プログラムのリバース・エンジニアリングの著作権保護

詹　智玲（せん　ち れい）

目　次

第1章　問題の所在

1．プログラムのリバース・エンジニアリングの意義

(1)　定義および検討対象の限定

リバース・エンジニアリングという言葉は，もともと電気，機械，化学などの技術製品に関連して生まれたもので，このような既存の製品を調査・解析し，その内容・方法・技法などの技術（アイディア）を抽出することを指している。通常のエンジニアリングが，技術から出発して製品の製造へとの過程を辿るのに対し，製品から技術へとの逆の過程を辿ることからリバース・エンジニアリングと呼んでいる。技術の発展は既存の技術を基礎にして行われるものであり，リバース・エンジニアリングは既存の製品の技術を知るための有力な手段の一つであるので，本来の技術保護法である特許法や，半導体チップ保護法では，それぞれリバース・エンジニアリングを許容する旨の規定が置かれている。

コンピュータ・プログラムは産業的に作成されており，産業的に製造される他の技術製品と同様，リバース・エンジニアリングの対象となっている。しかし，コンピュータ・プログラムの保護の基本的な法制度として著作権法が選択され，しかも従来，技術保護法としての性格が与えられていなかった著作権法には，リバース・エンジニアリングに関する規定が欠けていることから，リバース・エンジニアリングの著作権法上の扱いが重要な法律問題となった一方，コンピュータ・プログラムのリバース・エンジニアリング（以下，特に断らない限り，本稿での「リバース・エンジニアリング」とはコンピュータ・プログラムについてのそれをいうこととする。）については、技術上あるいは法律上の確立した定義も存在しなかった。既存のプログラムを調査・解析しその構造や製造方法などの技術を抽出すること，すなわち情報取得行為として捉えられる場合もあれば，その抽出した結果を利用して新しい製品を開発することまで指して用いられることもある。また，リバース・エンジニアリングという用語の曖昧さを避けるため，情報取得行為をリバース・アナリシスと呼び，取得した情報を利用して新たなプログラムを作成する行為をフォワード・プログラミングと呼ぶとの提案もなされている。そして，最も狭い意味では，リバース・エンジニアリングは，ディスアセンブルまたはデコンパイレーションのほぼ同義語として用いられている。

ここでは，検討対象となる行為を明確にするため，リバース・エンジニアリングの意味は情報取得行為に限定する。つまり，以下に検討するのは情報取得行為自体に伴う著作権法上の問題であり，取得された情報を利用して新しいプログラムを開発し頒布することは含まない。また，場合によって，ディスアセンブルまたはデコンパイレーションのほぼ同義語として用いることが多い。

(2)　目的・対象資料・抽出情報・手法

リバース・エンジニアリングを検討するには様々な視点が考えられるが，その一つとして，目的，対象資料，抽出情報及び手法からこれを検討することができる。

まず，リバース・エンジニアリングがどのような目的で行われるのかについては，様々な場合が考えられるが，主として次のとおりである。①著作権侵害の調査・発見。あるプログラムが別のプログラム著作権を侵害しているか否かを調査すること。②プログラムの保守（バクの発見，修正）。プログラムに不都合な動作が見られる場合に，その原因を調べたり，不都合な成分を改善すること。③プログラムの改良，移植。プログラムの機能追加や，あるコンピュータ・システムの上で作動しないプログラムを作動するように修正すること。④プログラムの性能・機能の調査。プログラムの性能がどの程度のものであるか，どのような機能が備えられているかを調査すること。⑤互換プログラムの開発。コンピュータ・システムの構成要素としてのあるプログラムと互換しても同種の機能を果たすことができるようなプログラムを開発すること。⑥接続プログラムの開発。あるプログ

ラムと直接又は回線を介してデータをやりとりできるようなプログラムを開発すること。⑦記憶媒体による情報交換(データの相互利用)。あるコンピュータ・システム（又はプログラム）によってデータを書き込んだ記憶媒体について，別のコンピュータ・システム（又はプログラム）によっても利用できるようにすること。⑧コンバータの開発。あるコンピュータ・システムにおいて使用していたデータやソース・プログラムを別のコンピュータ・システムにおいて使用できるように変換するためのコンバータ（プログラム）を開発すること。

次に，対象資料，つまりリバース・エンジニアリングを行う場合，どのようなものを使うかについては，大別すると，次の通りである。①対象プログラム。つまり，オブジェクト・コード形式のプログラム及びソース・コード形式のプログラムである。②周辺資料。つまり，仕様書・マニュアルなどの関連資料，プログラムの実行結果，およびハードウェアなどである。

また，対象資料からリバース・エンジニアリングによって抽出された情報については，次のリバース・エンジニアリングの対象資料となるという階層的構造をなすことがある。リバース・エンジニアリングの最終的な抽出情報としては次のようなものが考えられる。①ソース・コード。リバース・エンジニアリングによってオリジナルのソース・コードを知ることは困難であり，通常は擬似的なソース・コードを知るのに止まる。こうして取得されたソース・コードはそれ自体を最終的な情報とするというより，次のリバース・エンジニアリングの対象資料として使用するという場合が多い。②オブジェクト・コード。オブジェクト・コードも，それ自体が最終的な抽出情報となるというより次のリバース・エンジニアリングの対象資料として使用される場合が多い。③インターフェイス情報。コンピュータを中心として，種々のハードウェア及びソフトウェアから構成されるコンピュータ・システムにおいて，二つ以上の構成要素を接続させるために必要な情報がインターフェイス情報である。通信プロトコル，メディア・フォーマット，データ・フォーマット等は重要なインターフェイス情報である。④ユーザ・インターフェイス。入出力画面などプログラムの使用に当たって，ユーザからの入力またはユーザに対する出力の形式・方法などプログラムの使い勝手に関する情報である。

そして，リバース・エンジニアリングで用いられる手法は，主に次の通りであるが，その目的に応じ，幾つかの手法を組み合わせて実施される場合がある。①マニュアルの調査。マニュアルからプログラムについての情報を抽出する。②テストラン。入力に対するプログラムの反応を調査する。③接続テスト。相互接続のために作られたプログラムを実際に接続対象のプログラム等に接続し，正常に作動するか否かを確認する。④回線トレース。調査対象のプログラムに通信回線を介して様々なメッセージを送信し，それに対する応答のメッセージを（送信メッセージを含め）採取し，採取したメッセージを時系列的に印刷し，通信プロトコルなどを調査する。⑤記憶媒体のダンプ。記憶媒体に記録されたデータや制御情報をプリンタ等に出力し，記録形式を調査する。⑥メモリダンプ。主記憶装置（メインメモリ）上に記録されたプログラム（オブジェクト・コード）やデータの一部を印刷したり，ディスプレイ画面に表示し，調査する。⑦ディスアセンブル，デコンパイレーション[1]。機械語のプログラムを人間が理解できるレベルのオブジェクト・コード，又はソース・コードに近い状況に変換し，調査する。⑧ソース・コードの調査。調査対象プログラムのソース・コードを分析し，調査する。

2．著作権法上の問題と法的対応の現状

リバース・エンジニアリングが著作権法上問題となり得るのは，リバース・エンジニアリングの過程で複製又は翻訳行為が存在する場合である。

プログラムをコンピュータで実行させる場合，当該プログラムをコンピュータ本体内の記憶装置に一度記憶させなければならない。これは瞬間的

かつ過渡的なものであり，著作権法上の複製には該当しないとの解釈が一般的である。従って，著作権法上問題となるのは，情報取得の過程におけるプログラムの実行に必要な限度を超えた複製又は翻訳行為が存在する場合である。

これを前提とすると，上述したリバース・エンジニアリングの八つの手法には，①～⑤については，一般に著作権法上の問題となるような複製または翻訳行為は存在しないが，⑥～⑧については，このような複製または翻訳行為が存在する場合があると考えられる。この場合において，

(1) このような複製または翻訳行為に関して，著作権法上例外として適法とすべきか，あるいは他の伝統的著作物と同じような取扱い，違法とすべきか，

(2) このような複製または翻訳行為を著作権法保護の例外として適法とすれば，この例外に対して何らかの制限を設けるべきか，あるいは制限しないとすべきか，

(3) 制限すべきとすれば，どのように制限するのか，

などの問題については，激しい論争が行われた。この論争の結論として，リバース・エンジニアリング自体について，著作権法上，認めるべきと禁止すべきという対立した提案が出てきた。

