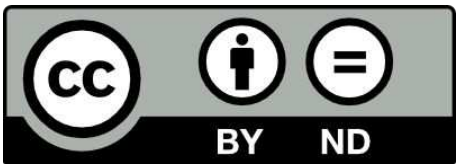
論語とコンピュータ

漆畑 晶



論語とコンピュータ

知識の共有と礼

オープンソースソフトウエアの最大の課題は何でしょうか。世にあるオープンソースソフトウエアの多くは無料ソフトウエアです。皆様も日頃オープンソースを利用して、お金を払ったことは無いのではないでしょうか。良いものをお金を払わず利用することはありあす。その時はお礼を言うものです。ただでもらっておきながら、お礼を欠いていると、おかしな事になります。価値観が歪んで参ります。

そうは言っても「礼」という言葉を聞くと身構えてしまうものです。その気持ちは理解できます。保守党の政治家等で「礼」を声高に叫んで自身の都合を押し付けてくるような人は世の中にいます。そのような乱用を避けるの方法があります。どうするかというと、ソース（原典）を当たるのです。(*1)

言葉のソースを調べるには言葉や文字の成り立ちを調べる方法と、思想を記した古典を読む方法とがあります。

まず礼の字を見てみましょう：　礼　禮

左の簡字体は旁（つくり）の方が跪（ひざまづ）く人の姿とされています。右の旧字体の旁の下には「豆」が見えますが、これは今日で言う「マメ」ではありません。お供え物を載せる器だとされています。上に見える「曲」は串に刺した団子の絵とされています。「曲がる」のとは違うという説が有力なようです。(*2)

どちらの字体も左は示す偏で、この偏は神を象徴します。昔の農業を考えるとこの字に表現されている人の神に対する思いは理解できるでしょう。最近の日本では東日本大震災の大地震、大津波で人の力を越えるものについて考えたました。その時は「想定外」という表現が使われました。さてこの「想定」がプログラミングで大事になる局面があります。開発過程でバグを無くすために検査を行いますがその検査の項目を決めるのには想定が必要になります。いかに見落としのない、的確な想定をするかはプログラムの品質に影響します。

中国の博物館にいくと祭器としての「豆」が展示されているのを見ます。今で言う「マメ」(植物、食品)はこの器にお供え物として載せられる事が多かったのでしょう。マメを煮てお汁粉を作れば友達を喜ばせることができます。しかしそれは一度だけです。同じマメを友達の所に持って行って育て方を教えれば彼は何度でもお汁粉を楽しめます。そしてさらにその友達に楽しみを伝えることができます。友達の輪は広がって行きます。こんな魔法みたいな、奇跡のような話、神様に感謝しなければなりません。これがお汁粉のようにたまに楽しむ程度ならまだしも、生業としてマメを栽培するようになり、それに生活がかかっている場合、小さなマメといえども軽視はできません。軽視していると大変な事になります。

もう一つの考えも紹介しておきましょう。食物は人にとって不可欠のもの、これなくして人は生きて行けません。植物の生産、安定的供給はたいへん重要なのでそれ

を行う業者が必ず利益を出すようにしておく必要があります。一般人が勝手に種を蒔いて植物を育てたら正規の業者の利益が減ります。正規の業者の利益が損なわれてはいけないので、一般の人の植物栽培は禁止しよう。そういう考えもあるようです。

これについて論語は何と言っているでしょうか。

　　子曰　君子喩於義　小人喩於利

利によって物事を進めようというのは小人の考えです。一方君子は義、即ち公正さを考えているものです。「義」は「犠牲」の「犠」にも通じます。正義のためには損な事も引き受けなければならないと君子は理解しています。

