システム管理論

システム（system）

システムとは、構成要素が連携し**干渉**（interaction）することによって全体として「ある目的」を果たすために構成された「ものの集合」をいう。構成要素は部品と考えてもよい。あらゆる場面でシステムという言葉を使うが、構成要素とそれら間の干渉が鍵である。例えば、自転車も一つのシステムと考えることができる。ハンドル、ペダル、サドル、チェーン、前輪、後輪、フレームなどが構成要素となり、構成要素が互いに力を伝え合って、全体として人間を乗せ移動するという目的を果たすことができる。この例では、人間も構成要素と考えることができる。一般にシステムを考える場合は、全体としての果たす役割が重要であるため、構成要素をどの程度抽象化した視点で考えるか、余分な要素をいかに排除してシステムを考えるかが重要である。上の例で、同じ自転車でも、競技用に早く走るための自転車、日常の買い物に利用する自転車、デザインを重視した自転車など、自転車としての用途に応じ構成要素をどのように選択し、また、構成要素をどのように干渉させるかがシステムを構成する上で考察の対象となる。

システムは実際に物理的に目に見える「もの」を扱う場合だけでなく、概念上の「もの」を構成することも多い。特に、研究段階において新しい概念を定義したり、仕組みを定義する場合には、システムとして抽象的に定義することにより一般的な議論が可能となる。計算機を学問として扱う計算機科学では、ソフトウェアという目に見えない「もの」を扱うのでシステムという言葉を多用し考察する。さらに、科学全般においても抽象的、普遍的考察を行う上でシステムをいかにうまく定義するかが重要である。以上の場合にも、構成要素、それら間の干渉に注目することが必要である。

情報とデータ（じょうほうでーた）

情報(information)とは、**データ**(data)という値を「ある目的」のために分類、整理した「もの」をいう。値には、文字や画像、音などマルチメディア上で扱う数値化したものも含む。最近、情報学という学問が注目されているが、情報学ではあらゆるものを数値として扱うため、その扱いが得意な計算機を中心に据えて議論を行う。しかし、情報学では、値としてのデータをある目的に向けていかにうまく構成し、整理し情報に変換するかが重要となる。まずは、多くのデータを収集することが必要であるが本質的には重要ではない。経営学の分野でも多くのデータを集めた後に、情報をうまく抽出し、計算機上の情報システムへ入力して、経営戦略のために役立たせるかが重要である。単に、計算機を効率よく扱うだけでなく、戦略立案のための手段として情報システムを活用するかがますます重要となってきている。

さらに、情報として整理した後に、**知**

識（knowledge）として情報や情報を操作する**手続き**（procedure）をパターン化したものを、蓄積することが必要である。最近の計算機の処理能力の向上は目覚ましく、多くのパターンを蓄積することが可能となったため、パターンも知識として整理するかまでが、情報を扱う上で考察すべき点である。

システムインテグレーション（system integration）

経営学において、企業全体をいかにシステム化し、経営戦略をたて効率よく運営し、利益に結びつけるかが重要である。企業活動全体をまとめシステム化し統合するかという**システムインテグレーション**（system integration）が鍵である。最近では、頭文字をとってSIと略して用い、企業に計算機を導入しコンサルテーションする企業をSIer といわれるようになってきた。歴史的には、主に大企業の一部門であった情報システム部門など計算機に注目して、情報システムを設計し開発する業務を担当していた部門が、ネットワーク技術を用いた複雑な情報システムが必須となってきている中で、様々な知識を応用したシステム作りをする企業、SIerとして独立する場合が多い。大企業とは関係なく、スタートアップ企業として起業し有名になったSIerも数多く存在する。システムインテグレーションには、単に、ネットワークまでを含んだ計算機まわりを担当するだけでなく企業全体をシステム化する必要から、企業活動、業務そのものにも精通している必要があり、幅広い知識が要求される。最近では、各企業に情報全体を統括するCIO（chief information officer）をおく企業も多くなってきており、企業の情報をいかに整理し、統括していくかが重要な中で、CIOをSI業務のトップに据え、企業情報統括と密接に絡ませながらシステムを構築することが重要になってきている。