この問題については，国際的範囲で，日・米・欧の三極において，活発な研究，議論または法的対応が行われた。

欧州共同体（EC，現欧州連合EU）においては，1991年に5月に，欧州共同体閣僚理事会が，コンピュータ・プログラムの法的保護に関するEC指令を採択した。これによって，一定の条件の下に，リバース・エンジニアリングのための複製又は翻訳を著作権法保護の例外として許容することが取り決められた。

アメリカでは，1992年9月と10月に，リバース・エンジニアリングのための中間的複製が著作権法上の公正使用に該当するとした二つの連邦巡回区控訴裁判所判決が出され，リバース・エンジニアリングのための複製については，公正使用の法理に照らして，ケース・バイ・ケースで，その適法性を判断する道を（判例上）選んだ。

日本においては，1993年7月22日文化庁が，リバース・エンジニアリングについては，著作権法を改正し明文で認める方向で検討を始めた。翌年5月，文化庁は「コンピュータ・プログラムに係る著作権問題に関する調査研究協力者会議報告書—既存プログラムの調査・解析等について—」という報告書を出したが，リバース・エンジニアリングを著作権保護の例外とすべきか否かについて，合意を見るに到っていなかったし，報告書の結論としては，具体的な法改正の提言にも至らなかった。

3．検討の目的と方法

本研究の目的は，このような国際情勢のなかで，日本の著作権法は，リバース・エンジニアリングに対して，どのように対応すれば，最も好ましいか，つまり，日本におけるリバース・エンジニアリングの著作権法保護の在り方を探して行きたい。

研究の方法としては，まず第2章で，EC指令におけるリバース・エンジニアリングの法的保護につき検討を行う。次に第3章で，アメリカにおけるリバース・エンジニアリングに関する判例及び公正使用の法理を検討する。そして第4章で，ECにおける保護およびアメリカの取り扱い方を踏まえて，日本におけるリバース・エンジニアリングに関する論争に基づいて，上述の三つの問題につき，著作権法上の総論的検討を行う。そして最後に，結びとして，日本におけるリバース・エンジニアリングの著作権法保護の在り方について述べる。

第2章　EC指令におけるリバース・エンジニアリングの法的保護

第1節　概説

ECは，1991年5月14日，「コンピュータ・プログラムの法的保護に関するEC閣僚理事会指令」[2]（以下，指令と呼ぶ）を正式に採択した。

指令はその成立までに，数年がかかって種々の

提案があった。1985年にEC委員会（the Commission）は，EC市場統合に向けて，「内部市場の完成に関する白書」[3]を公表し，その中で，知的財産法のハーモニゼーションがEC市場統合の実現にとって不可欠の領域の一つであると位置づけた。その中，コンピュータ・プログラムに関しては，1988年6月にEC委員会が著作権と技術の挑戦についてのグリーン・ペーパー[4]を公表し，それを検討したうえで，指令案の起草に着手した。1988年12月に，「コンピュータ・プログラムの法的保護に関するEC閣僚理事会指令案」（以下，指令案と呼ぶ）[5]が採択され，1989年1月5日に指令案は閣僚理事会（the Council）が提出された。その後，指令案は立法手続に従って審議された。1990年10月18日，EC委員会は欧州議会（the Parliament）によって提案された主要な修正点及び各種利害関係団体の意見を反映した「コンピュータ・プログラムの法的保護に関する閣僚理事会指令修正案」（以下，指令修正案と呼ぶ）[6]を公表した。この指令修正案について，閣僚理事会，EC加盟国常任代表委員会が討論し，討論した結果は欧州議会で審議され，結局1991年，正式な指令が形成されるに至った。

指令は，29項の前文と11条の本文とから成り立っている。前文には，指令制定の背景や目的が述べられ，指令における主要用語が定義され，指令の保護方向などが記述されている。従って前文は，指令の本文を解釈し，それを具体的なケースに適用する場合の重要な指針となっている。本文には，コンピュータ・プログラムの著作権による保護について，ベルヌ条約の一般原理を具体化する上で，明確化する必要のあるポイントを中心に規定し，かつEC加盟国の既存著作権法はこれらの規定に沿って立法措置を講じなければならないとし，その他の部分はEC加盟国の著作権法の規定に委ねる形を採っている。

指令の内容は，主に次のポイントを確かにするため調和のとれた立法措置の提示を意図している。つまり，すべての加盟国は，言語（リテラリ）の著作物として著作権法の下でコンピュータ・プログラムを保護すること，そして①保護主体と保護範囲，②著作権者の排他権およびこれらの権利に対する例外，③保護期間，の三つである。その中で最も注目すべきものは，著作権者の排他権の例外——第5条第3項のブラックボックスアナリシス及び第6条のデコンパイレーション，すなわち現在産業界と学界の両方で活発に議論されているリバース・エンジニアリングに関する規定であると思われる。これらの規定は世界で初めて設けられ，今後EC加盟国を拘束する一方，EC以外の国々の著作権法の改正や解釈，或いは著作権に関する条約の改正などに大きな影響を及ぼすものと考えられる。

第2節　EC指令におけるリバース・エンジニアリングの作成経緯

第1款　EC指令案

指令案前文第8項での，ソフトウェア，ハードウェア間の相互接続，相互作動を達成するための規約は一般的にインターフェイスとして知られ，インターフェイスに含まれているアイディアや原則は著作権保護の対象とならないとされ，指令案第1条第3項にも同じ旨の規定がおかれていたが[7]，インターフェイス情報を取得するためのリバース・エンジニアリングについては規定されていなかった。

これについて，指令案の第1部分では次のように説明した。つまり，相互操作システム（inter-operative systems）を達成するため，システム間のインターフェイスに含まれているアイディア，ルール，又は原則を複製することが必要であるが，これらのアイディア，ルールなどを供給するコードを複製することは必要ではない。何故なら，プログラムのデコンパイレーションについては，技術上は可能であるが，実際には非効率的である。通常，効率的な方法は契約により情報を提供することである。そして，相互操作システムを達成するのに必要な情報へのアクセスの問題は指令の範囲外の問題である[8]。

相互操作性に必要な情報へのアクセスの問題に

ついては，指令案と共に出された「Commission conclusions on the occasion of the adoption of the Commission's proposal for a Council Directive on the legal protection of computer programs」で，情報へのアクセスは著作権の問題ではなく，むしろある市場において独占的企業がそのような情報へのアクセスを濫用的に拒否し，或いはそのようなアクセスを不合理に制限する問題として，EEC条約第86条の独占禁止法上の問題である，とされていた(9)。

このように，指令案の段階では，リバース・エンジニアリングの著作権法上の取扱いについては，指令案中には規定が設けられず，契約および独占禁止法に委ねられていた。

第2款　EC指令案の審議及び指令の採択

しかし，指令案が審議される過程において，インターフェイス保護の可否，そして，リバース・エンジニアリングが著作権法の中に認められるべきか否か，及びどのように認められるかについてが，大きな議論となった。

まず，コンピュータ業界についてはこれに対する見解が二分された。フランスのBill，イタリアのOlivetti，日本の富士通，NCR，アメリカのUNISYSなどが主のメンバーとなっているECIS (European Committer for Interoperable System)グループは，①インターフェイスの情報は著作権保護の範囲外に出されるべきであり，②著作権法がアイディアを保護しないという原則に基づき，インターフェイス情報を含むプログラムの中の全てのアイディアや原則について，研究，調査またはこれらのアイディア，原則を抽出するための，プログラムの複製および翻訳は制限されるべきではない，と提案し，リバース・エンジニアリングが自由にできることを指令上明らかにすべきと主張した(10)。これは，これらの企業の多くの製品は，商業的成功のための相互操作性に依存していたためである。一方，アメリカの大手企業であるIBM，DEC，Apple，Lotus及びMicrosoft等によりなっているSAGE (Software Action Group for Europe) グループは，①インターフェイスはコンピュータ・プログラムの一部として保護されるべきであり，②リバース・エンジニアリングの必要はなく，指令案を変更しないほうがよいと主張した(11)。これは，EC市場においてこれらのアメリカ企業はコンピュータ産業のハード，ソフト両面で圧倒的に優勢な地位を占めていたためである。

コンピュータ業界の議論に応じて，EC委員会，EC閣僚理事会および欧州議会は色々な作業を行った。まず欧州議会の一つの委員会は，①プログラムの研究または調査のためのリバース・エンジニアリングは認容すべきであり，②インターフェイス情報のみではなく，プログラム中の著作権法が保護しない全ての要素について，リバース・エンジニアリングができる，と提案した(12)。

