

TRONプロジェクトの15年

□ 東京大学 坂村 健

TRONプロジェクトの第1ステップ

TRONとは“The Real-time Operating system Nucleus”の頭文字をとったものであるが、リアルタイム処理を基本とし、マイクロプロセッサを中心としたハードウェアから応用までをカバーする新しいコンピュータ体系開発プロジェクトである。

まず、TRON^{★1}の15年を振り返ってみよう。TRONは常にトップダウンにプロジェクトを進めてきた。しかも、哲学レベルからのトップダウンである。そもそもアカデミックとビジネスとコンシューマーの差が小さいコンピュータという分野では、コンセプト——さらにいえば——哲学を打ち立て、向かうべき方向性を示すことが、まずアカデミックのなすべき役割であると考えたからである。論文の発表と新製品の発表にレベルの差がない——さらには、企業の方が先をいっているという企業人もいる。しかし、バネパー・ブッシュのMemex¹⁾やアラン・ケイのPersonal Dynamic Media²⁾を始めとして、そのコンセプトがその後のコンピュータの進歩の方向性に大きな影響を与えるような研究はアカデミックのものであり、それこそが、アカデミックの存在意義といえる。

こう考えれば、TRONは間違いなくこの方法でプロジェクトを進めてきた。「生活の場にマイクロコンピュータがあふれる」というTRONの仮定から導いた、HFDS (Highly Functionally Distributed System)——いわば「どこでもコンピュータ」のコンセプトは広く認知され、後に米国で同様の考え方によるUbiquitous Computingという研究分野が生まれた。日本では商品化されないと理解できないということか、TRONはコンセプトのレベルではあまり評価されなかったようだが、Sun Microsystems社の開発者がインタビューに答えていっているように³⁾、むしろアメリカでは新しい方向性を示す研究として評価され、たとえば最近ではよく似た考え方に基づくJiniが商品化されるまでに至っている。

また、他のデータへのリンクを一般的なデータとして、テキスト等の中に埋め込めるようにするというBTRON

の実身/仮身モデルも、最初理解されなかったが、Apple社のHyperCardの出現ぐらいから日本でも理解されるようになり、今では逆に「ファイル構造自体がWWWと同じ」といわれるぐらいである。

さらに「オープンアーキテクチャ」というTRONの思想は、最近の「オープン～」の流れに少なからず影響を与えているだろう。Unixカルチャーに当初からあった思想ではあるが、それを整理し「標準化部分の仕様を公開し、複数の自由な実装を勧める」という方針として明文化し、レベル付けし検証規定を確立するなどしたやり方は、いわば「開発の仕方自体のアーキテクチャ」のようなものであり、オープンソース、組込み機器メーカー自製、製品を含む70からのITRONの独立した実装を生んだ。

特に、昨今のようにコンピュータが社会のバックボーンとまでいわれる状態では、ウイルスやバグ、2000年問題などが広く知られるようになった結果、単一の実装によりすべてを賄うことの危険性がより強く認識されるようになってきた。生物が性のメカニズムを取り入れ遺伝子をシャッフルし、ある環境下では必ずしも最適化していない個体の存在でも全体として常に維持しているのも、同様の理由である。いくら最適であっても同じ遺伝子でクローニングしたような生物群——たとえば人間の作る耕作地など——は環境が変わったり、何かのウイルスが出現したときにひとたまりもなく全滅してしまう恐れがあるからである。

ソフトウェアはクローニング——つまり単純なコピーで容易に増やせるモノである。多様性を持ち込むには、意図的に異なる実装を作るような努力——設計分散方式 (Multi Design Diversity) が必要とされる⁴⁾。

TRONでは、TRONの仮定を前提としてそのような危険を早くから考え、「標準化部分と実装を切り離し、標準化部分の仕様はフリーで公開し、複数の自由な実装を勧める」という「オープンアーキテクチャ」を提言してきた。この結果としてITRON, CTRON, TRONチップにおいて、多数の異なる実装が生まれ、ある意味で、パソコンOSなどよりはるかに社会を影で支えているともいえる組込みOS、制御OSの分野で「標準化の上の多様性」を実現し、社会の安定性に貢献できたと自負している。

