヨーロッパにおける研究

実時間システムに関するシンポジウムとしては 1989 年より Euromicro Real-Time Workshop が発足している。ところが、W.A.Halang によると、これよりも IFAC Workshop on Real-Time Programming (主にヨーロッパで開催されるが Worldwide のもの)のほうが良いとのことである。その他は各国内のもの(たとえば、ドイツでは "Prozessrechnungssysteme" とか "PEARL

Workshop") になるらしい。W.A.Halang はこの方面の牽引者的な役割をしている[3]。実時間システム (Real-Time System) の定義をきちんとふまえないと誤解を起こさせるとして、GIS (German Industry Standard) を引用して警告している(参考 2)。同時に International Communication の悪さを嘆きその改善に努めている(4。参照)。W.A.Halang のあげる実時間システム研究の方向は次の通りである[3]。

- 1) Conceptual Foundation of Real-Time Computing,
- 2) Predictability and Techniques for Scheduling Analysis,
- 3) Requirement Engineering and Design Tools,
- 4) Reliability and Safety Engineering with Special Emphasis on the Quality Assurance of Real-Time Software,
- 5) High-Level Languages and Their Concepts of Parallelism, Synchronization, and Time Control,
- 6) Real-Time Operating System,
- 7) Scheduling Algorithms,
- 8) Distributed, Fault-Tolerant, Language and/or Operating System Oriented Innovative Computer Architectures,
- 9) Hardware and Software of Process Interfacing,
- 10) Communication Systems,
- 11) Distributed Databases with Guaranteed Access Times,
- 12) Artificial Intelligence with Special Emphasis on Real-Time Expert and Planning Systems,
- 13) Practical Utilization in Process Automation and Real-Time Control,
- 14) Standardizations.

最後の標準化は別にしても、13項目は多すぎるという印象を受ける。それに、Parallel, Distributed, Fault-Tolerant, Communication, AI など既にそれなりのシンポジウムや会議が存在している分野からどのく

らい「実時間性を重視したもの」が独立した形で抽出できるかが課題となろう。もっとも、このくらいの大風呂敷から始めてちょうどいいところへ落ちてくかも知れない。

4. 実時間システムに関するジャーナル

--- Journal of Real-Time Systems ---

1989年に Kluwer Academic Pub. より発行されたジャーナルで Real-Time Systems をタイトルに掲げた唯一のジャーナルである。W.A.Halang の嘆く "The Sad Situation of International Communication" の解消をねらっていることは言うまでもない。Editors-in-Chief は

J. A. Stankovic (America)

W. A. Halang (Europe)

M. Tokoro (Far East)

の三名でそれぞれよく名前の知られているかたがたである。それぞれの地域に Editor が各10人づつくらいいる（ただし、日本は半分くらい）。筆者はこのジャーナルには創刊時から少なからず関わりがあり、Vol. 1, No. 4, April 1990において日本特集号の Guest Editor を務めたことがあるが（参考3）、当時は論文集めに結構苦労した。むろん、コンピュータのハードとソフトには IEEE の Transactions として Computers と Software Engineering があるから、そちらを第一に考えていただくとしても、実時間性にウェイトのある論文は是非このジャーナルへ投稿していただきたい思っている。

5. 実時間処理のための言語（その1）

--- Cを中心として ---

実時間処理の記述を目的とする言語も古くから存在するが、大雑把に言えば、アセンブラ、Fortran、Cという流れになろう。アセンブラは例外として、Fortran およびそれに類する言語は現場の技術者に好まれる。志は高く掲げても、現実とは別ということも多い。余談になるが、前述の W.A.Halang 教授のところの J. Scheepstra という若い研究者が昨年11月より3カ月当大学に滞在したが、シーケンサ記述のための言語を使っていた。また、3で触れたPEARLという実時間処理を目的としたドイツ産の言語も Fortran 的である。

一方、実時間処理の世界でも趨勢はCとなっている。実は、筆者もRCC (Real-Time C for Controller) [2] とか *C [4] などと称した実時間処理向きのCを作ったりしたこ

とがある。いずれも、並行処理+割込み処理をCに加えている。RCCはマルチプロセッサ上でのセルフ開発用であるが、*Cはシングルプロセッサ上でのクロス開発用である。前者の並行処理機能をもつCはその後各所で誕生した「並行C」と似てはいるが、割込み処理に重点をおくため、思想的には異なるところがある。しかし、最近のように Concurrent C がC（およびC++）のスーパーセットとして定義された上に実時間性をもたせた Real-Time Concurrent C [5] ともなると、筆者の出番はない。とくに、データベースシステムなど分散OSがらみの分野での実時間処理と Embedded Computer におけるコントローラ用の実時間処理とは分離する傾向にあり、相対的には前者のウェイトがますます大きくなろう。

