

# chapter - 8

---

オブジェクト用語辞書

AbstractFactory[Pattern]	具体クラスを明確にせずにオブジェクトを生成するためのデザインパターン。
ActiveX	Microsoft社が開発したインターネット関連技術の総称。主なものに、VBS ( Visual Basic Script ) , Java Script , ActiveXコントロール , ActiveXドキュメント , WinInet , IIS ( Microsoft Internet Information Server ) , ISAPI ( Internet Server API ) などがある。
Ada	1979年に米国防総省が規定した仕様をもとに開発されたプログラミング言語。Pascal や Algol をベースとした手続き型言語で、Fortran や COBOL の代替言語として開発された。並行処理 ( マルチスレッド ) や総称性 ( テンプレート ) が言語機能に組み込まれている。C++ の STL ( Standard Template Library ) の設計者である Alexander Stepanov は、Ada Generic Library で総称性を使ったコレクションクラスライブラリのアイデアを確立している。
Adapter[Pattern]	クライアントが求めるインタフェースと、既存のインタフェースを変換するデザインパターン。
Analysis Pattern	Martin Fowler による書籍「アナリシスパターン」で最初に紹介された、分析過程、主に概念モデリングの際に現れるパターン。例えば単位と数値を組み合わせて量を表現するQuantityパターンなど。
Another Level of Indirection	「もう一段の間接参照」を導入すると、ソフトウェアのほとんどの問題は解決できる、という法則。例えば、C++ において領域の解放忘れを防止するために、Handle/Body イディオムによって参照カウンタを用いる、OS で各プロセスに固有のアドレス空間を持たせるために MMU を用いて論理アドレスと物理アドレスをマッピングする、あるいは、プログラムの実行時ポータビリティを上げるために仮想マシンを定義してバイトコードを解釈実行する、など、コンピュータサイエンスの至るところで現われる。

Anti Pattern	プログラミング現場やプロジェクト管理で現れる、プロジェクトの悪しき兆候をパターンとして記述したもの。書籍「アンチパターン」には、コードが絡まって理解不能になる「スパゲッティコード」や、1つの解決法を覚えるとすべての問題をそれで解決したくなる「ゴールデンハンマー」といったパターンが紹介されている。
AOP ( Aspect Oriented Programming )	ゼロックス PARC で Gregor Kiczales らによって研究されているプログラミング原理。Dijkstra の「関心の分離」を保つために、オブジェクト指向のみではうまく補足できずコードに散らばってしまう視点を「アスペクト」と呼び、それらをクラスとは別に定義する。Java 言語用に開発されたツールAspectJ は、アスペクトをコンパイル時に Java コードに織り込むプリプロセッサ方式で実現されている。
Apple ( Apple Computer, Inc. )	1976年4月に Steve Jobs , Steve Wozniak らによって創立されたパーソナルコンピュータの開発・販売会社。1977年、業界初のオープンアーキテクチャを持ったパーソナルコンピュータ Apple を発表。1984年に発表した Macintosh の普及により、世界的企業に成長。MacOS は、Windows に並ぶ人気 OS であり、特にグラフィックスデザインやDTPの世界で利用者が多い。
Architecture Pattern	システムのアーキテクチャを表現するパターン。Frank Buschmann らの書籍「ソフトウェアアーキテクチャ」 ( Pattern Oriented Software Architecture ) で紹介された、Layers ( レイヤー ) , Pipes and Filters , Blackboard , MVC , PAC らのパターン。

base class	基底クラス。クラスの継承階層において上位にあるクラスを指す。C++言語の用語。派生クラスと対になる言葉であり、派生クラスを定義する際の派生元のクラス。基底クラスそれ自身が別の基底クラスから派生することもある。Java では「スーパークラス」という用語を使う。
BeOS	1990年に Jean-Louis Gassee が創業したBe社のパソコン用OS。マルチタスクやマルチスレッド、マルチプロセッサに対応しており、他のOSと比べて高速な処理が可能。当初は、同社が開発したハードウェア「BeBOX」を対象としていたが、後に Macintosh や PC/AT互換機で動作する BeOSも開発された。現在は、無料配布されている。
Booch Method	1994年 Grady Booch によって提唱されたオブジェクト開発方法論および記法。クラスを表現する雲形に特徴がある。書籍「Object-Oriented Analysis and Design With Applications」で紹介され、同書の初版には、雲形クラス図を書くためのテンプレート定義が付録として付いた。
Bridge[Pattern]	タイプとクラスを分離するためのデザインパターン。java.awt のように、多数のウィンドウシステムに同じ GUI クラス階層群を構成する、という設計での利用例がある。共通インタフェース階層と、各具体実装階層を橋渡しするのでこの名がある。欠点として、継承階層がパラレル・ツリー（並行木構造）となり、継承階層を修正するとすべての実装階層に影響が伝播する点が挙げられる。
Builder[Pattern]	複合オブジェクトの生成過程をカプセル化するためのデザインパターン。
byte code	特定のOSやハードウェアに依存しない、仮想マシン用のコード。ソースコードと実際にCPUで実行可能なネイティブコードの中間に当たる形式。特にSun Microsystems社が定めた Java仮想マシン用のバイトコードを指すこともある。Smalltalk もバイトコードと仮想マシンを持っており、Microsoft社の .NET 構想においても、IL (Immediate Language) と呼ばれるバイトコードと仮想マシン言語を持っている。

C	1972年に AT&T社のベル研究所で D.M.Ritchie, B.W.Kernighanによって開発された汎用のプログラミング言語。1989年にANSI (アメリカ規格協会)によって標準化された。豊富な演算子やデータ型、制御構造を持ち、構造化プログラミングに適している。特定のプラットフォームに依存しない言語仕様で移植性が高い。また、もともとOSなどのシステムの記述用に開発されたので、ハードウェア寄りの低水準な処理を記述することもできる。UNIX 自身が C で記述されている他、ほとんどのオープンソースソフトウェアは C 言語で記述されており、Eric Raymond によればハッカー文化では最強の言語である。また、C++, Objective-C, Java, C# などのオブジェクト指向言語は共通してシンタックスを C から継承しており、「プログラミング言語のラテン言語」と呼ばれることもある。
C#	2000年6月にMicrosoft社が発表したプログラミング言語。C++ ベースではあるが、Java と同様、コンパイル結果はIL (Immediate Language) と呼ばれるバイトコードであり、.NET プラットホームの実行環境仮想マシン (CLR -- Common Language Runtime) 上で動作させる仕組みになっている。Inprise社 (旧 Borland) でTurbo Pascal, Delphi を開発した、Anders Hejlsberg によって設計された。# には+が4つ集まっている意味 (C++) と、C管程を半音シャープした、という意味がある。
C++	1985年に AT&T社の Bjarne Stroustrup が開発した汎用のプログラミング言語。C では制約されているデータ抽象、オブジェクト指向プログラミング、ジェネリックプログラミングなどの複数のプログラミングパラダイムをサポートしている。当初は C を出力するプリプロセッサとして開発された。C++ の言語仕様は C とほぼ上位互換であり、C からの移行が容易である。1998年には、STL を含む豊富な機能を標準ライブラリとして取り入れ、ANSI/ISO 標準となった。
Chain of Responsibility [Pattern]	複数のオブジェクトをチェーン状に繋ぎ、それぞれに要求を処理する機会を与えるデザインパターン。イベント処理に利用される例がある。