これに対して，EC委員会は，プログラムの中の全てのアイディア，原則に対して，ブラックボックスアナリシスができるという条項を第5条に新しく設け，デコンパレーションはできないという妥協的な立場を取ったが，ECISグループは，これだけでは十分でなく，インターフェイス中のアイディアを取るためには時々デコンパイレーションも必要となると指摘した。EC委員会は，これを受け入れ，一般的な研究又は開発のためのデコンパイレーションを拒否した上で，インタオペラビリティ達成のためのデコンパイレーションのみを認めるようになり，指令案で新しい条項を設けることを決定した(13)。

これに対して，欧州議会は，①インタオペラビリティという目的だけでは不十分と思われ，メンテナンスという目的もリバース・エンジニアリングができるもう一つの目的として加えるべきであり，②プログラム「コードの複製及びコード形式の翻訳」というデコンパイレーションのための例外が，第4条に列挙した全ての行為まで拡張されるべきである，と提案したが，EC委員会は①メンテナンスの範囲が広すぎて，権利の濫用が招かれる，②デコンパレイションする場合のみには「コードの複製及びコード形式の翻訳」が必要である，という理由で，欧州議会の提案を認めなかっ

た(14)。

その後，インタオペラビリティの具体的な内容，例えば，インタオペラビリティの意味は二つのプログラムと共に働くのかそれとも競合プログラムの開発も含んでいるのか，インタオペラビリティ達成の対象プログラムはデコンパイレーションされた原プログラムに限定されるかどうか，インタオペラビリティの達成はソフトウェア間に限定されるのかそれともソフトウェアとハードウェアとの間でもできるのか，等について，激しい議論が行われた。

結局，前述したブラックボックスアナリシス及びプログラム間のインタオペラビリティ達成のためのデコンパイレーションを認める内容の修正が行われ，これらの内容が1990年10月18日の指令修正案の第5条及び第5条の2に規定された。これらはさらに修正され，1991年5月14日に採択された上記の指令において，これらの二つの内容はそれぞれ第5条第3項及び第6条のようになった。

第3節　著作権者の排他権の例外(1)——ブラックボックスアナリシス

第1款　序論

指令第4条「プログラム著作権者の排他的権利」は，次の行為を行い，又はその行為を行うことを承諾することを含む，と規定している。すなわち，

(1)　プログラムの複製。手段及び形式，部分か全体かを問わず，コンピュータ・プログラムのロード，表示，実行，転送または記憶などの過程に伴う必要な一時的若しくは恒久的な複製行為（第1項）。

(2)　プログラムの改変。プログラムの翻訳，翻案，改変及びその他の改変，並びにそれらの結果としての複製行為（第2項）。

(3)　プログラムの頒布。プログラムまたはその複製物を公衆へ頒布（貸与を含む）する行為（第3項）。

これらの広く述べられた排他的権利に対して，指令では一連の例外が設けられたが，第5条の規定はその一つである。

第5条の例外によれば，一定の場合には，著作者の許諾なしに複製や翻案などをすることが可能である。つまり，プログラム使用のために必要な複製と改変（第1項），パックアップ・コピー（a back-up copy）の作成（第2項），及びブラックボックスによるプログラムの機能の調査，研究，テスト（第3項）などの場合である。そのうち，リバース・エンジニアリングに関するのは，第3項である。

第2款　ブラックボックスアナリシスに関する内容

第5条第3項によれば，プログラムの複製物を使用する権利を有する者は，権利者の許諾なく，プログラムの要素の基礎にあるアイディアや原則を確認するために，プログラムのロード，実行，表示，転送又は記憶の過程で，プログラムの機能の調査，研究，テストを行う権利を有する。

これは，いわゆるブラックボックスアナリシス（Black Box Analysis）とよばれる行為である。つまり，ある入力に対して決まった出力を返す装置または機能によって，プログラムの中身や内部でどのような処理を行っているのかについて，手順通りに操作すれば望む結果が求められるという分析方法である(15)。従って，この分析方法はデコンパイレーションやアダプテーション，リアレンジメント，翻訳や改変などには及ばない。そして，翻訳や改変などには及ばないため，この分析方法によって，インタフェースにあるアイディアや原則のみならず，プログラムの全ての部分に含まれているアイディアや原則などを取り出すことができる。

指令の前文第19項にも，同じ内容の規定が置かれている。つまり，「コンピュータ・プログラムを使用する権利を有する者は，プログラムの機能の調査，研究，テストに不可欠な行為を実行することを妨げられない」。「但し，それらの行為は，プログラムの著作権を侵害するものであってはならない」。

更に，第9条は，本条項に規定する「例外に反

する契約条項は無効である」と明確に規定している。

第3款　ブラックボックスアナリシスが例外とされた意味

第5条第3項は,「プログラムの要素の基礎にあるアイディアや原則を確認するために」,ブラックボックスアナリシスができるという例外を設けた。一方，著作権法は，表現のみを保護し，アイディアまでを保護するものではないという点については，世界的なコンセンサスがある。その意味からは「アイディアや原則を確認するため」，プログラムに対するブラックボックスアナリシスができるはずであるが，それにしても，あえて本条項を例外として設けたということは，積極的に評価すべきと思われる。すなわち，プログラムには，他の古典的な著作物には見られない特色があり，そのゆえに明確に設けられた規定と解すべきである。

プログラムという著作物は，小説，映画，写真などの古典的な著作物と違って，直接目で見るとその中に含まれているアイディアを分かることはできず，必ずある機械（コンピュータ）で作動させて，はじめてそのプログラム中に含まれているアイディアを取り出すことが可能になる。つまり，人間はプログラムを直接使えず，コンピュータを通して使わなければならない。

プログラムの基本的な使い方は，コンピュータへのロード，実行，表示，転送又は記憶などの過程である。この過程において，指令第4条第1項で規定している「一時的な複製」に当たる場合があり得るであろう。というのは，まず，プログラムを実際にコンピュータで作動させるためには，当該プログラムをコンピュータ本体内の記憶装置に一度記憶させなければならない。これは，普通，著作権法上の複製行為に該当しないと解釈されているが，第4条第1項では「複製」という概念は必ずしも明確に規定されていない。電磁的または磁気的な方法により記憶させることは勿論複製の一つの形態であるし，そして広義には，ディスプレイに表示させる行為も一種の複製であるという見解もある(16)。これらの複製行為は，第4条第1項によって，著作権の侵害になりうるが，しかし，著作権法は表現のみを保護し，アイディアまでを保護するものでなく，つまりプログラムに対してもアイディアを取り出すことができることを認めるべきである以上，プログラムに含まれているアイディアを取り出すために上述のような不可欠な複製行為は著作権侵害とはならないのである。従って，本条項が例外として設けられたことは，少なくとも複製権侵害の抗弁に対する防衛のために意味があるであろう。

更に解釈論として，本条項におけるプログラムの観察，調査，テストは，複製，翻訳，翻案の範疇に入らず，プログラムを作動させる行為に過ぎないという見解もある(17)。

本条項が例外として設けられたもう一つの意味は，プログラムの要素の基礎にあるアイディアを確認するためにブラックボックスアナリシスができるとすることを明確にするところにあると思われる。小説，映画，写真などの古典的な著作物に対して，そのアイディアを取り出すための分析ができることはいうまでもないが，これに対して，プログラムのような新しい著作物の場合は，そのアイディアを取り出すための分析ができるか否かは必ずしも明確ではない。従って，法律上明確する必要があろう。

いずれにしても，EC指令において，リバース・エンジニアリングの一つの方法であるブラックボックスアナリシスが世界で初て明確に規定されたのは，重要な意味を持つと考えられる。

第4節　著作権者の排他権の例外(2)——デコンパイレーション

第4条に対するもう一つの例外は，リバース・エンジニアリングの一つの重要手段であり，指令第6条を中心として規定されているコンピュータ・プログラムのデコンパイレーションである。

第1款　デコンパイレーションに関する内容

指令第6条よって，一定の場合には，プログラ

ムのデコンパイレーションのための複製，翻訳が権利者の許諾なしに許容されることとした。このような場合は，どのような条件の下にデコンパイレーションのための複製，翻訳ができるか，得られた情報をどのように使用できるか，及びデコンパイレーションがどの限界までできるか，という三つの観点から制約されている。

1．デコンパイレーションのための複製・翻訳が許容される条件

まず，それぞれ独立に創作済のプログラムの間におけるインタオペラビリティ（interoperability）を達成するのに必要な情報（the information necessary）を取得するために不可欠な（indispensable）場合に限定されている。従って，インタオペラビリティの達成という目的以外の一般的な研究開発や，競合するプログラム開発のためのデコンパイレーションは認められない。そしてデコンパイレーションによって，インタオペラビリティ達成のため必要な情報しか取得できない。また，これらの必要な情報を取得する手段としてのデコンパイレーションは，不可欠な手段でなければならない。つまり，デコンパイレーション以外のその他の方法（第5条第3項に規定さているブラックボックスアナリシス）で，これらの必要な情報を取得することは不可能である。