オブジェクト指向（おぶじぇくとしこう）

元々は、世の中の物理現象をうまく記述するために考えだされた概念である。プログラミング言語の一つであるシミュレーション言語Simula67を開発した1967年に考えだされたといわれている。しかし、Simula67では**データ抽象**（data abstraction）といわれていた。その後、当時米国ゼロックスパロアルト研究所（シリコンバレー）にいたアラン・ケイが**オブジェクト指向**(object oriented)という概念として発展させ、オブジェクト指向言語Smalltalk-80を発表した。Smalltalkは1970年代より論文としては発表されていたが正式には1980年に発表したのでSmalltak-80という。オブジェクト指向という言葉は、プログラミング言語の一種であると勘違いされることも多いが、オブジェクト指向自身は概念を示す。オブジェクト指向という概念はプログラミング言語の分野だけでなく、現在では、様々な分野へ応用され、オブジェクト指向設計、オブジェクト指向オペレーティングシステム、オブジェクト指向データベースなど、非常に応用分野が広い概念である。

オブジェクト指向では、構成要素の単位を**オブジェクト**（object）として表現し、オブジェクト間の**メッセージの受渡し**（message passing）により処理が進むと考え**世界**（world）を定義する。オブジェクトには、**状態**（state）と状態を参照、変更す

ることができる**メソッド**（method）からなり、情報そのものである状態を**カプセル化**（encapuslation）する。全体の世界があり、**神**（god）や階層構造の頂点にある「もの」が存在しトップダウンに意思決定がされ構成要素が受動的に処理をするという概念とは正反対の概念である。各オブジェクトが能動的に処理をすると仮定するため、各オブジェクトは擬人化されて定義することが多い。各オブジェクトの関係、やり取りはすべてメッセージの受渡しを用いて定義する。オブジェクト指向には大きくわけて２種類あり、各オブジェクトがクラスから**個体化**(instantiation)し生成すると考えるクラスベースオブジェクト指向と、既にあるオブジェクトの**複製**（copy）から生成すると考えるプロトタイプベースオブジェクト指向が考えられている。最近では，Smalltalk-80に続いてジェームス・ゴスリンによって開発されたオブジェクト指向言語Javaの普及の影響によりクラスベースオブジェクト指向が主流となっている。オブジェクト指向には、その他、**継承**（inheritance）、**委譲**（delegation）、**並列性**（concurrence、parallelism）など様々な概念を含むようになり、さらに、オブジェクト指向と直行した概念である**アスペクト指向**（aspect oriented）という概念とも深い関係にある。

オペレーティングシステム（operating system）

計算機を司る重要なソフトウェア、**基本ソフトウェア**（basic software）の一つである。元々は、計算機が考案された1950年代から存在し、計算機の**記憶装置**（memory）や周辺機器との**入出力**（input / output）を司る制御ソフトウェアであり、人間が計算機を操作するために最低限必要なソフトウェアとして存在していた。「計算機、ソフトウェアなければただの箱」といわれるように、オペレーティングシステムがなければ計算機はただの箱である。

その後、計算機の目覚ましい発達によりさまざま機能を実現する中で、人間と計算機の操作系全般を司るソフトウェア群を示すようになってきた。マルチウィンドウシステムや、ネットワークシステムなど本来オペレーティングシステムとは関係なかった機能までも含んだソフトウェアがオペレーティングシステムといわれている。現在は、マイクロソフト社のWindows Vista、Windows XP、アップル社のMac OS X、オープンソースソフトウェアとして、Linux（Linuxにはベンダによりさまざまなバージョンが存在する)が主流となっている。

本来オペレーティングシステムは、計算機が非常に高価であったため周辺機器との入出力を司り、いかに効率よく計算機に処理をさせることができるかが重要な課題であった。記憶装置、通信装置、さらには処理そのものを**資源**（resource）として定義し、資源をいかに有効活用するかを研究する分野であった。

情報処理の単位を**プロセス**（process）または**タスク**（task）と定義（以下、プロセスという）する。第一段階としては、多くのユーザが様々な処理をプロセスとして要求する中で、マルチプロセッシングシステムをどのように実現するかが議論され、**TSS**（time sharing system）が考案された。その中心として、プロセスをいかにスケジューリングするかプロセススケジューリングが大きな問題であった。さらに、限られた記憶装置を用いて、いかに大きな処理をさせるかが問題となり**仮想記憶**(virtual memory)が考案された。現在では、ネットワークとの通信機能、ユーザとの操作系、いわゆる**ユーザインターフェース**