# C

cohesion	コヒージョン（凝集度）、ソフトウェアモジュール内の機能が協調している度合い。ソフトウェアのモジュールは、モジュール内の凝集度が高い方がよい。構造化設計で現われた概念であるが、オブジェクト指向設計においても重要である。
Command[Pattern]	要求をオブジェクトとしてカプセル化するデザインパターン。
Composite[Pattern]	木構造のオブジェクトを全体-部分の関係で表現するデザインパターン。
context	コンテキスト。現在処理中の状態。OS においては、コンテキストが切り替わることをコンテキストスイッチという。ソフトウェアパターンにおいての文脈や背景。
copy and paste programming (コピー&ペースト・プログラミン グ)	既存のプログラムで利用・応用できるものをコピー＆ペーストして新しいプログラムを作成すること。再利用の原始形だが、悪しきプロジェクトの兆候。共通部分をクラスやメソッドとして抽出したり、template method パターンを利用したりして、回避すべき。
CORBA (Common Object Request Broker Architecture)	OMG (Object Management Group) が規定した分散オブジェクト技術の仕様。異機種分散環境上のコンポーネント間でメッセージを交換するためのソフトウェア (ORB と呼ばれる) の仕様を定めている。代表的なものは IBM 社の SOM (System Object Model) や DSOM (Distributed System Object Model)。Java においては、日本で開発された HORB が有名。
coupling	結合度 (カプリング)。ソフトウェアモジュール間の依存度合い。ソフトウェアのモジュールは、他のモジュールとの結合度が低い方がよい。構造化設計で現われた概念であるが、オブジェクト指向設計においても重要である。
CVS (Concurrent Version System)	ソースファイルの管理作業を自動化するバージョン管理ツール。過去ファイルの参照、更新履歴の管理、複数人で作業する際の排他アクセスなどの機能がある。他人が変更中のファイルも変更でき、変更の衝突はマージによって解決するため、多サイトでの開発にも適している。RCS (Revision Control System) をバックエンドとして使用しており、オープンソースとして提供されている。特に、オープンソース形態のソフトウェア開発では必須となるツールであり、Linux、Apache 等のソースコードも CVS で管理されている。Web バージョン (webcvs) や Windows GUI バージョン (WinCVS) もある。

# D

DbC (Design by Contract)	Bertland Meyer により提唱された、「契約による設計」。クラスやメソッドの仕様 (サービス) を「契約」と捉え、「利用者がこれを満たせば私はこれを提供する」という形態でインタフェースを設計する手法。クラスには不変条件 (invariant)、メソッドには事前条件 (precondition)、事後条件 (postcondition) という契約がある。言語 Eiffel では、契約が守られているかどうかをチェックするコードをコンパイラが挿入することができ、ランタイムに契約のチェックを行うことができる。また、継承のセマンティクスを強化することができるため、Liskov's Substitution Principle を型やシングニチャ以上にチェックできる。C/C++ 言語では、標準 assert マクロによってある程度の代用が可能だが、継承セマンティクスやドキュメントのサポートは得られない。Java では、iContract というプリプロセッサ製品によってサポート可能。
Decorator[Pattern]	動的にオブジェクトに修飾機能を追加できるデザインパターン。java.io.OutputStream のように、FileOutput、BufferedOutput など様々な機能を必要に応じてオブジェクトに追加していく設計で利用される。継承を用いる機能修飾に比べて、ランタイムに変更が可能であることから、より柔軟な設計が可能。
Delphi	1991年、Inprise社が開発したプログラミング言語または開発環境のこと。Pascal にオブジェクト指向の考えを取り入れた Object Pascal をベースにした手続き型言語。データベースアプリケーションソフトの構築、プロトタイピングに使われることが多く、RAD (Rapid Application Development) に適している。
Demeter's Law	「デメテルの法則」あるオブジェクトが他のオブジェクトと通信する場合、そのオブジェクトは次に限る。  1. それ自身 2. そのパートナー 3. それが作成したり、保守したりするオブジェクト 4. そのオブジェクトが保持するオブジェクト

# D

Design Patterns	GoF (Gang of Four) によって著された「デザインパターン」を代表とする、設計パターン。GoF の著書には、23 のデザインパターンがカタログ化されている。
DOM (Document Object Model)	W3C が勧告する、Web ページの内容 (文章、画像、音声データなど) およびそれらの配置、スタイルをオブジェクトとして扱い、JavaScript などのスクリプト言語を用いて制御するための API 仕様。DOM に従って Web ページを記述するための言語が Dynamic HTML である。DOM によって、以下の3つの機能が実現される。  1. Dynamic Content 2. Dynamic Styles 3. Absolute Positioning
DRY (Don't Repeat Yourself)	ソースコードで、同じことを2箇所以上で繰り返してはならない、という法則。XP では、Once and Only Once と表現される。この兆候がでたらリファクタリングを行う。例えば、同じ処理手順が繰り返された場合はメソッドとして括り出す (Extract Method)、集合の要素を辿る処理が繰り返されたら Iterator を使う、タイプによる分岐が繰り返されたら継承によるポリモーフィズムを使う (Replace Type Code with Subclasses)、同じ null チェックが繰り返されたら Null Object を導入する (Introduce Null Object) など。

# E

Edelman's Law	「知らない人と話してはいけない」 (Don't talk to strangers) という法則。オブジェクトがメッセージを他のオブジェクトに送信する時に、オブジェクト同士の依存度、結合度を最小限にするための方針。
EJB (Enterprise Java Beans)	Java でプログラム部品を作成し、それらを組み合わせてアプリケーションソフトを構築するための Java Beans 仕様。ネットワーク分散型ビジネスアプリケーションのサーバ側の処理に必要な機能を追加したもの。サーバアプリケーション構築、特にビジネスロジックの記述に使われる。プラットフォームに依存しないアプリケーション構築が実現できる。EJB は、EJB コンテナとよばれる共通バス上に配置される。
Emacs	UNIX 環境で最も広く普及しているテキストエディタ。最初のバージョンは Richard Stallman によって作られた。後に GNU プロジェクトの一部となり、現在では、単に Emacs といえば、GNU Emacs のことを指す。Emacs Lisp と呼ばれる言語処理系を内蔵しており、Lisp を使って機能を拡張することができる。Java の開発者 James Gosling も Gosling Emacs と呼ばれる実装を書いているし、James O. Coplien も、最初に書いたプログラムは Emacs だったという。
ENIAC (Electronic Numerical Integrator And Computer)	1946年にペンシルバニア大学の John Presper Eckert, John Mauchly らによって開発された世界初の電子式コンピュータ。約18000本の真空管を使用し、1秒間に5000回の整数演算が可能だった。
E-R モデル (Entity-Relationship Model)	構造を分析したいデータを、実体 (エンティティ: entity)、関係 (リレーションシップ: relationship)、属性 (アトリビュート: attribute) という概念を用いて表した図式。データベースを設計するためのモデルとして普及している。E-R モデルは、実体を四角形、関係をひし形で表し、実体の属性を記入する。