次に，複製・翻訳ができるのは，プログラムの複製物を使用する権限を有する者に限られる。すなわち，プログラムの購入者，ライセンシー，或いはこれらの者から許諾を受けた者である。無権限者により行われたデコンパイレーションは，例えそれが独立に創作されたプログラム間のインタオペラビリティを達成するためであっても，許容されない。

そして，インタオペラビリティ達成のため必要な情報がデコンパイレーションする前に利用可能である状態にあること，つまり，公表されたプログラムの利用マニュアルや雑誌などにこれらの情報を公開しなかったり，他の方法（例えば原プログラム著作権者との相談又は契約など）でも入手できなかったことが必要である。

また，他人のプログラムの中でデコンパイレーションできるのはインタオペラビリティを達成するための必要な（necessary）部分に限られる。

2．デコンパイレーションにより取得された情報の使用範囲

デコンパイレーションにより取得された情報の使用範囲は，独立して開発されたプログラム間のインタオペラビリティを達成するための必要な範囲に限定されている。インタオペラビリティを達成する以外の目的での情報の使用及び他人への情報の提供などはできない。また，表現が実質的に類似しているコンピュータ・プログラムの開発，製造，販売またはその他の著作権侵害行為（例えば情報の出版や転送など）のために，情報を使用してはならない。

3．デコンパイレーションができる限界

第6条第3項では，デコンパイレーションすることは，プログラム著作権者の正当な利益を害さず，且つプログラムの通常の利用を妨げないと規定している。つまり，形式上は上述した指令第6条第1項，第2項の規定に合致するデコンパイレーションのための複製，翻訳であっても，権利者の正当な利益を害し，プログラムの通常の利用を妨げる場合には，権利者の許諾が必要である。また，指令第9条第1項には，デコンパイレーションを禁止する契約は無効であると規定された。

第2款　検討(1)——インタオペラビリティ

インタオペラビリティについては，指令前文第10項は，コンピュータ・プログラムの機能はコンピュータ・システムの他の構成部分及びユーザと通信及び共働することであり，この目的のために，ソフトウェア及びハードウェアの全要素が，所期の機能目的に必要な一切の場合において，他のソフトウェア，ハードウェア及びユーザと共働することを許容するために，論理的あるいは物理的な「相互接続」（interconnection）及び「相互稼働」（interaction）が必要であるとし，同第11項は，ソフトウェア及びハードウェアの全要素間のそのような「相互接続」及び「相互稼働」の手段を規

定したプログラム部分は，一般に「インターフェイス」というとする。そして同第12項は，この機能的相互接続及び相互稼働は，一般に「インタオペラビリティ」というとし，そのような「インタオペラビリティ」は，「情報交換し，交換された情報を相互に使用する能力」と定義した。

1．インタオペラビリティと競争製品の開発

第6条によってデコンパイレーションできるのは，まずインタオペラビリティを達成するためのプログラムの開発という目的に限られている。言い換えれば，「相互接続」及び「相互稼働」のインターフェイスを開発するために対象プログラムのインターフェイス仕様を理解する場合にのみデコンパイレーションが認められる。

実際に問題となるのは，「相互接続」及び「相互稼働」には，対象プログラムとの接続製品のほか，競争製品の開発も含まれているかということである。

指令の成立過程における同条第1項については当初，デコンパイレーションが「独自に創作されるプログラムとデコンパイレーションされたプログラム（decompiled program / original program)」とのインタオペラビリティを達成するためにのみ許容される、という提案があったが(18)，結局EC閣僚理事会では，現在の「独自に創作されるプログラムと他のプログラム（other programs)」と変更された。この変更は，指令審議過程における一つの議題――競争製品の開発の可否についての結論であると見做された(19)。

この点についてEC委員会は，その競争政策に関するレポートにおいて次のように説明した。すなわち，独自に創作されたプログラムはデコンパイレーションされたプログラムに接続することができるだけでなく，競争することもできる。競争する場合にはこのプログラムとデコンパイレーションされたプログラムとを接続しない。しかしながら第6条によって，デコンパイレーションがインタオペラビリティを達成するため以外の他のことには許容されないので，競争する場合には，インタオペラビリティと関係ないの，デコンパイレーションされたプログラムのある部分を複製するプログラムの開発のためのデコンパイレーションが許容されない(20)。

つまり，指令におけるデコンパイレーションが許容されるのは，「相互接続」プログラムの開発ばかりでなく「相互競争」プログラムの開発も含まれているが，この「相互競争」プログラムの開発はインタオペラビリティと無関係なことではなく，デコンパイレーションされたプログラムと同じようにこの「相互競争」プログラムは他のプログラムとインタオペラビリティを達成しなければならない(21)。すなわち，インタオペラビリティを達成する目的でなく，単に競争プログラムの開発のためのデコンパイレーションは禁止されている。何故なら，この場合には通常，インターフェイス仕様のみでなく，プログラム中の全てのアイディア（例えばプログラムの目新しいアルゴリズム）及びプログラムの機序，構造，機能などもデコンパイレーションの対象となり，かつ同じように開発しなければならない。この場合は，まず第6条におけるデコンパイレーションの対象の範囲（インターフェイス）を超えていて，そして開発したプログラムとデコンパイレーションされたプログラムとが類似製品になり著作権侵害になる恐れが多いからであると思われる。

従って指令において，競争製品の開発のためのデコンパイレーションが許容されるかについては，もっと厳密な答えとしてはその最初の目的によって決まると言えよう。つまり，インタオペラビリティ達成のための目的で，結局開発されたプログラムがデコンパイレーションされたプログラムと競争する関係となっても，デコンパイレーションは許容される。これに反し，最初の目的が単なる競争製品の開発であったら，デコンパイレーションは禁止される。この意味で，競争製品を開発する場合にはインタオペラビリティの達成という目的についての証明を保存することも大切であろう。

2．インタオペラビリティとハードウェア

もう一つの問題は，指令にいうインタオペラビリティとは，ソフトウェア間の相互接続及び相互

稼働以外に，ソフトウェアとハードウェアとの相互接続及び相互稼働をも意味しているのか，言い換えればハードウェアとのインタオペラビリティを達成するためのデコンパイレーションが行われることを許容するのか，ということである。

この問題に関しては，指令には次の通り規定されている。

まず第6条第1項には「第4条第1項，第2項に定めるコードの複製及び形態の翻訳が，独立して創作されたコンピュータ・プログラムと他のプログラムとのインタオペラビリティを達成するのに必要な情報を取得するために不可欠な場合には，権利者の許諾は必要とされない」と規定されている。そして指令前文第10項，第11項では「ソフトウェア及びハードウェアの全要素が，所期の機能目的に必要な一切の場合において，他のソフトウェア，ハードウェア及びユーザと共働することを許容するために，論理的あるいは物理的な相互接続及び相互稼働が必要である」，「ソフトウェア及びハードウェアの全要素間のかかる相互接続及び相互稼働の手段を規定したプログラム部分は，一般にインターフェイスと呼ばれており」とされ，また前文第23項には「この例外の目的は，コンピュータ・システムの一切の構成部品の接続を可能とすることであり，それによって，それらは共働し得るのである」と規定されている。(強調下線は筆者による。)

この問題については，上記の規定に基づいて，三つの解釈論があった。

一つは，指令の規定は不明確であるが，実際に大きな問題とならないという見解である[22]。つまり，指令には第6条と指令前文とは矛盾しているので，インタオペラブルなハードウェアを開発するためのデコンパイレーションができるか否かについては不明確と言える。そのため，ハードウェアとのインタオペラビリティを達成したい場合に，対象プログラムをデコンパイレーションした後，直接にインタオペラブルなハードウェアを開発することは，法律違反という恐れがあるが，その代わりに，対象プログラムとハードウェアの間に，同時にハードウェアと対象プログラムと両方ともインタオペラビリティを達成した新しいプログラムを作っていれば，法律上の問題もないし，ハードウェアと対象プログラムとのインタオペラビリティも簡単に達成される。