ところでここで論語は「善人、悪人」と言わない所が大事と思います。小人はその狭い見解の中で小人なりに善や悪を考えているものです。

孟子には「辞譲の心は礼の端（はじまり）也」という言葉があります。「辞譲」という言葉、最近では聞かれなくなったかもしれませんが、「辞退」の「辞」と「譲歩」の「譲」です。譲り合いの心が礼の始まりだと孟子は言っているのです。「知的所有権」と言いますが、その権利があるからと言って最大限主張しない人もいます。人は社会の中で生きるものですから、他者への配慮から権利の主張を控える、そういう考えがソースコード開示、変改造と再配布の容認の背後にあると思います。(*3)

「ハッカー」の本来の意味

さて礼を欠いた歪んだ価値観にどんな悪影響があるか、一例としてハッカーのお話をしたいと思います。

新聞やテレビではハッカーは悪いことする人ですが、本来は優秀な技術者という意味だったという話は聞かれたことがあると思います。ハッカーの語源は英語の hack という動詞です。斧などで勢いよく叩き割るという意味です。確かに乱暴ですね。しかし、次の漢字を見てみましょう：

　　分る　解る　判る

どれも字の中に刀が見えます。「わかる」ことは「わける」ことなのです。これだけはありません。「判断」の「断」、「分別」「識別」の「別」、「分析」「解析」の「析」はどれも「分ける」意味の字です。「ハッカー」は理解がすばやい人なのです。

荘子に「庖丁解牛」というエピソードがあります。中国人にハッカーの本来の意味を理解してもらうのにこの話をするといいでしょう。「庖丁」は日本語の「包丁」の語源になった言葉ですが、ここでは人の名前、さらに言うと、「調理場にいる人」という意味です。この人は牛を解体する名人です。彼によれば動物の体には自

然にできている隙間があり、そこに刃物を入れれば無理なく切断できる。刃も傷まない。下手な人は固い所を力ずくで折るからすぐ刃が欠けてしまう。ハッカーとは本来、庖丁氏が牛を解体するようにプログラミングを行う人を言う言葉でした。ところで今日は「論語とコンピュータ」という演題ですが、これは老荘思想ですね。

UNIX の原開発者のひとり、ダグラス・マッキルロイ(Douglas McIlroy)氏は「プログラミングにおける真のヒーローとはコードを減らしていく人だ」と言います。プログラムをよく観察して無駄を省く、重複を統合する。こうしてプログラムを減らしていく技術者こそ、本当に優秀な人だと彼は主張します。ちなみにこのマッキルロイ氏はパイプ(pipe)を発明した方です。(*4)皆さん、プログラムを減らして人からほめられた事、おありでしょうか？それができる技術者、お知り合いの中にいますか？

ハッカーの正しい意味を理解していないと、間違った人を相談相手にしてしまう危険性があります。セキュリティにおいても問題が起きます。ウイルス対策ソフトは年々大きさを増すばかり。足し算ではなく、引き算が正しい対処方法かもしれません。それを見落としてしまう危険があります。

漢字や漢語のお話をしてきましたが、その意義は三つあります。第一に、ここで見てきたように、新しい概念をよりよく理解し、人に伝えるのに漢字漢語は有用です。第二に、ソフトウエア開発における協業は国際的なものになっています。漢字を使う中国の方と意思疎通を図る事は重要です。第三に、時代の要請に応える漢語の新な使い方の考案、古代の字体や古典の研究は情報が公開されている中で、誰もが参加できる環境で行われています。言葉の研究の協同作業には歴史の積み重ねがあります。同じく協働作業として進められるソフトウエア開発の参考になるでしょう。

ベル研究所とは

さてこれから UNIX がどのような環境で生まれたのかお話しします。Linux は UNIX 系の基本ソフトとされています。その UNIX はベル研究所で生まれました。電話会社の研究機関ですが、どういう所でしょう。

電話はアレクサンダー・グラハム・ベル(Alexander Graham Bell)によって発明されました。ベル氏は電話会社を起業して発明を事業化しました。最初はベル電話会社でしたが、後にアメリカ電信電話会社 (AT&T) と呼ばれるようになります。初めは競合会社が多く、電話業界は混乱していました。ベル氏は電話の特許を取得していて、それを根拠にライバルを法廷で訴えました。一方この特許には異議を唱える者がいました。