# E

event-driven イベント駆動。ユーザや OS などから入出力などの要求が発生した時点で実際の処理を呼び出すプログラムの構造。この構造を用いたプログラミングを、イベント駆動型プログラミングという。Windows や Mac OS, X window などの GUI環境で動作するアプリケーションはこの構造を用いて開発されている。Java AWT のイベントモデルもイベント駆動型であり、イベントソースにて発生したイベントを、そのイベントに関心があるリスナーにMultiCast する。

# F

Facade[Pattern] 複雑なサブシステムを容易に扱うための統一インタフェースを与えるデザインパターン。Facade は「ファサード」と読み、建物の正面を意味するフランス語。

Factory Method[Pattern] オブジェクト生成時のインタフェースのみ規定し、実際のインスタンス化をサブクラスによって決めるようにすることで、生成をポリモーフィックに行うことができるデザインパターン。

Flyweight[Pattern] 多数の細かいオブジェクトを、効率よく扱うデザインパターン。

Force フォース。パターンにおいて、問題をその解決に導いた圧力。違うフォースによっては違う解決に導かれる。

FSM (Finite State Machine) 有限状態機械。有限の状態を持つ仮想マシンを実装したもの。あるいはそのモデル。与えるイベントに応じて、現在状態から次状態へと遷移する。通信、制御などほとんどのソフトウェア分野で利用される重要な概念である。UML では状態図で状態遷移を記述する。また、Strategy パターンよりも柔軟な設計をしたい場合に用いられることも多い。

# G

GCC (GNU Compiler Collection) GNUプロジェクトが開発した C, C++, Objective C, Java の複数の言語をコンパイルできるコンパイラ。ほとんどの UNIX系 OSに移植されている。当初は、C および C++ をサポートしていたため「GNU project C and C++ Compiler」と呼ばれていたが、複数の言語をサポートするようになり、「GNU Compiler Collection」に名称変更された。

Generative Programming ソフトウェアシステムファミリーを、ドメインの要求仕様からプログラムの再利用可能な実装コンポーネントを組み合わせることによって自動生成しようという新たなプログラミングパラダイム。問題空間をソリューション空間にマッピングする際に、構成知識を用いる。C++ の総称性 (template) とコンパイラの部分評価機能をヘビーに用いた Generic Programming, AOP, ドメインエンジニアリング, Intensional Programming などのトピックを内包する。

Generic Programming C++ の言語機能である template に代表される、総称性 (パラメタライズドタイプ) を用いたプログラミングパラダイム。コンテナの設計 (STL 等) で用いられる他、階乗計算なども、template, enum, template の特殊化、の組み合わせでコンパイル時評価が可能。

global variable グローバル変数。特定の関数の中だけでなく、プログラム全体で有効な変数。大域変数ともいう。Singleton デザインパターンの別名。

GNU (GNU's Not Unix) FSF (Free Software Foundation) が推進する、UNIX互換ソフトウェア群の開発プロジェクトの総称。フリーソフトウェアの理念に従った修正・再配布自由な UNIX互換システムの構築を目的としている。GNUの成果として、高機能エディタ環境の GNU Emacs やコンパイラのgcc などがある。Linux のほとんどのディストリビューションは GNU ツール群を含んでおり、Richard Stallman は Linux を GNU Linux と呼ぶべきだと主張している。

# G

GoF (Gang of Four)	書籍「デザインパターン」を著した4人組。Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides。4人は, OOPSLA '99 において, 裁判にかけられた。罪状は「Singleton という名前でグローバル変数を正当化した罪」。
GRASP[Pattern]	オブジェクトへの責務割り当てに関する基本原則をパターンの形式で記述したもの。General (汎用) Responsibility (責務) Assignment (割り当て) Software (ソフトウェア) Patterns (パターン)を表す頭字語。

# H

HAL	Stanley Kubric の1968年の映画, 「2001年宇宙の旅」に登場する, マザーコンピュータの名前。IBM をアルファベットで1つずつ進めたネーミングとなっている。
Hollywood's Law	「ハリウッドの法則」。フレームワークとアプリケーションの関係を表現する法則。「Don't call me, we call you.」すなわち, アプリケーションはフレームワーク内のホットスポットをオーバーライドすることにより, コールバックされる。アプリケーションがメインループを持ち, ライブラリをコールする形式との対比を表現している。ハリウッドでは, 脚本や役者が採用されるかどうかは, 高慢なプロデューサにより決定されるため, 持ち来んだ企画やオーディションの結果は, 採用されたものにだけ電話によりコールバックされる。自分で問い合わせはいけない, ということから。
HTML (Hyper Text Markup Language)	Webページを記述するための言語。文書の論理構造を記述したり, 文書の見栄えを記述したりするのに使う。また, 文書の中に画像や音声, 動画, 他の文書の位置などを埋め込むこともできる。もともとHTMLは SGML (Standard Generated Markup Language) のサブセットとして策定されたものだが, 現在は独自に拡張が施され, W3Cによって標準化が行われている。HTMLはタグとテキストによって構成され, このHTMLで記述された文書を開

# I

ICOT (Institute for new generation Computer Technology)	新世代コンピュータ技術開発機構。通産省が推進していた「第5世代コンピュータ開発プロジェクト」の中核組織。並列推進処理の開発などの成果を残したが, 1998年に解散。現在は先端情報技術研究所 (AITEC) によって, 第5世代コンピュータに関する技術の普及が図られている。
IDE (Integrated Development Environment)	総合開発環境。ソースコードを記述するエディタ, それらをコンパイルするコンパイラ, リンカなど, ソフトウェア開発の一連の作業を統合的に行えるようにした開発環境のことを総称して IDEと呼ぶ。Inprise社の Delphi, JBuilder, Microsoft社の Visual C++/J++ などがこれに相当する。別名 Idiot Developer's Environment。これを使うと一向にプログラミングの腕が上達しない。
idiom	言語に依存する慣用表現。粒度が小さいパターンとして分類される。C++ においてポインタをカプセル化する Handle/Body など。
inheritance	新しいクラスの定義に, 上位となる親クラスの機能を受け継ぐこと。java では extends キーワードによって継承するクラスを指定する。
inherited class	派生クラス。あるクラスを継承して作成したクラス。C++ では derived class (派生クラス), java では subclass (サブクラス) という。逆に, 継承元のクラスを C++ では base class (基底クラス), Java では superclass (スーパークラス) という。
Intensional Programming	Microsoft の C.Simonyi によって提唱された新たなプログラミングパラダイム。ASCII テキストのソースコードのみならず, active source (アクティブソース) と呼ばれるグラフデータ等を用いてプログラムを表現する。
Interpreter[Pattern]	言語の文法と解釈インタプリタと一緒に定義するデザインパターン。
Iterator[Pattern]	集約オブジェクトの内部表現を公開せずに, 要素に順にアクセスする方法を与えるデザインパターン。java.util.Iterator や java.util Enumeration が相当する。