★1 以下、“TRON”はTRONプロジェクトのこととする。

このように「どこでもコンピュータ」環境の情報インフラストラクチャとして望ましい性質という観点から開発におけるモデルを考えたのは、当時 TRON だけであった。そして、そこから標準化するべき部分と多様化し競争すべき部分を切り分けるという「オープンアーキテクチャ」を提唱し、それが要求にあっていたことが、現在の ITRON、CTRON の普及につながっていると考えられる^{★2}。

プロジェクト 15 年の分析

パソコンは「コンピュータの顔をしたコンピュータ」であり、OS をコンシューマーに意識させるものであり、マスコミの関心は常にパソコン OS に集中してきた。これは TRON プロジェクトに関していえば、BTRON である。このためにいまだに「TRON は失敗した」と書くマスコミもいるが、これは BTRON を TRON 全体と混同することによる間違いである。

では、本来的には、ITRON と同様に「オープンアーキテクチャ」を提唱した BTRON 仕様 OS が — 現在も進行中なので言いきることはできないが — 少なくとも現時点まで普及していない理由はなんだろうか。

マーケティング主導

これには 2 つの原因が考えられる。1 つは先に述べたように、パソコン OS がコンピュータの中でも特に、マーケティング主導の競争原理に支配されているということである。そもそもアカデミックとビジネスとコンシューマーの差が小さいコンピュータという分野では、本来技術的であるべき判断が、マーケティングに圧倒されるということが、往々にして起こるが、パソコン OS は特にその傾向が強い。

さらにいえば、BTRON に対して実際に影響したのは市場の評価よりさらに恣意的な — マーケティングというにはあまりに生臭い — 政治的圧力であった。当時 GUI タイプのパソコン OS の覇権は確定されていない状態で、提案された BTRON は実際に、松下電器、IBM などの複数の会社で実装され、「オープン」であり、特定の会社にコミットしないという点を評価されて、文部省の教育用標準 OS として採用される直前までいった。しかし、これに対して米国商務省通商代表部 — USTR が、不正貿易の疑いがあるということで、スーパー 301 条の候補品目リストに TRON をあげたのである。

1989 年当時アメリカに対する輸出を第一義に考える

^{★2} 本特集の他の論文でも触れているので、ここではコンセプトレベルより下の成果については多くを語らないが、ITRON はすでに組み込み分野の OS の世界レベルの標準仕様として確立したといえるし、CTRON は日本の電話網の電子交換機をすべてを支えている OS となっていることを付け加えておきたい。

各社は、まだビジネスにもなっていない TRON にかまってアメリカににらまれるのは得策でないとはばかりに、この圧力に対して一も二もなく TRON から撤退してしまった。

実際には、この後の我々の抗議に対し、USTR はアメリカの会社からの要求があれば一応なんでも候補リストに載せるというだけで、調査した結果としては「オープンアーキテクチャ」の BTRON を不正貿易候補としたことは誤りであったと認めるのであるが、時すでに遅くほとんどのメーカーは撤退してしまっていた。これは、もはや技術とはまったく別の世界であり、このような事態までは予想していなかった。

覇権構造

もう 1 つの BTRON が ITRON と異なる点は、組み込み OS はそれを組み込むシステムの開発という観点でワンサイトで個別評価されるのに対し、パソコン OS では、「際限のない互換性」の面から評価されるということである。パッケージアプリケーションが使えるかとか市販の周辺機器が接続できるか、自分に必要な特定のアプリケーションが使えるかどうかだけでなく、さらには将来使うかもしれないという可能性まで含めた互換性を、ここでは「際限のない互換性」と呼んでいる。

具体的にいえば、アプリケーション開発者はより大きな市場を期待して一番広まっている OS ベースのバージョンから開発するし、一方消費者の判断は「際限のない互換性」の評価基準により、より多くアプリケーションの乗っている OS — つまり覇権をとっている OS を選択する。

しかし、従来の工業製品であれば、たとえ互換性の拘束条件があっても、ビジネス的に十分な収益が見込める商品ならば互換性を持つ「もう 1 つ」の実装があらわれ、競争原理が働くはずである。

残念ながら、パソコン OS の世界ではそのようなことが実現していない。この状況を生んでいる特殊事情は何であろうか。まず、第 1 に考えられるのは、インタフェースが知的所有権で守られていることである。OS のインタフェースである API (Application Program Interface) は MPU の ISP より複雑であり、知的所有権の認められる範囲が広い。次に競争排除の仕組みとして考えられるのは、OS はソフトウェアであり、圧倒的な覇権を確立した状態であれば、他の追従者の開発コストが成り立たない戦略的な低価格で提供できることがあげられる。