20 世紀初頭、セオドア・ヴェイル (Theodore Vail) という人がベル氏の起した電話会社の社長になります。彼はアメリカ全国一律の電話サービスを目指し、法廷闘争を控え、競合会社を目立たないようにしかし着々と買収していきました。競争相手が消え、電話会社が一社となったら、電話料金をつり上げてくるのではないかと人々は心配しました。そこでヴェイル社長は政府と取引をします。電話事業の

独占を認めてもらうかわりに、電話料金は政府が決める公定価格とすることにしました。

ヴェイル社長は利益は必要だが、利益は最大化しない、という経営方針を表明しています。孟子の言う「辞譲」の精神に通じる所があると言えます。日本でもお馴染みのP.R.という言葉を言い出したのはこのヴェイル氏です。P.R. は会社の活動を世に知らしめる宣伝活動の意味ですが、元の英語は Public Relations、すなわち一般公衆との（良好な）関係（の構築と維持）という、もっと広い意味があります。

電話料金を自由に値上げできない電話会社は利益を増やす方法がありました。それはコストの削減です。規模の利益を活かして大量生産、大量購入する、部品の規格を統一することもしました。それとともに取り組んだのが各種部品の品質の向上です。電話網は広いアメリカの国土に張り巡らされていて、多くの電話線は地中埋設されていて、途中に複雑な増幅回路がついていました。また当時電話機は電話会社の所有で、利用者は毎月レンタル料を支払うしくみでした。故障の場合、電話会社の技師がすぐやってきて、無料で修理してくれます。故障が少なければ、メンテナンスの費用が節減されて会社の利益が増えるのです。優秀な科学者技術者を雇い、耐久性の高い、故障しにくい部品の開発、改良の研究をさせました。そのための研究組織がベル研究所です。研究の成果で故障が減り、会社の利益が増えればそれはベル研究所に投資され、さらなる研究開発が行われるという循環がありました。AT&T は大きな利益を出し続け、ベル研究所には潤沢な資金が供給されました。

ベル研究所から生み出された技法に統計的品質管理手法があります。統計的手法は、工場の製造品の品質を測定し、品質向上に役立つものでした。また、部品の寿命や故障頻度を事前把握するのにも有用でした。この技法を確立したのはベル研究所のウオルター・シューハート (Walter Shewhart) 博士です。もう少し詳しく言うと、当時調整し過ぎると却って品質が不安定になるという問題がありました。この問題に答えるのがシューハート氏の手法で、そのポイントは一般要因と特殊要因の区別とされています。

シューハート氏とともに統計的手法を研究したのはエドワーズ・デミング(W. Edwards Deming) 博士です。デミング氏はベル研究所を離れ、国勢調査局に移籍しましたが、戦後 GHQ の随伴員として日本に派遣されます。戦争直後の日本は混乱状態で人口も正確に把握できていない状態でした。当時の日本ではシューハート博士の名は一部技師の間で既に知られていました。来日したデミング氏がその弟子だということで、統計的手法について教えを請いました。(*5)日本の各産業の管理者は、敗戦を通じて日本の工業製品の品質が米国のそれに遠く及ばない事を思い知らされていました。日本は太平洋戦争（大東亜戦争）の開始時、戦艦大和、ゼロ戦に自信を持っていましたが、船や飛行機は実際に製造すると品質が安定せずに設計通りの性能が出せないことが多々ありました。デミング氏の講義に出席した人たちは大変熱心にその教えを聴きました。日本では毎年、品質管理に成果のあった企業に「デミング賞」という賞が贈られます。

ベル研究所では科学的経営管理の研究も行いました。イリノイ州シカゴ郊外で交換機部品を製造していたホーソン(Hawthorne)工場ではデミング、シューハート両氏が統計的手法を実際に適用するとともに、工場内の温度や照度がどれほど労働者の生産性に影響するかという詳細な調査も行われました。これはホーソン実験(Hawthorne experiments)と呼ばれていて経営学の分野では有名です。