J2EE (Java 2 Enterprise Edition)

Webベースの本格的企業システム構築やインターネット利用の電子商取引システムなどへの適用を狙ったサーバ側 Java API群を統合したもの。EJB (Enterprise Java Beans) , Servlet , JSP (Java Server Pages) が技術の核となり、さらにそのバックエンドで JDBC , JTS , JMS , JNDIなどの API群が動作する。

J2ME (Java 2 Micro Edition)

モバイル機器や組み込みデバイスに適した Javaプラットフォーム。J2MEの核となる共通部分は、KVM (K Virtual Machine) と呼ばれる小型 Java仮想マシンと、Embedded Java相当の最小限のクラスライブラリである。携帯電話向けの MIDP (Mobile Information Device Profile) やスクリーンフォン向けの Java Phone API , 双方向デジタルTV向けの Java TV API , 自動車向けの Auto APIなどの特定用途向け API群からなる。

J2SE (Java 2 Standard Edition)

デスクトップ環境 (クライアント) を想定した Javaプラットフォームであり、従来 JDK やJava2と呼んでいたもの。Java の基本環境。

Java

1995年に James Gosling らによって開発されたオブジェクト指向型プログラミング言語。強力なセキュリティ機構や豊富なネットワーク関連の機能が標準で搭載されており、ネットワーク環境で利用されることを強く意識した仕様になっている。Java で記述したソースコードは Javaコンパイラでコンパイルする。Javaコンパイラはバイトコードと呼ばれる中間コードを生成する。Java によるプログラムはこのバイトコードの状態に配布される。このバイトコードを Java仮想マシンと呼ばれるソフトウェアによって、そのプラットフォームで実行可能なネイティブコードに変換し、実行する。Javaで作成されたソフトウェアは Java仮想マシン上で動作するため、基本的に Java仮想マシンが動作する環境であればどのようなプラットフォームでも動作する。従来、JDK と呼ばれたものが、1998年の JDK 1.2 リリースの際に Java 2 という名前になった。前身は oak と呼ばれる家電への組み込みターゲットにした言語であったが、インターネットの普及の波に乗って20世紀最後で最強の言語となった。

Javaアプレット (Java Applet)

Java で記述されたクライアントで実行されるプログラム。単にアプレットともいう。WWW を使って配布でき、Java に対応した WWWブラウザで実行される。ダウンロードした Javaアプレットは基本的にユーザ側のハードディスクの内容を読み書きしたり、自分が呼び出された Webサーバ以外のコンピュータに接続したり、他のアプリケーションソフトを起動したりすることはできないよう、セキュリティが設定できる。java.applet.Applet クラスを継承して作成する。

Java仮想マシン (Java Virtual Machine)

Javaコンパイラによって生成されたバイトコードをそのプラットフォームのネイティブコードに変換し、実行するソフトウェア。Java仮想マシンを実装しているコンピュータであれば、そのプラットフォームに関係なく同じバイトコードを変更なしで実行できる。Javaのソースコードを書き直したり、再コンパイルする必要はない。Java仮想マシンの実現方式として、JITコンパイラ方式とインタプリタ方式、HotSpot 方式がある。また、Javaのバイトコードを解釈し実行できるマイクロプロセッサ、Javaチップもある。

JDK (Java Developer's Kit)

Sun Microsystems社が無償で提供している、Javaソフト開発ツール。コンパイラやデバッガ、Applet Viewer , Java仮想マシンなどが含まれる。

JG

Java への総称性導入プロジェクト。前身は Pizza 。

# J

## Jini

Sun Microsystems社の Bill Joy によって開発された。Java をベースにして互換性のない様々な機器をネットワークに接続する技術の総称。Java仮想マシンを実装していれば、カーナビゲーションやAV機器、家電製品までもをネットワークに接続することができる。アラジンの魔法のランプに出てくる「ジニー」という魔法使いの精霊から名付けられたといわれている。

## JITコンパイラ (Just-In-Time compiler)

Javaプログラムの中間コードを各種類のネイティブコードに変換するコンパイラ。Java仮想マシンの実現方式の1つ。Javaインタプリタ方式の10倍~20倍程度の性能が得られる。

# K

## KISS

"Keep It Simple and Stupid" もしくは "Keep It Simple and Small"。「KISS の法則」。シンプルに設計することを最も重要視せよ、という指針。

## KJ法 (Kawakita Jiro Method)

収集した多量の情報を効率よく整理するための手法、考案者の川喜田二郎の頭文字から命名された。ブレンストーミング等で用いられることが多く、収集されたカードを同じ系統のものでグループ化することで情報の整理・分析を行う。

# L

## Liskov's Substitution Principle

サブクラスのインスタンスは、スーパークラスのインスタンスが利用可能な場所では、いつでも利用可能であるべきである、という規則。継承の意味的な制約を表す。コンパイラ型システムの中でチェックすると同時に、プログラマがこの原則に基づいたセマンティクスを守る必要がある。

# L

## Linux

1991年、フィンランドのヘルシンキ大学の学生だった Linus B.Torvalds によって開発されたオープンソースの UNIX互換 OS。ソース公開を義務づけ、再配布と改変の自由を保証した GPL (GNU General Public Licence) と呼ばれるライセンスに沿って配布されている。開発当初は Intel社の x86を搭載したコンピュータでしか動作しなかったが、フリーソフトウェアとして公開され、全世界のボランティアの開発者によって改良を重ねられ、Alpha, SPARC, PowerPC など他のプラットフォームに移植された。なお、Linuxとして開発されているのは、主にカーネルなどの基本的な部分のみを指すため、シェルや各種コマンドなどのユーザ環境は、Internetで公開されているフリーソフトウェアが使われている。そのため、Linuxカーネルと各種コマンド、エディタ、コンパイラ、ウィンドウシステムなどをまとめたパッケージが、複数のディストリビュータからリリースされている。主な Linuxディストリビューションパッケージとして、Red Hat, Slackware, Debian, TurboLinux, Vine, S.U.S.E などがある。

## Lisa

1983年に Apple社が発売したパーソナルコンピュータ。世界初の GUIベースのパーソナルコンピュータとして、当時としては革新的なものでありながら、非常に高価だったために商業的には成功せず、基本思想のみが後の Macintoshに受け継がれた。