もう一つは，指令はハードウェアとのインタオペラビリティを達成するためのデコンパイレーションを認めず，且つこれを認める必要がないという見解である[23]。その理由としては次のように言われている。(1) 第6条ではインタオペラビリティを達成するためのデコンパイレーションは明らかにプログラムの間に限定されている。ハードウェアとのインタオペラビリティを達成したい場合には，新しく作ったインタオペラブルなプログラムをそのハードウェアと対象プログラムとの間に入れれば十分である。そして前文の規定と第6条の規定とは矛盾するのでなく，前文の規定は，上述したプログラムとハードウェアの間に入れる新しく創作されるインタオペラブルなプログラムに関する規定である。デコンパイレーションを認めるのは，インタオペラブルなハードウェアの開発のためでなく，この相互接続できるプログラムの開発のためである。(2) インタオペラブルなハードウェアの開発のデコンパイレーションに関する規定は他の法制度によるべきであって，これを認める必要がない。(3) 第6条の適用範囲をインタオペラブルなハードウェアの開発のデコンパイレーションまで拡大するとすれば，あるプログラムの場合にはそのプログラムの全体に対するデコンパイレーションが許容されるという結果になってしまう。

三つ目の見解は，指令の規定は不明確であり，明確な規定が必要だというものである[24]。つまり，(1) 指令は第6条と指令前文とが矛盾しているので，インタオペラブルなハードウェアを開発するためのデコンパイレーションができるか否かについては不明確と言える。(2) この点については明確な規定が必要である。ハードウェアとのインタオペラビリティを達成したい場合に，新しく作ったインタオペラブルなプログラムをそのハー

ドウェアと対象プログラムとの間に入れることによって，技術的には必ずしも全てのケースにおいてインタオペラビリティを達成することができるわけでなく，あるケースでは，ハードウェアとのインタオペラビリティを達成するためにプログラムへのデコンパイレーションが不可欠である場合がある。そしてインタオペラブルなハードウェアの開発のデコンパイレーションに関する規定は他の法制度によるべきという見解は間違いである。従って，もし上述したハードウェアとのインタオペラビリティを達成するためにプログラムへのデコンパイレーションが不可欠であるケースが存在するとすれば，この問題はソフトウェアの法的保護に関連する問題として，このようなデコンパイレーションができるという提案がもう一度必ず提出されることになる。

以上の解釈論を見ると，ハードウェアとのインタオペラビリティを達成するためにデコンパイレーションが行われることを許容するかについては，指令の規定が明確であるか否か，及び明確に規定する必要があるか否か，という二つのことで見解が分かれている。

まず，指令の規定が明確であるか否かについて，ポイントは第6条と前文の規定との関係についての理解であると言えよう。文字通りに見ると，第6条と前文とは矛盾しているようであるが，実際にはそうではない。前文の規定と第6条の規定との関係は，目的と手段との関係と考えられる。つまり，前文における「ソフトウェア及びハードウェアの全要素が，所期の機能目的に必要な一切の場合において，他のソフトウェア，ハードウェア及びユーザと共働することを許容する」などのような規定は，どのような要素（プログラムのみ，或いはプログラムとハードウェア）の間にデコンパイレーションができるかという例外についての規定ではなく，あくまでも指令の目的——ソフトウェア及びハードウェアの全要素の間にインタオペラビリティを達成することを可能とすること——に関する規定に過ぎないと思われる。また前文第23項の「この例外の目的は，（異なる製造者のものを含む）コンピュータ・システムの一切の構成部品の接続を可能とすることであり，それによって共働し得るのであり」という規定は，以上の理解の正しさを明らかに証明したものといえよう。このような目的を実現するために，第6条に手段として，デコンパイレーションについての規定が設けられた。つまりデコンパイレーションできるという例外は「独立して創作されたコンピュータ・プログラムと他のプログラムとのインタオペラビリティを達成する」ためのみに限定され，ハードウェアとのインタオペラビリティを達成するためのデコンパイレーションを明らかに排除した。そしてこの問題に関連するのは，指令の審議過程において，欧州議会は一度，デコンパイレーションの規定の適用範囲は「明確にインタオペラブルなハードウェアの開発までに拡大すべき」と提案し[25]，ECISグループも「ハードウェア装置とのインタオペラビリティを達成するためのデコンパイレーションを明白に許容すべき」[26]と提案したが，結局両方も却下された。これも，指令におけるデコンパイレーションがプログラム間にのみできるということの証明であろう。この手段を通して，指令の目的——コンピュータ・システムの一切の構成部品の接続または共働を可能とすることを実現する。従って，ハードウェアとのインタオペラビリティを達するためのデコンパイレーションについては，指令の指定は曖昧でなく，はっきり排除していると言える。

この手段によって，指令の目的を達成することができるか。つまり上述した三つの解釈論で議論されたもう一つの問題，ハードウェアとのインタオペラビリティ達成のためのデコンパイレーションの規定が必要であるか，という問題である。これについては，この問題自体はソフトウェアの保護以外の問題でないという第三番目の解釈論に賛成したい。何故なら，このデコンパイレーションが必要であるとすれば，結果としてインタオペラブルなハードウェアの保護に関する規定でなく，あくまでもソフトウェアに対してこのようなデコンパイレーションを認めたというソフトウェア法

的な保護の範囲以外のことであるからである。また，現時点でこのようなデコンパイレーションを指令で明確に認めるべきか否かというのは，技術上の問題と結び付く別の問題であろう。つまりハードウェアとのインタオペラビリティを達成するためにプログラムへのデコンパイレーションが不可欠であるケースが存在するとすれば，これを明確に認めるべきであろう。そうでないとすれば，現在の指令の規定のままでいいではないかと思われる。

以上の考察によれば，指令におけるハードウェアとインタオペラビリティ及びデコンパイレーションとの関係については，次のような結論となる。つまり，ハードウェアを含むコンピュータシステムの全要素間のインタオペラビリティを達するのが指令の目的である。この目的を実現するための手段として，プログラム間のインタオペラビリティを達するためのデコンパイレーションのみが認められている。

第3款　検討(2)——デコンパイレーションに対するその他の制限

1．「必要な情報」及び「必要な部分」

指令第6条第1項では，デコンパイレーションの対象が限定されている。つまり，デコンパイレーションによって取られる情報はインタオペラビリティ達成のための「必要な情報」(the information necessary)に限られ，具体的に対象プログラムに対してデコンパイレーションできる部分はインタオペラビリティ達成のための対象プログラム中の「必要な部分」に限られる。

しかしながらデコンパイレーションする前に，どのような情報が必要であるか，プログラムのどのような部分が必要な部分であるかについては，はっきり分からない時もありうるであろう。この場合には次の見解がある。つまり，必要であるような情報または必要であるようなプログラムの部分に対して，デコンパイレーションができる。デコンパイレーションした後，はじめてデコンパイレーションによって取られた情報が必要な情報でない，またデコンパイレーションされたプログラムの部分は必要な部分でないことが分かって，デコンパイレーションする時に十分な理由があったことの証明があれば，問題がない，ということである(27)。

これらの限定的な規定については，実際に大きな問題となれないと思われる。デコンパイレーションするのは，非常に難しく，時間や費用などもたくさんかかり，効率的な作業でない。そのため，必要でない情報または必要でないプログラムの部分に対しては，事実上デコンパイレーションされることはないであろう。しかし原プログラム開発を保護するためには，このような制限が明確に設けられることが必要である。

2．「不可欠な（indispensable）場合」

指令第6条第1項によれば，デコンパイレーションできるのは，インタオペラビリティを達するための必要な情報を取得するために「不可欠な場合」に限定されている。つまり，これらの必要な情報がどうしても入手できず，且つ第5条第3項に規定されているブラックボックスアナリシスによって取ることもできない場合である。言い換えれば，デコンパイレーション以外のおよそ考えられるあらゆる手段方法を尽くしてなおかつ情報が得られない場合にのみ，デコンパイレーションが認められる。

この制限については，デコンパイレーションしようとする者がどの程度それ以外の方法により必要な情報の入手について努力すれば十分とされるのか，どの程度の努力によりデコンパイレーションが不可欠な手段となるのか，という疑問が出されており，更にこのような「不可欠な場合」にのみデコンパイレーションが認められるのは，どの程度の努力をしたかとの立証が必要であるうえ，しかも努力すべき程度が不明確であるので，余りにデコンパイレーションする側に負担をかけることになりかねないのではないか，という見解もあった(28)。

この制限は一見厳し過ぎるようであるが，実際にはそうではなかった。上述したことと同じ原因

（デコンパイレーションするのは効率的な作業ではないこと）で，仮にこの制限がなくてもデコンパイレーションしようとする者はデコンパイレーションする前に，デコンパイレーション以外のおよそ考えられるあらゆる手段方法を尽くすことが自然なことであろう。そのため，デコンパイレーションしようとする者はデコンパイレーションする前に必要な情報を取るために何の努力をしたのかについて記載して且つ証明できれば十分であり，実際にあまり問題にならないであろう。