また、もう 1 つの理由としてパソコン OS が互換性の必要な多くの部分から成り立っているため、圧倒的な覇権を確立した状態であれば恣意的にインタフェース仕様を変えることで、常に追従者の開発がコストに合わない状態を作り出すことができることもあげられる。

この後者の戦略はコンパチつぶしとしては一般的に考えられるものであるが、従来は互換性に従う他の関連メーカーの造反も考えられ、一方的に行えるものではなかった。IBMがコンパチつぶしを目的にPCの規格を変更した際に、追従されず逆にIBM PC互換アーキテクチャがIBMのものでなく業界の共有財産であるというオープン意識を確立してしまった。

セットメーカーにとっても、アプリケーションメーカーにとっても、自社製品をパソコンOSの変化に応じて変えなければならないことは、本来望ましくない。BTRONのようなオープンアーキテクチャであれば、仕様設計者と実装者が異なるため、恣意的にインタフェースが変更されるということはない。このような考え方、やり方が、社会の情報インフラストラクチャとして望ましい性質だと我々は考えたのである。

しかし、メーカーの経営サイドの判断としては、バージョンアップは、新たな需要を生むということで、逆に歓迎された。ここで重要なのは常に機能を肥大化させ、より高性能のハードウェアを必須とすることである。これが、セットメーカーに買い換え需要創造、最低価格維持というメリットを与えた。もし機能の肥大化を行わないならば、MPUの高性能化により、たとえばMacintosh上でのWinエミュレータがWindowsの競争相手となってしまう。セット全体として量産による低コスト化のメリットを覇権に生かすために、むしろ本来なら高コスト化を招くはずの機能の肥大化を続けるという不思議な構図が生まれたのである。

パソコンOSは自分が持っている複数のアプリケーションを同時に使いたいという横方向の互換性、データを生かしておきバージョンアップ版を使い続けたいという縦方向の互換性、この2つの互換性でできた網がぐちゃぐちゃとユーザが囲い込まれているのである。その状況があるからこそ技術の進歩もバージョンアップも仕切り直しにならないという安定性があり、その結果むしろ特定の「製品」が、バージョンアップすることで———というか、バージョンアップし続けるがゆえに、覇権が維持される

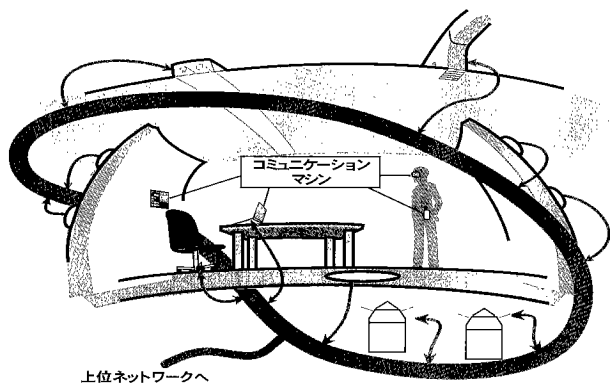


図-1 計算能力を持った環境とコミュニケーションマシン

などという摩訶不思議な構図になっているのである。

そして「オープンアーキテクチャ」は、その「互換性」と特定の「会社」、特定の「製品」を切り放そうというコンセプトであり、この互換性の網を解体するものなのである。

学ぶこととこれから

では、事態はこのまま安定しているかというところとはいえない。TRONでは、パソコンというコンピュータの形態自体は過渡的なものであり、今というパソコンは、「ユーザとユーザ」、「計算能力を持った環境とユーザ」とのまさに仲立ちをする機能に特化した「コミュニケーションマシン」に転化していくと予想している(図-1)。仲立ち機能以外の個々のアプリケーション的機能はむしろ「計算能力を持った環境」の方に分散され、必要な情報処理がされて、コミュニケーションマシンに帰ってくると考える⁵⁾。最近でいうところのネットワークコンピュータは、このコンセプトのはしりということもできる。

むしろBTRONはTRONの仮定から導かれるそのような環境を前提として考えられたシステムである。環境の方が日々変化し続ける中で、ユーザの側に立ってその変化を吸収し、そのユーザの身体・文化的属性や状態を把握し、それに適した形で安定したインタフェースを提供する。それこそがBTRONに求められるコミュニケーションマシンとしての機能である。