## LISP (LISt Processor)

1962年にマサチューセッツ工科大学の John McCarthy によって開発された関数型プログラミング言語。Prolog と並ぶ人工知能分野の基本言語で、リストと呼ばれるデータ操作命令の並びを処理するリスト処理や型宣言不要などの特徴がある。1984年に LISPの標準仕様である Common Lisp が策定された。「Lisp プログラムはメモリを気にしない。Cプログラムはメモリのことしか気にしない」というジョークがある。

# M

## Mach (Multiple Asynchronously Communicating Hosts)

1987年にカーネギーメロン大学が米国防総省の資金援助を受けて開発したマルチプロセッサ対応の OS。これまでの改良で肥大化した UNIXのカーネルの機能を必要最小限に絞り込んで移植性を高めたマイクロカーネルを備える。マイクロカーネルから機種依存の部分を極力排除してハードウェアから独立させたことが特徴。従来の UNIXとの互換性も高い。



Mac OS X

Apple社が 2001年に出荷を予定している、Macintosh用の新しい OS。従来の Mac OS と新 OSの Rhapsodyを統合するもの。Rhapsodyでアナウンスされていた Yellow Boxや Blue Box, Java, メモリ保護やプリエンティブマルチタスクといった先進の機能を提供する一方で、従来の Mac OSシリーズの主要APIをまとめた Carbon APIを移植し、これまでのアプリケーション資産を移植しやすくしている。全体は 4層から構成され、最下層に Machカーネルと BSDから成る Darwin, 第 2層にグラフィック機能として、2Dを受け持つ Quartz, 3Dを描画する OpenGL, メディアを統合する QuickTime, さらに第 3層には API群として、Mac OS 9 実行環境の Classic, ネイティブ環境の Carbon, 開発環境の Cocoa, 最上位に GUIである Aquaが位置する。なお、Rhapsodyは Mac OS X Serverに改名されており、サーバ用 OSとして位置づけられ、Mac OS Xはクライアント用 OSとなる。

microkernel

マイクロカーネル。OSの基本的な機能を提供するカーネルを、必要最小限の機能だけ残して小型化したもの。従来のカーネルから、ファイル管理、仮想記憶管理、ネットワーク管理などの機能を独立させてサブシステムとして実装し直し、マイクロカーネルには、メモリ管理やプロセス、スレッド管理などの機能だけを持たせた。マイクロカーネルを採用した OSの最大の利点は、各種プロセッサに容易に移植できることである。ただし、OSをマイクロカーネル化すると、サブシステム間での呼び出しや、カーネルモードとユーザモード間の遷移のオーバーヘッドなどのために、従来の一体型のカーネルに比べると若干パフォーマンスが低下する。マイクロカーネル技術を使った OSとして、Mach や Windows NT などが有名である。

Mediator[Pattern]

オブジェクト群の相互依存を減らすために、仲介役オブジェクトを導入するデザインパターン。

Microkernel [Pattern]

システムの核となる最小限の機能を、拡張機能や顧客依存部分から分離するアーキテクチャパターン。

Memento[Pattern]

オブジェクトの内部状態を不透明なまま外部に保存し、後で状態を復元できるようにするデザインパターン。

MINIX

オランダのライデン大学の Andrew S.Tanenbaum が、16ビットマイコン上で UNIXが学習できるように開発した UNIX互換の教育向け OS。Linux が最初の開発土台にしたが、newsgroup 上で Tanenbaum が「Linux is Obsolete」という記事を書いたことで Linus Torvalds や Ken Thompson も参加した「モノリシックなOSは終わった」論争が起った。

meta language

メタ言語。プログラミング言語や自然言語を定義するための言語。言語記述言語、超言語ともいう。代表的なメタ言語の記法に BN記法(バックス記法)があり、現在の多くのプログラミング言語の仕様記述に採用されている。

Mozilla

Netscape Communications社が自社の Webブラウザである Netscape Communicator 4.0のソースコードを公開したことを受けて、同社が設けたオープンソース開発プロジェクトMozilla.org で開発されている Webブラウザの開発コード名、またはそのプロジェクトの名称。

Metaphor

比喻。XPでは、どの様に全体のシステムが機能するかを示すシンプルな喩え話(メタファ)をメンバが共有することで全ての開発を導く(ガイドする)。

MultiCast

情報をオブジェクトにカプセル化し、型安全に転送するデザインパターン。GoF には独立したパターンとして掲載されていないが、John Vlissides は別個のパターンとして書籍「パターンハッチング」で取り上げた。Observer パターンとの差異として「型安全性」のみを取りだし、TypedMessage パターンと呼ぶこともある。JDK1.2 のイベントモデルが典型例。なお、その後 A.H.Eden らは LePUS というパターンの形式表現を用いることにより、Observer と MultiCast の差異は Typed Message 以上のものであることを示した。

# M

multi-inheritance	多重継承。オブジェクト指向プログラミング言語で、複数の基底クラスから継承を行うこと。C++ では多重継承が許されているが、Javaでは許されていない。
Multi-Paradigm Design	オブジェクト指向を含む複数の設計パラダイムを組み合わせて、より自然な設計を生み出す設計論。James O. Coplien によって C++上に提示された。問題領域と言語領域を共に共通性と可変性によって分析し、そのマッピングを行う。
multi-threading	マルチスレッド。1つのプロセスを複数のスレッドに分けて並行処理すること。ライトウェイトプロセスとも呼ばれる。Java では、仮想マシン上でのマルチスレディングをサポートしているが、それがOS のマルチスレッドにどのようにマッピングされるかは仮想マシンの実装・設定依存である。RTOS では、マルチタスキングと呼ばれることが多い。
MVC ( Model-View-Controller ) [Pattern]	MVCアーキテクチャパターンは、対話型アプリケーションを3つのコンポーネントに分割する。モデルコンポーネントは、アプリケーションの中核機能とデータを含むコンポーネントである。ビューコンポーネントは、情報をユーザに表示するコンポーネントである。そして、コントローラは、ユーザの入力を取り扱うコンポーネントである。ユーザインタフェースを実現するのは、ビューコンポーネントとコントローラコンポーネントである。更新伝播のメカニズムによって、ユーザインタフェースとそのモデルとの一貫性を保証することができると共に、モデルがビューに依存することを防ぐ。PAC アーキテクチャパターンと対比されることが多い。UMLにおける《entity》、《boundary》、《control》ステレオタイプと対応する。

# N

NIH ( Not Invented Here ) Syndrome	「ここで開発されたものではない」症候群。如何に優れたものであると、独自開発の技術でないからという理由で採用しない、という姿勢。
------------------------------------	---