（interface）の議論もオペレーティングシステムにおいては重要な研究分野の一つとなってきている。

ソフトウェア工学（そふとうぇあこうがく）

効率よく、ソフトウェアを設計し開発するかを研究する学問分野である。システムとソフトウェアは非常に深い関係にありシステム設計論とも深い関係にある。1960年代の後半から基本ソフトウェアであるオペレーティングシステムを中心としたソフトウェア群が大規模化するにつれ、その品質が問題視され、ソフトウェア危機といわれるようになった。1970年代にソフトウェア危機に対処するためソフトウェア工学が生まれた。当初は、ソフトウェアそのものであるプログラムの構造に注目し、構造化プログラミングが主流となり、理論面、実践面から活発な研究がなされた。1980年代に入ると、ソフトウェアの開発への需要が増大し、ソフトウェア開発方法論や**道具**（tool）が数多く開発され、実践面での発達がある一方で生産性や信頼性への議論が高まり、理論と実践との乖離が際立って来た。オブジェクト指向言語とオブジェクト指向開発手法が叫ばれたのもこの時代である。1990年代に入ると、PCとインターネットの爆発的な普及により、ソフトウェアの大規模化、複雑化に伴って、生産性や信頼性へ要求が高まり、活発な研究が行われるようになってきた。2000年以後は、大規模ソフトウェアをいかに開発するかに注目した開発プロセス、マイクロプロセッサの発達により、あらゆる**装置**（device）に組み込む形式の組み込みシステム開発への注目が集まり、これまでと同様に、理論面、実践面の研究が盛んである。

システム設計、開発手法としては、最初に提案されたウォータフォールモデルが最も有名である。開発段階を要求定義、外部設計、内部設計、プログラミング、テスト、運用に分け、「手戻りなく」工場のラインで製品を作るようにソフトウェアを設計、開発するという手法である。しかし、現在ではウォータフォールモデルが否定され、「手戻りがある」開発手法、エクストリーム・プログラミング（XP）が主流となっている。

ネットワークとインターネット（ねっとわーくといんたーねっと）

1960年代に米国の軍事基地を接続する目的で始まったARPANETが現在の広範囲ネットワークシステムの始まりとされている（ARPAは、Advanced Research Project Agency、高等研究計画局の略で、その後、DARPA、Defense Advanced Research Project Agencyとなった）。ネットワーク接続には、互いに通信する間での**プロトコル**（protocol、通信規約）を定める必要があるが、ARPANETで考案されたTCP/IPの他にIBM社のSNA、ゼロックス社のXNSなど実際に大企業内で設計され使われたプロトコルが考案された。その後、**ISO**（International Standard Organization）により、SNAを元にした7階層のプロトコルが国際標準として定められたが、結果として最も単純であるTCP/IP（2階層）が世界に広まった。ネットワーク接続そのものは、1970年代までは、主に大組織で採用された汎用計算機のためのネットワークシステムためのものであった。1980年代にPCが登場したが、当初は専用線ではなく毎回、接続、切断操作を行わなければならない通信手段であり、1990年代より電話回線を使ったADSLという専用線方式の接続方法が考案され、PCでも簡単にネットワーク接続ができるようになってきた。

インターネットとは、元々あったネットワーク同士を接続する意味での**ネットワーク間接続**（Inter-networking）が由来であり、INTERNETという8文字の新しい単語が生まれた。PCによるネットワーク接続が爆発的に広まったのが1990年代であり、ADSLなど簡単な専用線接続が可能となったためと考えられる。その後、現在まで、世界中に張り巡らされたインターネットでさまざまなサービスが提供されている。代表的なサービスがWWW、電子メールである。

WWWは、World Wide Webの略で「世界に張り巡らされたクモの巣状のネットワーク」を示し、実際には、WWW上で提供されるホームページサービスを示す。

情報システム（じょうほうしすてむ）

計算機が経営の分野で活用されるようになり、情報システムといわれるようになってきた。情報システムのキャッチフレーズは、**EDP**（Electronic Data Processing）、**MIS**（Management Information System）、**DSS**（Decision Support System）、**SIS**（Strategic Information System）、**BRP**（Business Process Re-engineering）、**EAI**（Enterprise Application Integration）、**ERP**（Enterprise Resource Planning）と変遷している。EDPの時代には、数値計算のために主に計算機が使われていたころに、事務データをいかに効率よく処理するかに注目した。MISの時代には、マネジメント（管理）部門でいかに計算機を用いるかに注目したが、管理者層は電子メールすら自ら打たず、秘書に代筆させる時代であったため活用されることは少なかった。その後、DSSにより意思決定を支援するためのシステムとして、SISでは、当時流行した人工知能を用いて戦略的に意思決定するシステムとして研究が進められた。しかし、汎用計算機など大企業でのみ利用が進む、高価な計算機を対象とした研究であったため一般には道具として広まることはなかった。さらに、ERP、EAI、ERPとキャッチフレーズが変遷することによって、情報システムはPCとネットワークの普及に伴って、中小企業や一般にも活用されることになった。