しかし、初期にBTRON仕様OSを教育用標準OSとして採用しようという動きがあり、そのために教育用の種々の———特定されない———アプリケーションを乗せるための汎用的プラットフォーム———つまりパソコンとしての性格を要求され、残念なことに本来のコミュニケーションマシンというコンセプトからは異なる方向に注意が向けられた。

その結果、ネットワークや多国語対応、2つのシステムの間で最適の負荷分散を行うためのネットワーク経由でローディングできるスクリプト言語、ハイパーメディアの標準データフォーマットなど、コミュニケーションマシンのための「オープン」環境として重要なアイデアは———提示したもの———当時まったく一般的でなく理解も得られなかったため後回しにされ、むしろその時点でよくあるパソコンOSと比べて、カタログ的に同じ項目にマルが付くというような観点で開発が行われたのである。

ところが、これはTRONのコンセプト自体は正しかったという結果だと考えられるが、近年になってこれらのコミュニケーションマシンのための「オープン」環境が進展し、これがパソコンOS市場の普通の市場への変化を進め出したと我々は捉えている。

具体的にいうならば、たとえばPDAである。PIMツ

ールと電子メール機能とWWWブラウザ機能だけで、多くの仕事がこなせるようになってきている。これはモバイルネットワークといったネットワークレベルでの環境が整ってきたことの影響が大きい。

そして、このPDAでは、ユーザは購入後、特にアプリケーションを追加購入しようとも思わない。その結果、パソコンとのデータ互換は考えても、そのPDAのOSが何かということに対する関心はパソコンに比べはるかに小さくなる。むしろ、ユーザの関心は大きさ、重量、電池駆動時間、反応速度、価格といった単体としての性能に向けられる。これらの項目については、むしろ肥大化により覇権構造を維持するパソコンOSは適していないことが確実である。

ネットワークの普及とモバイル技術により、ウェアラブルコンピュータについて語られる現在の状況は、まさに「コミュニケーションマシン」に向かっているといえるだろう。そしてこの方向性が明確になるにつれ、パソコンという商品ジャンルの持つ他の応用は、機能ごとに解体され、ネットワークAV機器やインターネット対応家電のような形で環境中にばらまかれるようになるであろう。

このような判断から、最近ではこの方向性を前面に出してBTRONの研究開発を行っている。具体的にはネットワークでの既存の「オープン」環境で特に遅れていると考えられる多国語環境に重点を置いている。この分野は、次に述べるTRONの新しいステップにおいても重要なものである。

TRONの新しいステップ

正しさの再検討

TRONプロジェクトでは、当時はまだ片りんもなかった「どこでもコンピュータ」環境を仮定し、アイデアのレベルで突き詰め基本的な可能性の追及を早い段階に完了した。そのため、いわば15年目の現在に至ってようやくそのコンセプトが実現されはじめているともいえる。

しかし、技術進歩のスピードを考えると15年前に考えた仮定が——現在は正しいにしても——これからの15年のうちで正しくあり続けるかの保証はない。

たとえばITRONにおいて、当初仮想マシンを採用するという検討も行った。さまざまなマイクロプロセッサを利用する組込みシステムの分野で、標準化を行おうとする場合、それが最も理想的なモデルであるからである。しかし、組込み制御プログラムの場合、開発工数をかけても個々のプロセッサに最適実装し、少しでも高性能で安いプロセッサを使い製品の量産単価を少しでも抑えた方がいいということとなった。このためITRONではシステムコールレベルでの標準化にとどめ、さらに細かく

実装レベルを規定することで、製品に適した実装バリエーションを許すという方向をとった。それにより、完全な移植性は実現されないにしても、ある程度の移植性は確保されるし、開発者の教育の標準化というだけでもメリットがある。この判断がITRONの現在の標準化、普及化にあたってカギとなったことは事実である。

しかし、組込み制御プログラムの複雑度がある範囲に収まること、組込み用プロセッサにおける技術進歩が性能はそのまま価格が下がるという方向に行き続けること（価格維持のために最低性能を上げるというハードディスクのような方向に行かない）といった前提条件が崩れれば、この方針の「正しさ」も再検討が必要になる。