# O

Observer[Pattern]	オブジェクトの状態変化を、それに依存する全オブジェクトに自動的に通知するためのデザインパターン。POSA では Publisher/Subscriber パターンという。GoFの1人、John Vlissidesは、著作 Pattern Hatching の中で、型安全な Observer パターンとしてMultiCast パターンを提示した。
Occam's Razor	物事を説明する 2つの方法がある場合、単純な方が正しい。オッカムのカミソリ。
OCP ( Open-Close Principle )	あるモジュールは、出荷のために Close されなければならない。しかし、拡張のためにOpen でなければならない。という原則。これを両立するために、継承を用いたクラスの拡張がある。すなわち、モジュールのソースコードを変更するのではなく、継承を用いて拡張せよ、という指針を与えている。Bertland Meyer によって提唱された。
OLE ( Object Linking and Embedding )	Windows用として Microsoft社が開発した機能で、異なるアプリケーション・プログラム間で、データの連携と埋め込みを可能にする機能の名称。OLEを利用することにより、あるアプリケーションで作成した情報を別のアプリケーションに取り込んだり、別のアプリケーションの機能を自身の機能の一部であるかのように利用できる。OLEはサーバアプリケーションとクライアントアプリケーションから構成される。サーバアプリケーションは単独で動作する OLEサーバと、部品の1つである OLEコントロール ( OCX ) に分類される。クライアントアプリケーションは コンテナと呼ばれる。1996年に Microsoft社は OLEコントロールを ActiveXコントロールと改名した。
OMG ( Object Management Group )	1989年にオブジェクト指向技術の普及と標準化を図るために設立された団体。600社以上が加盟している。OMA ( Object Management Architecture ) や CORBA を定めている。



OMT ( Object Modeling Technique )

オブジェクト指向を使ってシステム分析や設計を進めるオブジェクト指向分析/設計 ( OOA/OOD ) 開発方法論の1つ . James Rumbaugh によって開発された . OMT法では , システムを , オブジェクトモデル , 動的モデル , 機能モデルという 3つのモデルを使って表現する . 使用するモデルがそれほど複雑でない , 構造化分析/設計 ( SA/SD ) 手法からの移行も容易 , システム分析や設計手順が詳細に規定されているなどの利点を持つ .

OODB ( Object Oriented DataBase )

データと手続きを一体化した単位で扱うオブジェクト指向の考え方を応用したデータベースの総称 . 継承階層と集約階層でデータを管理し , データとそのデータに対しての処理をオブジェクトとしてまとめて扱うことができる . また , オブジェクト指向言語とのマッピングがしやすく , 複雑で大量のデータを扱わなければならないデータベースや , オブジェクト指向言語により永続性を得るソフトウェア技術として注目されている .

Open Source

ソースコードをインターネットなどを通じて無償で公開し , 誰でもそのソフトウェアの改良 , 再配布が行えるようにすること . オープンソースの考え方は , ソースコードを公開して有用な技術を共有することで , 世界中の誰もが自由にソフトウェアの開発に参加することができ , その方がずっと素晴らしいソフトウェアが生まれるはずだという思想に基づいている . 主なものに , WWWブラウザである Netscape Communicator や UNIX互換 OSである Linux などがある . Eric Ramond の論文「伽藍とバザール」の中で衝撃的に紹介された .

operator

演算子 . C++等では , ユーザ定義型に対してオペレータを定義できる . この機能をオペレータオーバーロードという .

ORB ( Object Request Broker )

分散オブジェクト環境で , オブジェクト間のデータや処理要求などのメッセージをやりとりする際に用いられる仲介ソフトウェア . OMG が CORBA の一環として標準仕様をまとめている . クライアントが受けたいサービスの名前と要求内容を ORBに渡す . ORBはその名前と要求内容からそのサービスを提供する側を探し , 要求内容を伝える . 提供側はそのサービスを実行し , その結果を ORBに戻し , ORBはそれをクライアントに伝えるといった手順 .



overload

多重定義 . プログラミング言語において , 名前が同じで引数の型の異なる関数を利用することができる機能のこと . 呼び出し側の引数の型に応じて , 適切な関数がコンパイル時に選択される .

override

基底クラスのメソッドを派生クラスで書き換えて再定義すること . C++やC#では , 基底クラスのメソッドが virtual であることが要求される . javaでは , 明示的に final と指定していないメソッドは自動的に override 可能である . どのクラスのメソッドが呼ばれるかは , 実行時のオブジェクト型によって選択される .



PAC ( Presentation-Abstraction-Control ) [Pattern]

PACアーキテクチャパターンは , 協調するエージェントの階層という形で , 対話型ソフトウェアシステムのための構造を定義する . このエージェントはその 1つずつがアプリケーション機能の特定の一面を実現する責務を負っており , 3つの部分から構成される . 3つの部分とは , Presentationコンポーネント , Abstractionコンポーネント , Controlコンポーネントである . これらのコンポーネントへ分割することにより , エージェントとの通信から , 人間とコンピュータの対話機能が分離される . PAC は再帰的 , 階層的に拡張することができる . MVC パターンと対比されることが多い .

Pascal

1971年に Niklaus Wirth によって開発された手続き型言語 . 構造化プログラミングが可能 , 再帰アルゴリズムが記述できる , 自由にデータの型を定義する機能を持つ , などの特徴を持つ . 開発用というよりも教育用 , 研究用などの用途が多かったが , Borland 社の Turbo Pascal , その後の Delphi によるオブジェクト指向拡張により , RAD 用語として注目を浴びた .

Peopleware

T.Delmarco とT.Lister の書籍「Peopleware:Productive Projects and Teams」 . ソフトウェアの開発に , 人の問題を避けて通ることはできない . Delmarcoは近年XPに骨入れしており , 「Planning Extreme Programming」に序文を寄せている .

# P

Perl	1987年に Larry Wall によって開発されたインタプリタ言語。awk や sed の機能を拡張したもので、非常に強力なテキスト処理、ファイル処理機能を備えている。さらに、C の特徴も取り入れられ、従来は C でしか記述できなかったような処理も Perl で手軽に記述できるようになり、汎用プログラミング言語としても広く利用されている。インターネット普及後は、CGI を利用したプログラムを制作する際に利用される言語のひとつとして知られるようになった。当初、UNIX上でしか動作しなかったが、現在では Windows や Mac OS にも移植されている。シンタクスが記号的で、注意しないと自分で書いたプログラムでもすぐに理解不能になる。その面で毛嫌いするプログラマも多い。
Polymorphism	多態性・多様性・多相性。継承された各メソッドは、継承元のそれと振舞の意味は同一であるものの、その外部に対する動作に異なる特徴を与えることを可能とする。
pattern	software pattern
Prolog	1972年にフランスのマルセイユ大学の Alain Colmerauer によって開発されたプログラミング言語。述語論理を応用した論理型言語と呼ばれる非手続き型言語で、推論機構を簡潔に記述できることから、エキスパートシステムなどの人工知能分野で広く利用されている。登録した論理に対する自動バターン照合機能、および自動バックトラック機能で、三段論法的な演算をするのが特徴。日本における第5世代コンピュータプロジェクトの言語として採用された。
Prototype[Pattern]	オブジェクトの原型 (prototype) をコピーすることで、オブジェクト生成をガリモーフィックに行うデザインパターン。
Proxy[Pattern]	あるオブジェクトへのアクセスを、同じインタフェースを持つ代理を通じて行うデザインパターン。