そして指令の成立過程を見ると，そもそもこの制限は，対立しているECISグループとSAGEグループのそれぞれの提案[29]のバランスを取るための妥協的な規定であり，特別な配慮がないということが分かる。

3．表現の実質的類似性（substantially similar in its expression）

第6条第2項(3)によれば，原プログラムと実質的に類似する表現を持つプログラムの開発，製造，販売またはその他の著作権侵害行為のために，デコンパイレーションにより取られた情報を使用してはならないとされている。

問題となるのは，どのレベルの表現が，どの程度類似していれば実質的に類似していることになるのか，つまり表現の実質的類似性の判断基準である。

この問題を解決する際には，幾つかの点に注意しなければならない。つまり①プログラム著作物はプログラム言語の制約を強く受けるため，表現の面では自由な選択の幅が狭いこと，②プログラム著作物はその実用性が強いこと，③プログラムの言語・規約・解法などを著作権法が保護しないこと，④インタオペラビリティやコンピュータ間の通信などがその特性であること，及び⑤プログラム著作物が伝統的著作物に比べると，一定のレベルの創作性が要求されていることである[30]。

また，この問題は単なる類似性の問題だけではなく，今まで解決されていないプログラム著作権保護の一連の基本問題――表現とアイディアの区分，プログラム著作権の保護範囲（類似性の範囲）にも関連している。

従ってこの問題については，少なくとも現時点で，具体的事実の下で妥当な解決を図るという方法しかないであろう。

第5節　評価

EC指令は，プログラムのリバース・エンジニアリングの重要手段であるデコンパイレーション及びブラックボックスアナリシスについて，一定の制限の下に，世界で初めて法的に許容したものであり，画期的なものである。

その成立までの審議過程を見れば，妥協の産物という側面があるが，しかし，プログラムのリバース・エンジニアリングにおけるプログラムの保護と利用者の利用との調整という非常に難しい問題について，対立する二つの立場の主張の微妙なバランスを取ったものであり，リバース・エンジニアリングを合法とするための一定線を示したものとして評価できる。

その内容を纏めて考察すれば，①EC指令の前文では，コンピュータ・システムの一切の構成部品の接続または共働を可能とすることを実現するという指令の目的が設けられ，②この目的を実現するため，第4条「プログラム著作権者の排他的権利」に対する例外――プログラムのブラックボックスアナリシス及びデコンパイレーションが許容され，③原プログラムの開発に十分なインセンティブを与えるため，これらの例外に一定の制限を設けられ，つまりブラックボックスアナリシスができる場合がプログラムのロード，実行，表示，転送又は記憶の過程に限定され，デコンパイレーションできる場合がプログラムの間のインタオペラビリティを達するのに必要な情報を取得するため不可欠な場合に限定され，④これらの例外の実現を確保するため，これらの例外を禁止する契約が無効であるとし，⑤プログラム著作権を保護するため，これらの例外とプログラムの著作権との衝突がある場合には，著作権を優先させ，⑥著作権の濫用及び権利行使によってもたらされる市場の独占を防止するため，指令前文第27項には，「本

指令の規定は，支配的供給者が本指令に定義されたインタオペラビリティのために必要な情報を利用とすることを拒絶した場合には，EEC条約第85条および第86条に基づく競争規律の適用を妨げるものではない」と規定された。よって，EC指令におけるリバース・エンジニアリングに関する上述の規定は，コンピュータ・プログラムの開発者の利益と社会利益とのバランスがよく取られ，内容も十分明確であると評価できる(31)。

そして，最も興味深いのは，インタオペラビリティという目的であると思われる。まず，この目的を達成するためにのみ行われたデコンパイレーションが認められるという規定の効果について考察すれば，事実上プログラム中のアイディアの一部を保護するようになったということが分かる。このような結果は伝統的な著作権の考え方からは，一見不適切に見えるかもしれないが，プログラムの特性とプログラム著作権の保護範囲を考慮する必要があると思われる。つまり，「プログラムにおいて真に重要なのは表現でなく，その機能である」(32)という特性を考えると，プログラムに対して表現のみを保護することが十分であるかという問題が予想され，そしてプログラムにはどの要素が表現であり，どの要素がアイディアであるか，プログラムの著作権の保護範囲がどのように設定されればいいのかという問題も出てくるであろう。これらの問題については，以上考察したように，EC指令では正面から取り扱われていないが，インタオペラビリティという目的を決めるということは，ある側面で言えば，事実上伝統的な著作権保護の「アイディア／表現二分法」という考え方と違って，コンピュータ・プログラムの著作権保護の適正な範囲を決定する際に生じる様々な問題に対し，一つの新しい考え方を提示したものと言えよう。次に，インタオペラビリティの内容，つまり相互接続と相互稼働（競争製品の開発を含む）について考察すれば，社会利益のために原プログラムの開発者に対する制限であるという側面が重視されることがわかり，適当な制限であるといえよう。

また，EC指令におけるこれらの例外規定は最近のアメリカの判決に一定の影響を与え，WIPOでのリバース・エンジニアリングに関する検討にも重要な参考となっていることからすると，今後の日本での立法論を含めた議論においても一つの基礎となるものと考えられる。

問題となるのは，インタオペラビリティ（それ自体は上述した通り，積極的な評価ができる）という目的だけでリバース・エンジニアリングを許すことは，リバース・エンジニアリングの目的にとって，限定しすぎてはないかという問題が予想される。製品開発以外の目的，例えば著作権侵害の発見というような正当な目的で，リバース・エンジニアリングができるのか，ということについては，EC指令は言及していない。このような目的のためのリバース・エンジニアリングを禁止するのは，適切とは言えないであろう。

第3章　アメリカにおけるリバース・エンジニアリングの取扱い

第1節　公正使用の内容

アメリカの著作権法においては，プログラムのリバース・エンジニアリングが適法であるか否かについての直接の規定はなく，主に著作権法第107条の公正使用（fair use）に該当するか否かの問題として議論されている。

第107条は，以下のように規定している。

第106条（著作権のある著作物の排他的権利）の規定にかかわらず，批評，解説，ニュース報道，授業（教室における使用のための多数の複製を含む），研究，調査等を目的とする著作権のある著作物の公正使用（複製物又はレコードへの複製その他第106条に明記する手段による公正利用を含む）は，著作権侵害とはならない。特定の場合には著作物の使用が公正使用となるかどうかを判定する場合には，次の要素を考慮すべきものとする。

(1) 使用の目的及び性格（使用が商業性を有するかどうか又は非営利の教育を目的とするかどうかの別を含む）

(2) 著作権のある著作物の性質
(3) 著作権のある著作物全体との関連における使用された部分の量及び実質性
(4) 著作権のある著作物の潜在的市場又は価格に対する使用の影響

(なお，第107条は1992年10月の著作権法改正により一部改正されており，次の文言が条文の最後に加えられた)。

上記のすべての要素を考慮した上で公正使用であるという認定が行われる場合には，著作物が未公表であるという事実はそれ自体では公正使用と認定することを妨げない。

このように第107条では，適法な使用方法の例と，さらに個々の事例における使用方法が公正であるか否かを判断する上で考慮されるべきいくつかの要素を示している。適法な使用方法の例は，公正な使用方法を網羅しているものではなく，特定の使用方法が公正使用であるか否かを決定する際には，裁判所に裁量の自由が認められている。そして，上記の四つの要素については，どれか一つの要素が他の要素より重要である訳ではなく，すべての要素について判断し問題がなければ公正使用に該当する訳でもない。これらの四つの要素のそれぞれについて認定し，かつ他の要素についての認定と比較衡量し，総合的に判断することになる。つまり，ある要素についてはマイナスに判断されても，他の要素を含め全体として判断すれば公正使用に該当すると判断される場合もある。

第2節 リバース・エンジニアリングに関する判例と検討

第1款 Sega v. Accolade事件とAtari v. Nintendo事件

1992年に，プログラムのリバース・エンジニアリング自体又はそのための中間複製がアメリカ連邦著作権法第107条の公正使用に該当するという趣旨の二つの判決——Sega v. Accolade事件[33]とAtari v. Nintendo事件[34](以下それぞれ「Sega事件」「Atari事件」)——が初めて出された。この二判決は，両方とも控訴審判決であってその影響力が大きく，アメリカにおけるリバース・エンジニアリングに関する基本的な姿勢が固まったと評価される一方，二つの事件は特定の事実関係に基づくものであり，これらの判決をもってアメリカにおける一般的な考え方が示されたと考えるのは適切ではないとの声もあるが，少なくともアメリカにおいては，この問題について公正使用条項で処理することとして，特定の場合のリバース・エンジニアリングが公正使用に該当し得るという方向で議論が流れていると考えられる。