さらにいえば、HFDSのように、多数の異なるバージョンのシステムが開かれたネットワークに抜き差しされる中で、協調動作を行うような応用を考えると、どうしても実行しながらのネットワーク経由のプログラムのロードという必要が考えられ、これは仮想マシンを必要とする。事実、TRONでもそのような必要から、互換性と実行効率のバランスをとれるように考えた、ITRONとJavaVMのハイブリッドのJTRONを提唱している。

ITRONでの検討は、逆にチップの多様性のために完全な標準化ができないならば「オープンアーキテクチャ」の考えに基づいた標準MPUを開発しようという方向に進み、TRONチップにつながった。しかし、これもやはり企業側の希望との摺り合わせで高性能MPUを対象としたため、先に述べたようなパソコン市場の特殊性に翻弄される結果となった。アプリケーションとOSの関係のような周辺チップの整備が当初計画のようには揃わず、チップのリリースが遅れるなどの悪条件もあり、アメリカのMPUが覇権構造を確立した。

新たな仮定

ここで、考えなければならないのは、TRONの次のステップにおいて、コンピュータのインタフェース階層のどのレイヤを対象とするかについて15年前と同じでいいかということである。15年前は、すでに述べてきたようにそのレイヤは主にOSとMPUであった。他に

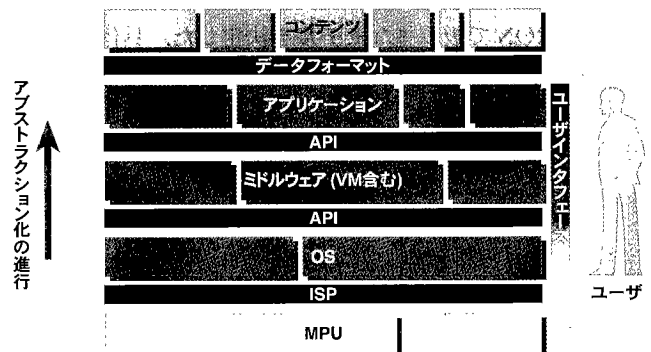


図2 コンピュータのインタフェース階層

もユーザインタフェースとデータフォーマットを対象としたが、これは当初BTRONの中に含まれていた。その後ユーザインタフェースについては「トロンヒューマンインタフェース標準」として独立し、パソコンのGUIだけでなく、物理的スイッチを含むコンピュータ組込み機器の操作まで範囲を広げた設計ガイドラインとして一応の完成をみている。

ここで新たな仮定として、コンピュータのインタフェース階層として図-2のような構成を考える。そして、新たなTRONのステップを考える時、MPUについては標準化の必要を感じないし、OSについてもBTRONの完成を達成すればそれ以上の必要を感じないことに気がつく。

実際現在のネットワークがらみのクライアントサーバ(以下C/S)システム開発においては、重点は徐々にミドルウェアと呼ばれる部分に置かれてきている。そこでは、MPUが何であるかはあまり話題にならないし、OSについてもしかりである。

つまり、15年の技術進歩により、図の下の方からアプリケーションが実用レベルで進んできているとも考えられる。次のステップ—それが15年になるか、加速度的進歩を考えると10年かもしれないが—においては、この傾向が進むことが必然であり、最後に標準として残るインタフェース階層はデータフォーマットであろう。これを新しいTRONの仮定とする。

ついでにいえば、やはり最後まで取替え可能でなく残るのはコンテンツのみとなる。コンテンツより下の層はいわば道具であり、極端にいえば役に立つモノが1つあればいい。そのためいくら開発に労力がかかっても完成すればコピーにより増やすことで、そのコストはゼロに近づく。これに対し、コンテンツはその1つですべてが賄えるというものではない。なんとなれば、どんな名作映画であっても、それだけあれば他の映画はいらないということはないからである。

新たなモデル

この仮定に基づき、MPUとOSに代わる次のステップのTRONで扱う主なモデルを決定した。具体的には、基礎プロジェクトはデータフォーマットを、応用プロジェクトはコンテンツを主に考えている。

データフォーマットについては、本来TADがあったが、従来はBTRONの一部と考えられていた。しかし、次のステップでは、HFDS実現のためにむしろこのTADをTRONの中心と考える。そして、そのために必要で、TRONで開発しなければならないTADの持つべき機能を以下と規定する。