最近では、経営の分野ではERPといわれているが、より一般的にインターネットを使うことが当たり前となり利用が進んでいる。大きく分けて**CMS**（Contents Management System）、**SNS**（Social Network Service）、**LMS**（Learning Management System）と分類することができる。CMSは、インターネット上で簡単にデジタルコンテンツを整理し、公開することができる情報システムで、XoopsやPloneなどが有名である。SNSは、登録した会員間でシステムを通して意見交換したり、情報を共有する目的で構築された情報システムである。日本ではmixiの爆発的な普及により広まったが、海外ではFacebookや、MySpaceが有名である。OpenPNEなどオープンソースソフトウェアとして個人や小規模組織で簡単に構築することができるSNSもある。LMSは、学習の場（学習支援）として提供される情報システムを指す。Blackboardという米国企業が開発した商用のLMSが最も有名であるが、Sakai（Sakai財団）、Moodle、CEAS（関西大学）など無料で配布され、広く使われているLMSもある。LMSも別の意味での**CMS**（Course Management System）ということも多い。最近では**CLE**(Collaborative Learning Environment)ともいわれている。

コンパイラ（compiler）

人間がプログラミング言語で記述したプログラムを機械語にコンパイル（compile、翻訳の意味）するためのソフトウェアである。計算機は、2進数で構成される機械語のみ解釈することができるので、プログラムを機械語に翻訳する必要がある。**コンパイラ**（compiler）は、プログラム（ソースコード）を字句解析、構文解析、意味解析、コード生成という処理を通して機械語に翻訳をする。字句解析は、プログラムを意味のある単位（字句）に細かく分解する処理を示し、構文解析は、字句を組み合わせた文法に沿って、曖昧ない記述であるかをチェックし、中間形式である構文木という木構造に変換する処理である。字句解析は、字句に分解する際には、最低限1文字に分解すればよいので**エラー**（error）は発生しないが、構文解析は、文法に合致した記述であるかをチェックするため**文法エラー**（syntax error）となることがある。文法に合致しているかを解析する手法として、大まかな文法からより細かな文法に沿っているかを解析するトップダウン解析手法と逆のボトムアップ解析手法がある。トップダウン解析手法としてはLL(k)、ボトムアップ解析手法としてはLR(k)が有名である。kは、先読みする字句の数を示す。多くのプログラミング言語の処理系では、1文字のみ先読するLR(1)が使われている。意味解析は、構文解析で生成された構文木に含まれる要素の意味を解析する。コード生成は、最終的な機械語に変換する処理である。対象となる機械語は、計算機の種類毎に異なるので複雑な処理となる。

最近の研究対象は、意味解析、コード生成が中心である。意味解析では、構文木より抽出された意味を詳しく調査しプログラムの実行効率の向上などに役立てている。自然言語処理で必須である形態素解析は、意味解析を発展した処理である。コード生成では、最適化に関する研究が活発である。

クラウドコンピューティング（くらうどこんぴゅーてぃんぐ）

1990年代からのインターネットの爆発的な広まりにより、PCからネットワーク接続を通してソフトウェアを実行することが当たり前となっている。これまでの**ローカル**（local）にHDDなど大規模記憶装置をおいて処理する方法から、ネットワークを通した**遠隔地**（remote）に大規模記憶装置をおいて処理する方法へと変化しつつある。この変遷を通して「こちら側」から「あちら側」へ情報処理の主体をおくという表現が使われるようになってきた。これら現象は1998年からのGoogleという強力な検索エンジンの登場がきっかけである。Google社は、検索エンジンを構築する中で世界中にネットワーク接続されたサーバで公開されている情報を自社のデータセンターに集積して情報検索を行う処理を通して、世界中の膨大な情報を分類、整理している。整理だけでなく「整理するための道具」であるソフトウェアをインターネット上に無料で提供している。Google検索エンジンの活用の広まりによってあらゆる処理が「あちら側」で処理することが多くなってきた。

一方で、ネットワーク接続の広がりと共に、セキュリティへの関心が高まり「あちら側」に情報を移して処理を行う方がよいか、「こちら側」で処理を行う方がよいかの議論が高まってきた。結局、セキュリティを高めて実行しなければいけない処理は、独自のデータセンターや商用のASPサービスを利用し、公にしてもよい情報は、Google社などで提供する「あ

ちら側」にある無料の記憶場所を利用するようになってきた。以上のような「あちら側で行う」処理形態をクラウドコンピューティングという。クラウドは、雲の意味であり、雲のように張り巡らされたネットワーク接続（インターネット）を相手に情報処理することを示す。クラウドコンピューティングに関する議論は始まったばかりであり、今後、利点、欠点が議論され、よりよい方向へ流れが進むであろう。