# Q

QWAN (Quality Without A Name)	無名の質。建築家 C. Alexander の Timeless Way of Building で、美が表出する場に存在するとした、名付けることができない質。
-------------------------------	--

# R

RCS (Revision Control System)	ファイルの改版管理。版の蓄積、取り出し、更新記録、判断、複数の版の合成といった作業を自動化することができる。CVS のバックエンド。
Refactoring	リファクタリング。機能を変えずにプログラムの内部構造を再設計する手法。XPの1つのプラクティスになっている。
ROPES (Rapid Object-Oriented Process for Embedded Systems)	Bruce P. Douglassが提唱する、リアルタイムのオブジェクト指向方法論。
RTOS (Real-Time Operating System)	Vxworks, QNX, ITRON など、実時間制約を満たすことができるOS。
Ruby	1995年に まつもとゆきひろ によって開発されたオブジェクト指向スクリプト言語。強力な文字列処理能力に加えて、例外処理、イテレータ、クラスといったオブジェクト指向的な機能も備える。移植性が高く、UNIXだけでなく、Windows, Mac OS 上でも動作する。XP コミュニティで最も支持され、米国でも人気が出ている。
RUP (Rational Unified Process)	UP

## S

script language	機械語への変換作業を省略して簡単に実行できるようにした簡易プログラムをスクリプトといい、そのスクリプトを記述するための言語をスクリプト言語という。主なスクリプト言語として、MS-DOS のバッチファイル、UNIX の sed, awk, Perl や Macintosh の Applescript、最近では、Ruby、Web ページ作成用の JavaScript、VBScript などがある。
Separation of Concerns	Dijkstra の「関心の分離」。
Singleton[Pattern]	あるクラスに対してインスタンスが1つしかないことを保証すると同時に、そのインスタンスへのグローバルなアクセスを与えるデザインパターン。
Smalltalk	1972年に XEROX社のパロアルト研究所で Alan C.Kay によって開発された世界初の本格かつ純粋オブジェクト指向プログラミング言語。ダイナミック構想の一環として Alto上で開発された。品質のよいクラスライブラリが豊富であり、開発効率が高い。強力な GUI 開発環境を持つ。その後、1997年のOOPSLAで Squeak として再登場した。
software pattern	ある状況下での典型的な問題に対して、洗練された解決方法を示したもの。
SSL (Secure Sockets Layer)	1994年に Netscape Communications社によって開発されたセキュリティ機能の追加されたHTTPプロトコル。SSL は TCP層とアプリケーション層間に位置する 2階層からなるプロトコル層で、下位層はデータの配送、圧縮などを担当し、上位層では認証やデジタル署名、暗号化などのネゴシエーションを行う。HTTP に限らず Telnet や FTP、SMTP などのさまざまなアプリケーションプロトコルを暗号化できる。ただし、UDPパケットの暗号化には対応していない。秘密鍵暗号方式、公開鍵暗号方式、デジタル証明書などの技術がある。

## S

STL	Alexander Stepanov が設計した C++によるコレクションライブラリ。テンプレートを利用した型安全性と、時間効率が仕様として表現されている点に特徴がある。C++ 標準ライブラリとして組み込まれている。
State[Pattern]	オブジェクトの内部状態の変化を、別の状態オブジェクトで表現することで、振舞の変化を状態オブジェクトのポリモーフィズムによって記述するデザインパターン。FSM を使った実装が考えられる場面で、可読性と拡張性のバランスがとれた設計解を得ることができる。
Strategy[Pattern]	アルゴリズムの集合を定義し、各アルゴリズムをサブクラスとしてカプセル化して交換可能にするデザインパターン。java.awt.LayoutManager が例。
syntax	構文。命令を記述したものを文、文の規則を構文（シンタックス）という。プログラミング言語の文法や書式のことを表す。
syntax sugar	構文糖。プログラミング言語の設計において、本質的なセマンティクスを変えないことなく、シンタックスレベルで見栄えを整える細工。

## T

Tcl/Tk	Tcl は 1988年に J.K.Ousterhout によってマクロ言語を統一することを目的として開発されたインタプリタ言語。Tk は X Window System を使って GUI を構築するためのライブラリで、Tcl から利用するツールキット。現在では、Tcl と Tk を組み合わせた Tcl/Tk が簡易 X Window や Windows の GUI構築プログラミング環境として広く利用されている。ミニプログラムやプラグインの作成に適している。また、Perl 等の他のスクリプト言語の GUI として、Tk のみ切り出されて利用されることもある。
--------	---

# T

Template Method[Pattern]	1つのメソッドにアルゴリズムのスケルトンを定義し、その中のいくつかのステップについてサブクラスで変更可能にすることで、テンプレートとフックを分離するデザインパターン。
TeX	1977年にスタンフォード大学の Donald E. Knuth によって開発された理系学術文書向け文書処理システム。文書の中に組版情報を埋め込み、それを処理系で印刷可能な形式に変換するというスタイルを取っている。フリーソフトウェアとして無償で公開されており、Windows や Macintosh, UNIX など様々なプラットフォームに移植されているが、標準化が行き届いていないので、どの機種でもまったく同じ出力が得られる。Leslie Lamport による LATEX, Michael Spivak による AmS TEX など、いくつかの機能強化版がある。TeX のバージョンは、3.1, 3.14, 3.141 と進み、Knuth の逝去をもってバージョン にフリーズされる。
The 5th Generation Computer	第 5 世代コンピュータ。ICOT (新世代コンピュータ技術開発機構) が開発を目指した、連想機能や推論機能などを持つコンピュータ。第 1~ 4 世代のノイマン型コンピュータと異なり、非ノイマン型のアーキテクチャで、並列処理が可能。

# U

UML (Unified Modeling Language)	1997年に Rational Software社の Grady Booch, James Rumbaugh, Ivar Jacobson の 3人によって発表された、オブジェクト指向分析、設計においてシステムをモデル化する際の記法 (図法) を規定した言語 (ビジュアル・ランゲージ)。UML を使うことで、ソフトウェア集約的なシステムの成果物をビジュアル化、仕様化、構築、および文書化することができる。
undo	直前にユーザが実行した操作を取り消して、元の状態に戻すこと。またはその機能。Command/パターンと Memento/パターンを用いてサポートする手法が一般的。