- 人、モノ、環境など現実世界の属性情報の意味記述
- 文字主体の文化的コンテンツの取扱い

TRONの第1ステップから、TADを中心とする新たなステップへの関係を、図-3の開発ロードマップに示す。

属性記述 TAD

このような枠組みにより、最終的には現実世界の対応モデルをコンピュータの世界の中に構築し、それをもとにコンピュータが現実世界を理解できるようにすることを考えている。HFDSで考えているような、情報処理能力を持ったインテリジェントオブジェクトからなる環境の総体による即応的な調整には、コンピュータが対象とする人、モノ、環境などの状況を把握していることが重要であるからである。

ここで「現実世界の理解」というのは、本格的な人工知能のようなものを指しているのではない。TADで考えている意味記述は、異なるコンピュータ間でメタタグを付けて送り、送られたコンピュータでどういう処理が可能かが一般的に分かるという標準的な記述である。

ただし、ここで考えなければならないのは、単にメタタグを付けただけでは、今度はそれは何かということになるということである。ネットワークの中で、このタグが「理解できる」というのは、そのタグの定義がより基本的なタグとの関係でどんな処理ができるかを含め定義されており、必要に応じてその定義が取り寄せられるという環境を構築するということである。このような環境とすることで、応用ごとの細かい必要に応じていつでもタグ定義を追加できるようになる。

TRONではこのためにオブジェクト指向型の属性記述言語⁶⁾を開発し、これにより、拡張可能な分野ごとの記述クラス・ライブラリとして、人、モノ、環境さらにはサービス、商品、顧客、取引条件など商品や収蔵物のカテゴリーごとの属性を定義する。具体的な個々の商品、収蔵物の記述はそのクラスのインスタンスとして、パラメータに具体値を入れて作る。クラスは継承関係を持てるので、「商品」という基本クラスから、個々の業界で必要な属性を追加していくことで、「文房具」さらには「ホッチキス」というような具体的なクラスを定義することになる。

このような枠組みを利用することで、ネットワークでの属性定義のための通信量を最小限に抑えることができる。クラス定義をキャッシングすることにより、よく使う分野の定義については通信せず、定義が更新された時や、取り込んだデータの中に知らないタグがあった場合のみ最小限の通信を行うようにできる。

さらに、オブジェクト指向とすることで、個々のクラスの継承関係を逆にたどり相互の項目を基本クラスからどう展開しているかを知ることができ、2つの異なるクラスベースに記述された属性を相互の関連に基づき翻訳できるようになる。

TRON 言語環境

TRON 言語環境は言語層、文字属層、スクリプト層、フォント層の4階層からなる。入力アルゴリズム、表現アルゴリズム、ソーティングアルゴリズム、文字写像アルゴリズムの切替は言語層で管理され、文字属層の実際の文字を処理する⁸⁾。この構造により、「言語層＝日本語、文字属層＝ラテン文字」として「ローマ字表記日本語」といった環境も別個に処理できる。文字属層とスクリプト層は内部的記述と外的表現の間の写像を行うための層である。これにより、たとえば“fi”のような合字の場合、内部的には“fi”で記述し検索等も可能でありながら、表示は“fi”で美しく行うといったことを環境レベルで処理できる。

TRONの考える言語処理をコードだけのものと考え、「コードだけで～のような問題は解決できない」といった、批判も見かけるが、1987年の時点ですでに上記のような環境を提示しており、これはTRON言語環境全体を知らないことによる誤解である。

さらに、単にTRONコードといっても、ここでいうコードは従来のコードの概念とだいぶ異なっている。これは、漢字という文字が時代により使われる場所により変遷し、新しい字が生まれたり、また別と思われていた漢字が同じ文字と見なされるようになるなど、集合として閉じていなくても、決定してもいないという事情による。これは何も過去の文献だけの問題ではない。文字として意味的には同一と見做せる新字体と旧字体の差や異字体が、

地名や人名、小説や詩での効果など、使用目的により明確に別の文字として処理しなければならないといった、非決定性を漢字は本質的に持っているのである。

文字集合が決定しない以上本来は決定された変換表を前提とするコードという概念は漢字にはあてはめられない。TRONでは、字体、土地、用途などで移り変わる文字の「異同感」や教育用漢字等の制限、派生や統合等の文字間の関係性、などを納めた「漢字感データベース」を背景に「閉じていない」——いわば「オープン」な文字コードという新しい概念を提案し、その構築作業を文科系の専門家とともに進めている。