1．Sega事件

(事実経過)

本件では，原告Segaはジェネスというビデオゲーム機とゲームソフトを製造販売する会社であり，被告Accoladeはコンピュータ関連ゲームソフトの開発と販売を行う会社である。Accoladeが開発した製品はジュネスのゲーム機で遊べるソフトが含まれている。

Accoladeがジェネスのゲーム機で遊べるソフトを開発した過程は，2段階に分けられる。まずAccoladeはSegaの本体一つと3つのゲーム・カートリッジを購入し，デコンパイレーションして，ソースコードをプリント・アウトし，3つのプログラムの共通領域を見つけ出し研究した。そして，インターフェイス仕様を発見するためにプログラムを修正したりして結果を研究した。リバース・エンジニアリングの最後の段階では必要な情報を入れた開発マニュアル(Segaのコードを一切含まないもの)を作成した。この段階でAccoladeは中間的複製物を作成した。第2段階では，Accoladeはこのマニュアル中の情報だけを使って「イシド」という互換性のあるゲームプログラム(もともとIBM機とアップル機用に開発販売されたものであり，Segaのプログラムと類似していない)を開発販売した。

これに対して，Segeが著作権，商標権[35]を侵害されたとして，1991年10月カリフォルニア州に所在する連邦地裁に訴えたのが本件である。

著作権の侵害については，Segaは①オブジェクトコードの複製は違法であり，違法に複製したオ

ブジェクトコードの使用及び複製に基づくプログラム開発は差し止められるべきであり，さらに②違法に複製したオブジェクトコードから得られた情報を基づく開発したプログラムは違法な製品であるとして，その販売流通を差し止めるように裁判所に求めた。これに対してAccolade側は，インドプログラムとSageのプログラムの間には類似性が存在せず，ただ単に開発の過程で複製を行っただけで著作権の侵害が成立するとした判例はいままでなく，②プログラム開発過程で行われるリバース・エンジニアリングに伴うプログラムの中間複製は連邦著作権法が規定している公正使用に当たるので侵害でない，と主張した。

（地裁の判断）

地裁はAccolade側の主張を退け，中間的であれ複製であれば著作権侵害に当たり，そしてリバース・エンジニアリングのための複製が著作権法第107条の公正使用に該当しないと判断した。その理由としては，①Accoladeによる複製は営利の目的を持っている。②この複製によって作られたプログラムはSegaのプログラムと直接競合しているものであり，Segaのプログラムの価格に不利な影響を与える。③Accoladeがデコンパイレーションされ たSegaのプログラムは未公表であり公正使用が適用される範囲は狭くなる。④半導体チップ法ではリバース・エンジニアリングのため特別の規定が設けられており，著作権法にはこのような規定が設けられていない以上，プログラムのリバース・エンジニアリングのための複製は違法である，ということである(36)。

Accoladeは地裁の判決全体に対して第9巡回区控訴裁に控訴した。

（控訴裁の判断）

第9巡回区控訴裁，デコンパイレーション行為自体が複製行為の定義（第101条）に当たり，プログラムの複製物の合法的所有者に与えられている「複製・翻案権」（第117条）の許容範囲を越えているとしながら，第107条の公正使用を適用して，Accoladeの複製行為は公正使用に当たり，著作権の侵害とならないとした。

判決では，公正使用の四つの判断要素について以下の通りに判断した。

(1) 1番目の要素（使用の目的および性格）

複製行為に営利上の動機があるのは公正使用の認定に不利に働くが，特定の営利目的の性質によって反駁し得うる、絶対的というより相対的な問題である。本件では①Accoladeによる複製は中間的なものであり，営利的利用は間接的である。②複製の直接の目的はSegaのプログラムを調査研究することであり，且つAccoladeとしては複製してデコンパイレーションする以外に方法がなかった。③複製の究極の目的は互換（Segaのゲーム機で動くこと）プログラムを開発販売して利益を得ることであるが，これによってSegaのゲーム機で動く独自に開発されるゲームソフトが増加することになる。社会的利益の面からこのことを考えれば，創作作品の普及という著作権法の目的にかなう。よって，営利目的による不公正の推定は破られ，この要素はAccoladeの有利に傾く。

(2) 4番目の要素（潜在的市場又は価格への影響）

AccoladeはSegaのゲーム機用ゲームソフトを市場に参入させ，合法的な競争者になろうとしただけである。ユーザは一つのゲームソフトだけを買うわけではない。SegaとAccoladeとが別のゲームソフトを販売している限り，Segaが参入してもSegaの市場に重大な影響を与えることは考えられない。また，他人の競争を妨げて市場を独占しようとするのは，創作的表現の促進という著作権制度の目的に反する。Segaが経済的損失を破るとしても少しである。よって，この要素もAccoladeにとって有利である。

(3) 2番目の要素（著作物の性質）

著作物の性質により保護の程度は異なる。アイディア／表現の区別をプログラムに適用するにも特殊な問題を提供する。つまり，プログラムは実用的な著作物であり，伝統的な文芸の著作物より著作権の保護範囲が狭い。AccoladeがSegaのプログラムの中で保護されない表現を複製したことは確かであるが，プログラムは特殊な作品であり，

その他の機能作品のように肉眼で直接読むことが不可能であり，読むためにはデコンパイレーション(そのためには複製が伴う)が不可欠(indispensable)である。このため，著作物の性質からも公正使用と言いうる。仮にオブジェクト・コードのデコンパイレーションそれ自体で不公正な利用になるなら，著作権者は機能的側面につき事実上の独占を得てしまう（第102条(b)）。この要素もAccoladeに有利である。

(4) 3番目の要素（複製された部分の量および実質性）

AccoladeがSegaのプログラム全体を複製しており，この点はSegaに有利である。しかしAccoladeの究極的利用が本件のように限定されている時（小さい部分だけを利用した。そして，究極的目的によって，創作作品の普及という著作権法の目的にかなう。）は，この要素の重要性は著しく低い。

このような総合的な判断によれば，Accoladeが行ったディスアセンブリは全体として公正使用に当たる。

2．Atari事件

（事実経過）

NintendoはNintendo Entertainment System (NES) というビデオゲーム機を販売している。NESはモニター，本体及びコントロールから成っており，本体にゲームのプログラムをおさめたゲーム・カートリッジを差し込んでプレーされる。Atariはゲーム・カートリッジのメーカーである。

Nintendoは非許諾のゲーム・カートリッジを受け付けないように10NESプログラムを開発した。本体とカートリッジの双方に10NESプログラムのマイクロプロセッサー又はチップが入っており，本体側はロック，カートリッジ側はキーの役割を果たす。カートリッジ側に「ロックをはずすメッセージ」がないと本体はカートリッジを動かせない。

AtariはNintendoのNESで動くカートリッジを作るため，本体とカートリッジ間の交信をモニタリングしたが，10NESプログラムを破れなかった。そして，チップの層を化学的に剝がしてオブジェクト・コードを顕微鏡的に調べたが成功しなかった。その後，AtariはNintendoのライセンシーとなったが，契約上，10NESなどの技術へのアクセスは厳しく制限されていた。

次の年，Atariの弁護士は「侵害訴訟の被告であってこの訴訟のためにプログラムのコピーが必要である」と虚偽の事実を述べて，著作権局から10NESのソース・コードを手に入れた。その後Atariは再び剝がしたチップからオブジェクト・コードを読むことを試み，10NESを0，1に書き直した。Atariは著作権局から得た情報をこの移記上の誤りを正すために使った。

10NES解読後，Atariは，NESのキーを外す「ラビット」というプログラムを開発した。Atariは異なるチップと言語を使ったので，ラビットは10NESと機能的には区別できない信号を発する。

この開発過程において，Atariは2通りの中間複製物を作った。①著作権局からプログラムを入手する前に，チップからオブジェクト・コードの一部を複製した。②著作権局からプログラムの複製物を入手した後，Atariはこの複製物をフォトコピーし，チップをデプロセスし，これから10NESのオブジェクト・コードを複製した。

これに対して，Nintendoはカリフォルニア州北部地区連邦地裁に，Atariの行為が著作権侵害になると主張し，仮差止めを求めた。その理由は，①Atariがリバース・エンジニアリングの過程で中間的複製物を作製したこと，②著作権局から入手したソース・コードを複製したこと，及び③Nintendoのプログラムと著しく類似したプログラムを作製したことである。

連邦地裁はNintendoの主張に分があると判断し，仮差止め命令を発し[37]，Atariの中間複製は著作権侵害であるとした。連邦区控訴裁は地裁の判決の結果は支持したが，中間的複製については地裁と意見を異にし，著作権第107条の公正使用に該当し違法でないと判断した。