トランジスタ、デジタル通信、UNIX

AT&T の潤沢な資金を使ってベル研究所では各分野をリードする研究員を雇っていました。最も有名な人は通信理論のクロード・シャノン(Claude Shannon)博士ではないでしょうか。ベル研究所では半導体の最先端の研究も行われました。電話網には増幅器が不可欠でしたが、そこに使われる真空管はよく故障しました。抵抗やコンデンサのような静的部品に置き換えたいという考えがありました。1952 年にベル研究所で半導体トランジスタが発明されます。しかし AT&T 社はトランジスタの製造に乗り出して利益を得ようとしませんでした。あくまでも電話事業に専念するのです。半導体トランジスタができると通信網を減衰しないデジタル信号を利用したものに置き換えていこうという機運が生まれます。デジタル通信の交換機は一種のコンピュータです。そこでベル研究所はコンピュータの優れた専門家を積極的に雇って開発に当たらせます。しかし、デジタル交換機が完成しても、AT&T はそれが本質的にはコンピュータである事実は極力宣伝しないようにしていました。

ベル研究所のソフトウエア研究部門にはパイプを発明した上述のマッキルロイ博士、c 言語を生み出したカーニハン(Brian Kernighan)博士、リッチー(Dennis Ritchie)博士を始め、優れた技術者がいました。UNIX は研究研修用のオペレーティング・システムという位置付けで、AT&T の正規業務にはほとんど使われなかったようです。加入者の料金算出請求は IBM あたりに任せていたと思います。UNIX はアメリカの各大学に安い使用料で提供され、学者学生がソースコードを閲覧し、プログラムを改造するのを容認していました。電子計算機は会社の本業ではない、UNIX OS は研究から生まれた副産物という意識があったからです。また、政府との取り決めでソースコードを要求した者には提供しなければならない、という決まりだったようです。しかし独創的な構想がいくつも込められた UNIX に触れた多くの学生や研究者が啓発を受け、UNIX との関わりを通して得た知見を活かし、情報工学発展に寄与した事は間違いありません。(*6)

ベル研究所で発明された半導体トランジスタはやがて AT&T の主力事業に影響していきます。トランジスタを利用したデジタル通信会社がいくつも設立されていきました。これらの会社は AT&T が大きな利益を出していた長距離の通信に参入していきます。電話料金は市内通話が安く、長距離が高い体系でした。（この料金体系は考えてみれば当然で、市内なら徒歩や自転車に乗って顔を合わせてお話しできますが、長距離となると昔なら寝台特急、戦後なら飛行機に乗らなければならず、宿泊代もかかります。）電信電話事業は自分達が独占する事になっていたはずと考えていた AT&T は不満を露わにします。一方で電話機のレンタル料収益が通信網を補助しているカラクリが独占禁止法に抵触するという指摘もなされます。結局米国の通信事業は各社の新規参入が自由化され、一方 AT&T は通信以外の分野に参入する経

営の多角化が認められます。(*7)　この裁定が出たのが 1982 年で、発効は 1984 年、そのた頃から UNIX についての締め付けが強くなり、プログラムの改変やソースコードの配布に対する禁止が徹底され、ベル研究所で製作された最新版ソースコードも公開されなくなります。