文化的な文字コンテンツの取扱い

先に述べたTADの持つべき機能の2つ目については、少し説明が必要であろう。文字情報の取扱いより高度なハイパーメディアの取扱いがすでにある程度見えているのに、なぜいま文字情報か、ということである。これは、2つ理由がある。

1つはコンピュータにおける文字情報の取扱いが最終的に紙に出力するという前提で考えられてきたため、コンピュータ上で文字情報をどう表示するかということについての文化的な考察がまだ満足になされていないということである。これに対し、マルチメディアで扱う画像、動画、音楽については、テレビやオーディオ機器の出力デバイスとコンピュータハードウェアの差がほとんどないため、そのような問題が少ない。

もう1つの理由は画像、動画、音楽と違い文字情報が国ごとの文化の違いと深いかかわりを持つということである。ここでいう、文化的な文字コンテンツの取扱いとは、コンピュータ上での表示を前提とした出版文化の再構築である。出版文化は本の作り方から、ルビや柱の付け方などを含む美的感覚まで含む複雑な体系であり、たとえばアメリカで開発された技術がそのまま日本の文字文化の表現のために適しているとはいえない。特に多文字言語の言語とそうでない言語の違いは大きく、多文字言語の文化的コンテンツの取扱い機能については、TRONで開発することが適切だと考える。

文字コードだけについても、本来多文字言語に実感のないアメリカ主導のUnicodeでは不十分というのが我々の認識である⁷⁾。この分野での基礎的な仕様開発が、多文字言語を文化を壊すことなく取り扱えるTRONコードを中心とするTRON言語環境である。TRONコー

ドについては先に述べたようにBTRONと関連して現在積極的に作業を進めており、近々に成果を発表できる予定である。

新たな応用プロジェクト

TRONプロジェクトでは従来から、基礎プロジェクトで行っている仕様開発に対応する応用プロジェクトを並行して進め、検証を行い、仕様にフィードバックをかけるという進め方をとってきた。

従来もITRONやBTRONに対応させ、電腦住宅や電腦ビルなどのプロジェクトを進めてきたが⁹⁾、データフォーマットに対応する応用プロジェクトとして開始したのが、Electronic Commerce (以下EC) と博物館応用、電子ブック応用である。

すでに述べているようにTADでは、最終的には現実世界をコンピュータが理解可能なように意味記述することを考えている。

ここで重要なのは、ECにおける企業や業界、博物館などにそれぞれ、必要な属性記述があり、それらは、共通するモノもあれば、特異なものもあるということである。さらにいえば、新製品が出たり、新たな発見がなされることにより、新たな属性記述の枠組みが必要とされる可能性が常にあるということである。その意味で、最初から、完全な属性セットを規定することは不可能である。

そこで、我々の考える記述の枠組みでは、オブジェクト指向言語により、基になるクラスから、ローカルに、必要最小限の定義を加えることで、新しい属性記述のセットを生成することができるようにした。そしてその定義はネットワーク中に分散して保存され、解釈のため

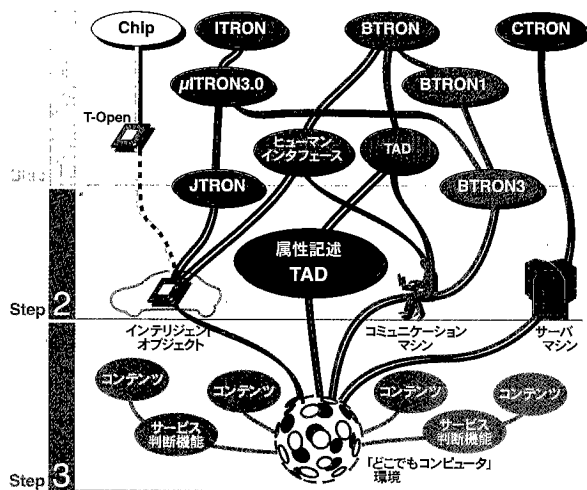


図-3 TRONプロジェクトのロードマップ

要になるとそれがネットワーク経由で参照される。

世界にあるものすべての属性記述といったことは、誰かがプロジェクトとして実現しようとしたら、あまりの労力に不可能と思われることである。しかし、このような枠組みが広く運用されることにより、全体のためのボランティアという意識はなく、あくまで自分達のビジネスのためというインセンティブを持った多くの人々によって、互いに関連付けられた属性セットのネットとして、世界の記述が広く分散的に充実していくことを期待している。