6. 実時間処理のための言語（その2）

--- AI 応用指向 ---

これもかなり筆者の趣味ではあるが、AI 応用を目的とするシステムに1985年頃から興味をもった。とはいえ、ICOTのようなビッグプロジェクトなどは夢にも思いつかない。作ろうとしたのは実時間プロダクションシステムである。並列プロセッサによる高速化の検証には既存のマルチプロセッサも活用できた[6]。しかし、よく言われていたように、ルールベース記述のプログラムの並列性はそれほど大きくなく、コスト高になるという欠点は避けられない。再び、シングルプロセッサの上でのアルゴリズムの改良に戻った[7]。この間、世の中での試みを調べたことがある[8]。概略は次の通りである。

一般に、手続き型言語にはアセンブラで書いたルーチンを適宜挿入できる、いわゆるインライン機能を備えていて、ハードウェアとの間を受け持っている。しかし、アセンブラのレベルをCと考えてよい時代になってからは、むしろ、Cより上の言語がC（あるいは同等の言語）がインライン的に書ければその役割をもつ。AI 応用を目的とする言語 Lisp, Prolog, Production System 記述言語（PS と略す）でも便宜上このような機能をもつことが多い。しかし、これは外部入力とのインタフェース記述用であり、速度改善にはならない（ほとんどメインルーチンで速度が決まるから）。つまり、発想は逆でなければならない。メインルーチンをCで書き、AI的に書きたいところだけそうすれば速度もそれほど落ちない。このような機能を「アウトライン機能」と呼ぶと、一般にアウトライン機能をもつ言語を

X-Y (X:メイン、Y:サブ)と表現したとき、C-Y (YはAI記述向き言語)は実時間向きである。筆者はその例としてC-PS的なものをこころみだが、時同じくしてC-OPSなるものも発表された。その他には Modula-Prolog, Ada-Prolog がある(参考4)。当時、筆者ら以外の試作はいずれもヨーロッパにおける仕事である。筆者がC-PS (RBC: Rule-Base C という名称の言語)をつくって発表したときも、「どうしてPSコンパイラの良いものをつくらないのか?」と質問された。筆者は「大企業ならそうでしょうが、中小企業はこのようなやりかたをします」と答えた。少なくとも Embedded Computer 用の実時間処理に関係する人はこのような感覚を大事にしている。しかし、実時間処理の応用が広がるにつれ、小数派になっていることも否めない。もうひとつの筆者らの作品であるDPS (Demand-driven Production System)のほうは要求駆動型に動作形式を変えてはいるが、オーソドックスにPSの高速化を試みたものである[7]。

その後はニューラルネットのハードウェア化にのめり込んでいる[9]。これも、実時間応用にはソフトでのシミュレーションではダメという感覚での自然な発想によるものであるが、所詮は筆者の育ちからくるものかも知れない。

7. 理論的なよりどころは何か?

実時間性をもつシステムは実用指向であるから、とくに理論は必要としないという考え方はある。しかし、バックボーンとなる理論なしではやはり寂しい。筆者らはかつて公理的なアプローチを試みたことがあるが[10]、並行プログラムの亜流という感は免れない。これは、テンポラルロジックの利用についても言え、ペトリネットでも同じような感は否めない。個人的な興味としてはスケジューラがあるが、アルゴリズムとしては純粹なものがつい頭に浮かんでしまい、実用面での実時間性の条件を入れた手法を考えるのがわずらわしくなりがちである。これは、そのような論文の程度がどうこうというのではなくて(それはそれで立派な内容をもつことが多い)、借り物でない「実時間システムにオリジナルな理論」を待望したいという気持ちのあらわれなのである。この意味で、若き牽引者が我が国にも登場することを期待したい。

8. さて、これからは?

確かに、アメリカでは既に10年の歴史をもつ Real-Time Systems Symposium があり、関連ワークショップも盛んである。ヨーロッパもそれなりの形を整えつつある。「さて、我が国は?」と思っていた矢先に RTP'92 が開かれることとなり、まずは一安心というところであろうか。ただ、W.A. Halang のようにアメリカ主導型のものとヨーロッパ型との違いは残っている。ところで、日本型はどういう形で展開していくのであろうか?