GNU と Linux

こうした中、マサチューセツ工科大学人工知能研究所(MIT AI Laboratory)のリチャード・ストールマン(Richard Stallman)博士が登場、UNIX と全く同機能の OS を開発すると宣言します。1980 年代始めはソフトウエアが事業として成り立つという考えが広まっていた時期で、MIT AI 研の同僚も多く民間に移っていました。MIT では ITS という OS を自分達で開発して運用していました。ハードウエアが更新され、性能が飛躍的に向上したのを機に、OS はメーカーの提供するものを使う方針となりました。15 年かけて開発した成果は廃棄され、新たな OS のソースコードは非公開でした。ストールマン氏はソフトウエアに対する諸制約が人々の協力を阻んでいる事態を身を以て経験し、「開発者のコミュニティが崩壊した」と憂いていました。彼は「自由（フリー）ソフトウエア」を提唱して運動していきます。(*8)　そして誰もが自由に利用でき、ソースコード全面公開のオペレーティングシステムをゼロから作り上げると宣言します。目指す OS を GNU (グヌー)と名づけました。(*9)　OS を全部作り上げるのは大変なことですが、UNIX の要素を一つずつ互換ソフトを書いて置き換えていけばできると主張しました。最初はこの野望が実現すると考えた人はほとんどいませんでした。しかし GNU ソフトが発表されるとともにその品質の高さから注目されるようになり、利用者が増えるとともに、GNU OS 実現も不可能でないと考える人も増えて行きました。

1990 年代の始め、日本では徳川さんという技術者が GNU ソフトウエアの普及に力を尽くしました。(*10)日本ではまだインターネットが普及していない時期、彼のところに磁気テープを郵送するとソフトを書き込んで返送してくれました。このテープをワークステーションに読ませてプログラムをインストールするのですが、インストールは何の問題もなく済み、プログラムもバグが見当たらないのです。当時サン・マイクロシステムズのワークステーションの UNIX が先進的で品質もよかったのですが、コンパイラやインタプリタ(*11)のバグはたまにありました。自分で書いたプログラムが思い通りに動作しないと、大抵は自分の思い違いなのですが、コンピュータ側に原因があるとなると厄介です。OS の不備を回避するために不自然なコードが書かれていました。そのようなおかしな工夫が必要ないので、GNU 環境で開発される応用ソフトも品質が高いものとなりました。

コンパイラ、シェル、ライブラリ（*11)等 GNU の各構成要素は次々と完成し、残るは中核部（カーネル）だけとなりましたが、これは開発が難航しました。そんな折フィンランドの学生のリーナス・トルバルズ(Linus Torvalds)氏が自作のカーネルを発表します。このプログラムは Linux(リナックス）と呼ばれ、それまでに完成していた GNU ソフトウエア群と組み合わせるとオペレーティングシステムとして完全なものとなりました。やげてフィンランドの学生が独りで OS を作成

し、作者の名前からそれはLinuxと呼ぶ、との話が世界に伝わり、GNU という言葉は消えていきます。ストールマン氏はこれに怒っています。(*12)Linuxが出現した当初、一時的に組み合わせて使われていても、GNUとLinuxは別の道を歩むだろうという観測がありました。GNU はワークステーション用の本格的なOSで、そのワークステーションは何百万円もする高価な機材で個人では買えず、会社が買うものでした。トルバルズ氏はパソコンでも動く小型軽快なOSを構想していました。その線に沿っていくらかカーネル以外の部分も作成しましたが、今でも使われていて見るべきものはファイル・システムくらいです。(*13)(*14)

GNU という名称が聞かれなくなった理由はいくつかあると思います。ストールマン氏が最初に開発したのはエディタで、Emacs （イーマクス）と呼ばれます。このEmacsは好き嫌いがはっきり分れるプログラムです。Emacs は機能豊富で、何でもできます。そして大きなプログラムです。(*15)これは小さなプログラムを上手に組み合わせるUNIXの基本方針に反するもので、ここからGNU はUNIXから離れてその長所が損なわれると懸念する声がありました。(*16)

このOSの開発の道程が一般人はもとより、一般的な情報技術者の理解までを凌ぐものだったというのも大きな要因だったと思います。OSの実物がまだ無い中でそのOS用の応用ソフトを作成できる技術者は滅多にいません。OSの中核部分ができる前に周辺部分を開発、しかもそれが完成して実際に利用されるというのは普通想像できない事です。喩えるなら、ビルの２階から上を先に構築して、そこで人が生活している状態で最後に１階部分を造成して挿入する、そんな離れ業が行われたのです。UNIXの巧みな要素分解、GNUの高い汎用性、移植のしやすさがこれを可能にしました。移植しやすかったのはまずは情報が全面公開されていたから、そしてそこまでの開発においてノウハウが蓄積されていたからだと思います。