おわりに

時代の変化により前提が変わり、「オープン」のレベルは変化してきている。しかし、TRONと名付けたプロジェクトで最初から最後まで貫かれる哲学——それがあるからTRONであるといえる思想とは「オープンアーキテクチャ」であり、それは新しいステップに入っても変わっていない。

「オープンアーキテクチャ」は、「アーキテクチャの決め方のアーキテクチャ」——いわばメタアーキテクチャである。それにより「多様化と標準化の両立」を実現する。

そして、「多様化と標準化の両立」を最も必要とするのが、コンテンツの世界である。コンテンツは必然的に多様であるが、同時にそれが標準的に記述されていることにより、ネットワークを通して、統合的に利用することができるようになる。

HFDSは多数のコンピュータが、裏方として環境中に分散し、ネットワークを通して協調しながら、環境中のユーザに奉仕することを目指している。しかし、HFDS環境が構築されたというのはスタートに過ぎない。サービスの質において重要なのは、単なる環境条件だけでなく、そのユーザの欲求、嗜好、スケジュール、その時に

ユーザが得ている情報、環境中にどんなモノがあるか、さらには外の世界の交通や物流の状況といった「状況」をどこまでHFDSのコンピュータ群が把握できるかといった問題である。

もちろん当初はそれは直接個々の機能にかかわるような断片的な把握であろう。しかし、記述が「多様化と標準化の両立」を許せば、HFDSに組み込まれる環境機能が増え、個々の設備が進歩し、状況の把握も質、量ともに進歩する。一方社会全体としては、ECや博物館での利用のため相互利用可能な属性セットのネットが充実していく。

知能は決して知識と別に存在しているわけではない。むしろ、十分な質と量の、相互利用可能な「状況に対する情報」——知識が集まった時に初めて実用的「知能」が生まれるのではないかと思う。人間と高度な対話をするような人工知能より、「窓を開けてくれ」というユーザの音声指示に対し「なぜか、思ったとおり」の場所の窓が「思ったとおり」だけ開くというようなレベルの判断技術は十分な状況把握があれば比較的現実的に実現できるのではないかと考えている。

TRONが第3のステップとして「判断」に踏み込むかどうかは、現時点では断言できないものの、否定的である。なぜなら、TRONは「多様化と標準化の両立」を「オープンアーキテクチャ」を通して実現しようという環境整備プロジェクトだからである。第2のステップの基盤が確立すれば、いわばなしくずしにネットワーク中に相互利用できる属性情報が蓄積されることで——ちょうど今インターネットにいろいろなサービスが生まれているように——それらの応用として、多様な「判断」機能が開発されることを期待している。

その時にこそ、第五世代コンピュータの課題に対する筆者なりの回答ができるのではないかと考えている。

参考文献

- 1) Bush, V.: As We May Think, Atlantic Monthly, Vol.176, No.1, pp.101-108 (1945).
- 2) Kay, A. C. and Goldberg, A.: Personal Dynamic Media, Computer (IEEE COMPUTER MAGAZINE), Vol.10, No.3, pp.31-41 (1977).
- 3) 辻 篤子: 新星企業がIBMを食う, AERA, pp.6-9 (1991.7.16).
- 4) Tomita, A. W. and Sakamura, K.: Improving Design Dependability by Exploiting an Open Model-Based Specification, IEEE on Trans. Comput., Vol.48, No.1, pp.24-37 (1999).
- 5) Sakamura, K.: After a Decade of TRON, What Comes Next?, Proc. of the Eleventh TRON Project Symposium, pp.2-16 (1994).
- 6) 倉光君朗, 坂村 健: オープン分散ECアーキテクチャを実現するための商品記述言語の提案, デジタル・ドキュメント・シンポジウム'98, 情報処理学会 (1998).
- 7) Sakamura, K.: Multilingual Computing as a Global Communications Infrastructure, Proc. of the Twelfth TRON Project Symposium, pp.2-14 (1995).
- 8) Sakamura, K.: Multi-Language Character Sets Handling in TAD, TRON Project 1987 (Proc. of the Third TRON Project Symposium), pp.95-111 (1987).
- 9) Sakamura, K.: TRON Application Project: Gearing Up for HFDS, Proc. of the Eighth TRON Project Symposium, pp.2-14 (1991).

(平成11年2月9日受付)