どちらにしても、筆者はハードウェア依存性の強い実時間システムという枠組みでお役に立てればと考えている。

最後に、今回の講演に際し、資料をEmailで送付された Prof. W.A. Halang に感謝する。

文献

- [1] T.Ae and R.Aibara: "Experimentation and analysis of multiprocessor systems", Proc.IEEE Real-Time Systems Symposium, L.A., pp.69-80 (Dec.1982).
- [2] T.Ae, S.Tenma, R.Yamasaki and M.Kitagawa: "A distributed real-time processing language on multimicroprocessor system", Proc.IEEE Real-Time Systems Symposium, Washington D.C., pp.20-29 (Dec.1983).
- [3] W.A.Halang: "Real-time systems: Another perspective" (prepared for publication).
- [4] Y.Kagawa, W.C.Cunha, R.Aibara, M.Yamashita and T.Ae: "C system: A development tool for real-time embedded computer", IECEJ Trans., Vol.E-79, No.9, pp.1011-1019 (Sept.1986) and : "A concurrent language and its development tool for embedded computer", Proc.IEEE Workshop on Languages for Automation, Singapore, 166-170 (Aug.1986).
- [5] N.Gehani and K.Ramamritham: "Real-time concurrent C: A language for programming dynamic real-time systems", Journal of Real-Time Systems, Vol.3, No.4, pp.377-405 (Dec.1991).
- [6] 竹内拓二、藤田 聡、相原玲二、山下雅史、阿江 忠: "プロダクションシステムのための並列マッチング方式とマルチプロセスによる一評価", 情報処理学会論文誌, 30巻, 4号, pp.486-494 (April 1989) and S.Fujita, M.Yamashita and T.Ae: "Search level parallel processing of production systems", Proc. PARLE'91, Parallel Architectures and Language Europe, Eindhoven, pp.471-486, Lecture Notes in Computer Science 506, Springer-Verlag (June 1991).

- [7] 藤田 聡、介弘達也、山下雅史、阿江忠：“前向きプロダクションシステムのための要求駆動マッチングアルゴリズム”、情報処理学会計算機アーキテクチャ研究会(Mar. 1989).
- [8] 阿江 忠：“知識工学の実用化をめざして（連載：マイコンソフト開発のための知識工学12）、インターフェース、pp.283-287 (April 1989).
- [9] 阿江 忠：VLSIニューロコンピュータ（21世紀のアーキテクチャをめざして）、共立出版（1991）.
- [10] W.C.Cunha, M.Yamashita and T.Ae：“On a formalization of real-time programs”, IECEJ Trans., Vol.E69, No.8, pp.866-876 (Aug. 1986).

参考1。IEEE Real-Time Systems Symposium における論文の分野別一覧

年	82	83	85	86	87	89	90
アーキテクチャ (ソフト)	3	4		3	3	4 5	3
フォールトトレランス	2	6	8	6	3	3	3
設計、性能、評価	6	4		5	4		3
通信、結合ネット	3			2	3		3
OS、分散システム	5	3	2	3	5	8	3
アロケーション、 スケジューリング	3		3	3	5	3 5	9
言語、ソフトウェア		6	7	2	4	3	4
その他		6		3	3	5	5
合計	22	29	20	30	30	36	33

注1：81,84, 88年は欠、また、分類は形式的であり、必ずしも中身を吟味したものではない。

注2：前身のワークショップ名は Data Acquisition Computing and Contorol が正確。

参考 2。 G I S (DIN 44 300) における実時間システムの定義

The operating mode of a computer system in which the programs for the processing of data arriving from the outside are permanently ready, so that their results will be available within predetermined periods of time; the arrival times of the data can be randomly distributed or be already a priori determined depending on the different applications.

参考 3。 Journal of Real-Time Systems, Vol.1, No.4 (April 1990)

Guest Editor's Introduction, T.Re
Real-Time Control Software for Transaction Processing Based on
Colored Safe Petri Net Model, T.Murata and N.Komoda
An Application of Structural Modeling and Automated Reasoning to
Real-Time Systems Design,
S.Honiden, N.Uchihara, K.Matsumoto, K.Matsumura and M.Arai
Dataflow Computer Extension Towards Real-Time Processing,
M.Takesue
Programmable Real-Time Scheduler Using a Neurocomputer,
T.Re and R.Ribar

参考 4。 アウトライン機能をもつ言語の例
(X-Y という表記は X : メイン言語、 Y : 上位言語)

Modula - Prolog (Brown Bovari Research Center)
Ada - Prolog (University of Milano)
C - OPS (Helsinki Institut of Technology)
C - PS (Hiroshima University)