もう一つ、ストールマン氏はUNIXの開発母体のベル研究所に批判的です。UNIXのライセンスは何も自由ではない、一時ソースコードが流通し、改造が行われていたのは本来は禁止だったものを見逃していただけ、と彼は言います。しかし、UNIX には要素分解、入出力をパイプで結合する処理、あらゆるものはファイルという構成等、基本的なアイデアのレベルで評価すべき所は多いと思います。これらを生み出してくれた先人に対するはっきりした敬意がストールマン氏にあったら違っていたかもしれないと思う事があります。

UNIX が誕生したベル研究所を中心に歴史についてお話ししました。今我々が日常的に使っている便利なものを作り上げてくれた先人について考える、というのも礼の一つだと考えています。技術者や学生はもっと歴史に関心を持って欲しいと感じています。それを知る事は手元にある道具をより有効に使うことにつながると思います。今や便利な携帯端末は広く普及し、日常的に使われています。どのようなソフトウエア技術が組み合わさってこれが実現したのか、専門の技術者ばかりでなく、一般利用者にまで理解が広まることを願います。

２０１８年５月１２日　漆畑晶

注
(*1) source code　コンピュータでのソースコードは人が書く形式のプログラム。数式と命令の組み合わせ。

(*2) 白川静著「字統」等参照。

(*3) これらはオープンソースの思想の基本とされているが、オープンソースという用語は 1998 年に登場した。最初は自由(フリー)ソフトウエアと呼ばれていた。

(*4) 一つのプログラムの出力を他のプログラムの入力とする機構。簡単な機構ながら、応用範囲が広い。パイプの使い方が上手な人はきっと UNIX の達人である。

(*5) 彼らが日本科学技術連盟（日科技連）の創立時のメンバーである。

(*6) その後に登場したコンピュータ言語はこの c 言語の影響が強く見られる。マイクロソフト社が大成功を収めた MS-DOS には UNIX の影響が見られる。

(*7) この時点で電話会社が地域分割された。数年後日本でも電電公社が民営化され NTT が誕生した。

(*8) 彼はプログラムを実行する自由、調査して改変する自由(そのためにはソースコードが読めることが必要)、再配布する自由、改変した版を配布する自由が不可欠と唱えた。

(*9) Gnu はアフリカに生息する野牛。GNU's Not Unix と頭文字を取った略語で、その中に名称自身を入れるのが当時コンピュータ関係者ではやった機知。UNIX では無い事を強調するのはゼロから書かれた全く別のプログラムで、ベル研究所の著作権が及ばないことを主張するため。

(*10)日本で早くから個人ホームページを持っていた人で、そこでは三つ葉葵の印籠が画面に表示され「この紋所が目に入らぬか」との台詞が聞かれた。将軍家の末裔の方と聞く。

(*11) 人が記述するソースコードを中央演算処理装置が理解できる実行形式に変換するプログラム。OS の主要要素である。シェルは人が実際に接する OS 外殻部。ライブラリはプログラムの共通部品集。

(*12) GNU/Linux と呼ぶべきと主張しているが、賛同者は少数派にとどまっている。GNU/Linux の呼び名に反感を持つ者は積極的な運動を展開していて、Wikipedia から排除しようという動きも見られる。

(*13) file system　磁気媒体の 0 と 1 の羅列を意味づけしてファイルやディレクトリ（別名フォルダ）を構成する規程。

(*14) GNU は UNIX でない事を強調したが、Linux は UNIX 系 OS であると宣伝されたことも混乱の原因となった。

(*15) editor エディタは人がプログラムを書くための道具だが、英語や日本語の文章を書くワープロにもなる。Emacs はプログラム開発支援、電子メールの送受信、画像の閲覧の各機能及びゲームまで備えている。人工知能言語 Lisp を解釈する機能も備えていて、Lisp 言語で書かれたのプログラムを追加することによって自由に機能を拡張できる。