# U

Uniform Access Principle	"モジュールによって提供されるサービスは統一されたノテーションでアクセスできなくてはならない"という Eiffel の設計者 Bertland Meyer によって提唱されたモジュール設計原則。例えば Eiffel では、引数なしの値を返すメソッドと属性へのアクセスは同じ記法であり、クライアントからは区別がつかない。これにより、属性からメソッドへと仕様を変更してもクライアントコードは修正の必要がない。C++ や Java においては、属性へのアクセスをすべて getメソッドによって行うことでこの原則を守ることができる。
Unit Test	ソフトウェアをコンポーネント (クラス) 毎に独立してテストすること。XP やリファクタリングで中心的な役割をはたす。Unit Test をサポートするテストフレームワークとして、JUnit と呼ばれる言語毎のツールがフリーで公開されている。JUnit (Smalltalk), JUnit (Java), NUnit (.NET), JUnitEE (J2EE, EJB, Servlet), PalmUnit (Palm), RubyUnit (Ruby) など。
UP (Unified Process)	UML に準拠し、ソフトウェア開発のライフサイクル全体のプロセス。繰り返し型 (イテラティブ) 開発、リスク管理、ユースケース駆動に特徴がある。多くのプロセスの和集合的な意味をもち、カスタマイズ性に優れている。
UseCase	アクタに対して何らかの結果やサービスを提供するためにシステムが実行する一連の操作のこと。
UseCase Point	ユースケースの数やそれぞれの難易度などをもとにした見積法。Gustav Karner が提唱。
virtual function	仮想関数。C++/C# において、キーワード virtual で宣言されたクラスのメンバ関数。ユーザが仮想関数を呼び出すとき実行されるインプリメンテーションは、仮想関数が呼び出される対象のオブジェクトのクラスに依存する。これは、実行時に判別される。

# V

# V

virtual inheritance	仮想継承。多重継承において、あるクラスの複数の継承元をたどった時に、同じスーパークラスが2度以上現れる場合、それらを1オブジェクト内に1つのみ持つ継承。UMLでは、{overlapping}制約で表現される。継承クラス図がひし形になるため、ダイヤモンド継承ともいう。
Visitor[Pattern]	あるオブジェクト構造上で実行される操作を表現するクラスを導入することで、操作されるオブジェクトのクラスを変更せずに新しい操作を定義できるようにするデザインパターン。ダブルディスパッチを行う。
VM (Virtual Machine)	仮想マシン。コンピュータ上で、ソフトウェア的に仮想のCPUとその命令セットを実現する。Smalltalk, Java, C# などは VM 上で動作する言語である。

# W

Windows	Microsoft社が開発した、Windows を冠する OS の総称。最初のバージョンが 1986年に発売され、1992年に発売された Windows 3.1 で PC/AT互換機用の標準 OSとして世界に広まった。Windows 3.1までは MS-DOSに GUI環境を構築するための拡張ソフトウェアだったが、Windows 3.1の後継として 1995年に発売された Windows 95からは独立した OSとして機能するようになった。1998年に Windows 98, 2000年に Windows Me が発売されている。また、1993年から発売が開始された Windows NTシリーズはネットワークサーバ用途を前提に 0から開発された Windows 3.x/9x とは基本構造が全く異なる OS系列で、純粋な 32ビットOSとして高い安定性と優れたパフォーマンス、高度なセキュリティ機能を備える。2000年には Windows NTシリーズと Windows 9x シリーズを統合した、Windows 2000が発表されている。他に携帯端末用の Windows CE も販売されている。OS がクラッシュした場合でも、Ctrl-Alt-Delete の 3つのキーの組み合わせで再起動できるという特徴がある。
---------	--

# X

XP (eXtreme Programming)	Kent Beck らによって提唱される新しいプログラミング方法論。「ベアプログラミング」、「オンサイト顧客」などの12のプラクティスからなる。
xUnit	Unit Test

# Y

YAGNI (You Ain't Going to Need It)	XP において、後で必要になると予測して余計な複雑性(新たなメソッドやクラス)を導入しようとしたときに与える忠告。「それは必要にはならない!」
------------------------------------	---

あ

アーキテクチャパターン  
アンドゥ Architecture Pattern  
undo

基底クラス base class

き

い

イディオム  
イテレータ  
イベント駆動  
インタプリタ idiom  
Iterator  
event-driven  
Interpreter

グローバル変数 global variable

く

お

オーバーロード  
オーバーライド  
オペレータ overload  
override  
operator

継承 inheritance

け

か

仮想関数  
仮想継承 virtual function  
virtual inheritance

構文糖  
コピー&ペースト・プログラ  
ミング  
コンテキスト syntax sugar  
copy and paste programming  
context

こ

し

シンタクス  
シンタクス・シュガー syntax  
syntax sugar

す

スクリプト言語 script language

マイクロカーネル microkernel  
マルチスレッド multi-threading

そ

ソフトウェアパターン software pattern

メタ言語 meta language

た

第 5 世代コンピュータ The 5th Generation Computer  
多重継承 multi-inheritance  
多重定義 overload  
デザインパターン Design Patterns

ユースケース UseCase  
ユースケースポイント UseCase Point  
ユニットテスト Unit Test

は

派生クラス inherited class  
パターン software pattern  
ビジュアルウェア Peopleware  
バイトコード byte code

リファクタリング Refactoring

ま

め

ゆ

り

# あとがき

この小冊子は、オブジェクト技術者が頻繁に参照するであろう用語、アーキテクチャ、パターン等をコンパクトなハンドブックの形式にまとめたものです。

90年代後半は、UMLの統一、Java<sup>™</sup> 言語の登場、CORBA アーキテクチャの標準化など、ソフトウェア工学がコンピュータ基礎技術やIT産業と歩調を合わせ、オブジェクト指向を中心に進歩した時代といえるでしょう。一方で、プログラミング現場の実践的なプラクティスを収集、洗練し、共有しようというソフトウェアパターンの一大ムーブメントが起こりました。特にXP は、ソフトウェア開発を工学を超えたコミュニケーションとして再認識する視点を、アンチテーゼとして提出したと捉えられるでしょう。

21世紀を向かえ、これらは数年で止揚されるかもしれません。振り返って自分の手もとを見ると、多くの新しい概念が氾濫し、ともするとその中での位置感覚を失ってしまいそうです。

このハンドブックは、オブジェクト技術のリテラシを一望することを目指していますが、充実度はまだ低いレベルです。これを基礎に毎年内容を更新していくつもりです。また、現場で手元に置いて頂けるよう、余白を多く使ってメモを書き込める構成にしています。

日本においてもさらにオブジェクト指向が浸透し、このコミュニティに対して貢献できることを祈って、このハンドブックを贈ります。

2000年クリスマス、福井にて。

關永和システムマネジメント  
オブジェクト倶楽部  
平鍋 健児

info@ObjectClub.esm.co.jp  
<http://ObjectClub.esm.co.jp/>

Memo

Memo

Memo

Memo

Memo

Memo

Memo

Memo

オブジェクト・ハンドブック 2001

2000年12月24日 初版第1刷発行

企画制作： 株式会社 永和システムマネジメント  
オブジェクト倶楽部  
info@ObjectClub.esm.co.jp  
http://ObjectClub.esm.co.jp/

編集：牧野弘嗣

監修：平鍋健児

ブックデザイン：株式会社 baus

製本印刷：福島印刷株式会社 福井営業所

Copyright 2000 Eiwa System Management

[非売品]