(*16) 上記注に見られるように、Emacs の機能拡張は Lisp 言語により行われ、愛用者の間では Lisp プログラムが流通した。Lisp 言語は興味深い言語だが、Emacs の外では余り使われていない。それでも、Linux と一般に言われている基本ソフトの UNIX らしい部分で人が操作するプログラムは多くは GNU 由来で、c 言語で書かれている。Android OS は Linux カーネルを利用しながら「Linux ではない」と謳っている。一方でマイクロソフト社の Windows 10 は「Linux が利用できる」と謳っているが、カーネルは Linux ではなく、マイクロソフトが独自に開発したものである。こうした状況は Linux という名称が指すのは一般に Linux カーネルではなく、むしろ OS の中でもカーネルの外側の層であることを示唆している。

参考書籍

１．「ハッカー」の本来の意味について

ハッカーズ
スティーブン・レビー著　古橋芳恵，松田信子訳

真のハッカー、およびそのように呼ばれる人が集った研究所についての本。ベル研究所以外の研究所での早期の情報工学研究の状況に詳しい。ハッカーにとって「処理を最少の命令で実現する」事がいかに大切なのか伝わってくる。世界初のコンピュータゲーム誕生の経緯も紹介。

２．ベル研究所について

世界の技術を支配するベル研究所の興亡
ジョン・ガートナー著　土方奈美訳

ベル研究所の研究と親会社 AT&T の事業方針、合衆国の政策の関係を詳しく解説。クロード・シャノン博士の人となりと業績、半導体トランジスタの発明、ベル研究所を去った研究者がカリフォルニアの現在シリコンバレーと呼ばれる地で半導体製造業を起業した様子など詳しい。トランジスタから携帯電話まで、今は身の回りに普通に見られるようになった技術のいかに多くがベル研究所由来かと知って驚く。結果的に失敗に終わった研究についても触れていて興味深い。

３．統計的品質管理について

デミング博士の新経営システム論
産業・行政・教育のために
W・エドワーズ・デミング著　NTT データ通信品質管理研究会訳

統計的品質管理を戦後日本に伝えたのはデミング博士だが、彼が教えたのは統計学の技法にとどまらない。企業と労働者、企業と企業のあるべき関係についての主張はいわゆる日本的経営に取り込まれ、経済成長の基礎となった。本書は日本で成功した手法をアメリカの企業経営者、管理職に伝えようと記されたもの。誰でも簡単にできるおはじきの散布実験を通して「調整し過ぎると、却って品質が悪化する状況」を紹介し、特殊要因と一般要因を区別する事の重要性を上手に解説している。

４．UNIX について

UNIX とはどういうオペレーション・システムか、言葉だけで説明するのは難しい。UNIX 系 OS は広く普及していて、多くの人が「UNIX を使える」と称し、履歴書にもその旨記す。しかし、大多数の利用者は主に応用ソフト上で作業をしていて、OS を使うと言っても、使い慣れたアプリケーションの起動等、限られた操作しか知らない。コンピュータは毎日使っていても機能のごく一部しか知らない人が多いが、UNIX 系はその傾向が特に顕著である。

ソフトウェアの道具箱
A．ロビンズ著
https://linuxjm.osdn.jp/info/GNU_coreutils/coreutils-ja_210.html#Opening-the-software-toolbox

短い文ながら、UNIX の思想に沿った小さなプログラムの組み合わせ処理を端的に解説している。情報技術者なら一度ここに挙げられている例を実際に試してみられるとよい。著者は GNU 版 AWK の製作者の一人。

UNIX プログラミング環境
B.W. カーニハン,R. パイク著　野中浩一訳

UNIX の使用方法の教書。開発者自身の著作で設計思想がよくわかる。開発経緯の話も所々に挿入されている。残念ながら古い本で用例は現在主流の GNU 由来のコマンド群と合致しない事が多い。

プログラミング言語 AWK
A.V. エイホ, B.W. カーニハン, P.J. ワインバーガー著　足立高徳訳

AWK 言語は UNIX から生まれ、UNIX には標準装備の言語。開発者自身による教書。多くのコンピュータ言語が手続き型言語である中、AWK 言語はフィルタ型言語と呼ばれる。入力の中である条件に合致したものを選び出し、必要なら何らかの加工を施して出力する。単純な発想ながら応用は広い。UNIX の思想がよく表れている。後の言語、特に Perl 言語に大きな影響を与えて、それらの発展の影で注目されなくなった。一方で後発の言語は人気が出て「何でも屋」となって焦点がぼけて UNIX らしさが失われしまったが、AWK 言語は原型をよくとどめている。この本は古いが、用例は今でもそのまま使える。

上記の「ソフトウエアの道具箱」の方法に沿った問題処理に柔軟性が求められる状況でこの AWK 言語の短いプログラムが使われる。これができるようになれば、UNIX らしいオペレーション（操作)が理解される。Perl, PHP, Ruby などを使い慣れているプログラマはそれらと AWK との違いを考えると UNIX の思想、言語の進化の歴史双方が見えてくるだろう。

５．自由ソフトウエア、オープンソースの思想について

フリーソフトウェアと自由な社会
リチャード・ストールマン著　ロングテール, 長尾高弘訳

コンピュータの利用については自由が何より大事だと主張している。その自由も自分の都合だけを押し通すのではなく、他者の自由を尊重するものでなければならない。自由を制限すると、人々の協力が阻害されるという問題についてもいくつもの実例を挙げている。自由を守るために積極的に活動することが必要と説く。ストールマン氏はプログラマだが、この著作は政治思想に属する。

フリーソフトウエア財団 GNU プロジェクトの公式サイトにストールマン氏の文章が掲載されている:
http://www.gnu.org/philosophy/philosophy.ja.html

伽藍とバザール
The cathedral and the bazaar
エリック・レイモンド著 山形浩生訳・解説

短い文章ながら有名である。文系の人にも広く読まれている。優れたプログラマであるレイモンド氏は、Linux の PR の中心人物であり、多くの有力企業をオープンソースに呼び込んだ実績がある。ストールマン氏より穏健とされ、オープンソースの実用性を主張している。

オープンソースソフトウェア
彼らはいかにしてビジネススタンダードになったのか
クリス・ディボナ, サム・オックマン, マーク・ストーン編著 倉骨彰訳

主要プログラムの開発者がそれぞれ自分の言葉で経験を語る短編エッセー集。

ソースコードの反逆
Linux 開発の軌跡とオープンソース革命
グリン・ムーディ著　小山裕司監訳

GNU プロジェクトと Linux カーネルの開発と普及の経緯を詳しく記した本。

著者紹介

漆畑 晶

プロフィール

- 画像処理ソフトウェアユーティリティ群 Netpbm の主要開発者
- FSF 創設者のリチャード・ストールマン博士と親交あり
- 中国の大学（上海復旦大学、南京大学、天津大学、中国科学院大学等）で講演多数

信州大学経済学部在学中に「国富論」やアメリカ産業史の本に加えて、「論語」「老子」などの古典を読んで引用する力を磨く一方で、R 言語による多変量回帰分析プログラムや、駅すぱーとの核心部である最短距離経路探索のプログラムを自作していました。

オープンソース・ソフトウェアについて

なにか良いものをタダで貰ったり使わせてもらったときは、必ずお礼を言うものです。お礼を欠いていると、おかしなことになってしまいます。誰でも自由に利用できるオープンソース・ソフトウェアも同じです。感謝を欠かしてはならないのです。

論語とコンピュータ

著　者：漆畑 晶

発行者：（特非）オープンソースソフトウェア協会
http://www.ossaj.org/
info@ossaj.org

発行日：2024年2月1日