（控訴裁の判断）

Atariのリバース・エンジニアリング行為が公正

使用に当たるか否かについて，裁判所は次のように判断した。

(1) リバース・エンジニアリング自体は公正使用に該当し違法ではない

著作者は，著作物に含まれる全ての要素について排他権を持つわけでなく，著作物のアイディア，プロセス，操作方法には権利が及ばない。著作権法は，ある作品の複製物の合法的な所有者が，その作品のアイディア，プロセスおよび操作方法を理解するために必要な努力をすることを許している。

この趣旨から，著作権の排他性に対する公正使用の例外が著作権法の第107条に「批評，注釈，ニュース報道，教育，学問また研究などの目的のため複製物を作成するような使用を含め著作物を公正に利用すること」は侵害にならないと規定されている。つまり，著作権法は「批評，注釈……又は研究」のための複製を著作権保護から外している。

第107条の立法経過では，新たな技術進歩を取り込むために裁判所は公正使用の例外を調整すべきであると示唆している。

第107条は，ある複製が公正使用になるか否かを決定するに際し，作品の性質を吟味することを要求している。作品の性質により，著作物のアイディアやプロセスを理解するために中間的複製行為が必要な場合には，その行為は公正使用である。

従って，コンピュータ・プログラムの中の保護されないアイディアを認識するためにオブジェクト・コードをリバース・エンジニアリングすることは公正使用である。

ただし，その際コンピュータ・プログラムの複製が公正使用となるためには，①保護される表現に対して営利的利用ができない。②保護されない要素を理解するのに必要な (necessary) 範囲を越えてはならない。③リバース・エンジニアリングをする者はプログラムの複製物の正当な所持者でなければならない。

(2) 本件では，Atariはプログラム複製物の正当な所持者ではないので，公正使用の有資格者ではない。

10NESプログラムの複製物を盗んだといった汚れのない，10NESを理解するために必要なリバース・エンジニアリングは公正使用である。

さらにAtariの最終プログラムとNintendoの10NESプログラムの間に著しい類似性が認められるため，Nintendoは著作権侵害を立証できる可能性が高いとして，地裁の仮差止めを認めた。

3．Sega事件とAtari事件における公正使用の適用

Sega事件とAtari事件において共に，地方裁判所は中間的複製行為は公正使用ではないと判断したが，しかし控訴審は前の判決を覆し，中間的複製は公正使用であるとの判決を下した。控訴の段階で反対の結論を導き出すに至ったキーは何であろうか。

Sega事件において，第107条に規定された公正使用の四つの要素を照らして具体的に分析する方法を取った。そのうち，第1番目の使用の目的と性質，第4番目の潜在的市場への影響および第3番目の使用された量と実質性については，中間的使用（直接の使用）と究極的使用（間接の使用）に分けて判断してきた。そして第2番目の著作物の性質については，プログラムは機能作品であり，これを理解するためにデコンパイレーションが不可欠であることによって，中間的複製行為の公正性を肯定した。

これについて，判決では次のように説明した。もっと伝統的な文脈で著作権問題を考えることに慣れた人々にとって，この結論は一見不適切に見えるかもしれない。本件は単純に考えると，Segaの商業競争相手であるAccoladeは競争製品を開発する準備の段階でSegaの著作権を持つプログラムのコードを殆ど丸ごと複製した事件ということになる。しかし，本件のキーは，我々はコンピュータ・ソフトウェア，即ち著作権法の世界における相対的な未探検の領域を取り扱っていることにある。「丸い穴に四角い釘」という諺にあるようなことは避けなければならない(38)。つまり，プログラムという機能的な著作物をデコンパイレー

ションするための中間的複製について，単純に従来の公正使用の考え方をそのまま適用することは避けるべきである。

一方，Atari事件においては，公正使用の四つの要素に照らして具体的に分析する方法に代わって，主として著作権法は，著作物のアイディア，プロセスおよび操作方法などを保護しないこと，および第107条の公正使用は作品のアイディア，プロセスおよび操作方法を理解するために必要な努力をすることを許していること，という基本的な立場から，コンピュータ・プログラムの中の保護されないアイディアを認識するためにオブジェクト・コードをリバース・エンジニアリングすることは公正使用であるとした。

4．中間的複製行為に適用される公正使用の意味

Sega事件とAtari事件における限られた事実に基づいてであるが，中間的複製行為に適用される公正使用の具体的意味は次のようになるであろう。

まず，中間的複製を行う主体は対象プログラムの合法的な所持者でなければならない。

そして，公正使用の四つの要素については，

①（使用の目的及び性質）その直接の目的は調査研究である。究極の目的は営利であっても，著作権法の目的にかなう場合，

②（使用される著作物の性質）その性質によって，プログラムを理解するために中間的複製が不可欠である場合，

③（使用された著作物の量と実質性）直接の使用が保護されない要素を理解するのに必要な範囲を越えてはならない場合であり，究極の使用が少量である場合（最終製品と使用された著作物と実質的に類似していない場合），

④（潜在的市場への影響）直接の使用として市場への影響がなく，究極の利用として市場への影響が小さい場合。

第2款　最新判例——1993年Atari v. Nintendo事件

Atari事件につき1992年9月10日に第二審判決が出された後，同事件においてまだ解決しなかった幾つかの問題が残っているとして，これらの残った問題につき同事件は再びカリフォルニア州北部地区連邦地裁に訴えられた。これらの問題を解決するために地裁は，1993年4月15日及び5月17日に，それぞれ二つの略式判決を下した。ここでは仮にこの事件を1993年Atari v. Nintendo事件[39]という。

この事件でリバース・エンジニアリングに関しては，一つの新しい問題——将来の互換のために，現在デコンパイレーションされるプログラムにおける（将来の何時かに）チェンジされる可能性が大きい部分に対してデコンパイレーションができるか否か——が提起された。

（問題の提起）

前述したように，Nintendoは非許諾のゲーム・カートリッジを受け付けないように「ロック・キー」のような安全システムの10NESプログラムを開発した。そのゲーム機の本体とカートリッジの双方に10NESプログラムのマイクロプロセッサー又はチップが入っており，本体側はロック，カートリッジ側はキーの役割を果たす。カートリッジ側に「ロックをはずすメッセージ」があれば，本体はカートリッジを働かせる。具体的に言えば，カートリッジ側から本体側へある特別な信号（音声）を流し，本体側がこの信号を受けてチェックし，正しい信号だったから本体はカートリッジを働かせるのである。

技術上の原因により，上記の信号を流れる過程には沈黙期間（silence）がある。つまり信号の流れが，音—沈黙—音—沈黙—音—……のような形になっている。この沈黙期間には，プログラムには何らかの指令も入っておらず，全く空白のような状態である。このため，本体側から信号をチェックするのは，信号の音の部分だけであり，沈黙期間とは関係がなかった。つまり，10NESプログラムにおいて，この信号の沈黙期間という部分は使われていなかった。

一方，Atariは10NESを理解するために10NESのオブジェクト・コードの一部をデコンパイレーションした。そのうち，信号の流れについては，

音の部分だけでなく，沈黙期間の部分をも含んでいた。

これについてAtariは次のように説明した。つまり，沈黙期間の部分は現在のゲーム機の本体ではチェックしていなかったが，しかし，将来の何時かこれをチェックする（例えば沈黙期間の長さが正しいかどうか）可能性が非常に大きい。そうすると，同じプログラム中の同じ機能について再びリバース・エンジニアリングをしなければならない。しかし，リバース・エンジニアリング自体は，時間と金が沢山かかり，効率でない大変な作業である。このためAtariは，将来の互換性を達成するため，上記の沈黙期間の部分に対応するコードに対しても中間複製の正当化を認めるよう，裁判所に請求したのである(40)。

（裁判所の判断）

Atariの請求に対して裁判所は次のような理由により，リバース・エンジニアリングは現在の互換を達成することに限定すべきと判断して，Atariの請求を認めなかった(41)。

(1) 従来の判例には将来の互換性を認めたものはなかった。

従来の判決を見ると，将来の互換性を達成するための中間複製を認めたケースがない。

(2) 社会的利益の立場から考えると現在の互換性のみを認めるべきである。

社会的利益を考慮する立場を取るのは，二つのことを意味している。一つは，Sega事件のように沢山のプログラムを開発させることであり，もう一つは，沢山のハードウェアを開発させることである。本件では，後者を考えなければならない。すなわち，ハードウェアの開発者は，ハードウェアの販売または使用ライセンスから利益を受ける。この利益は新しいハードウェアを開発するインセンティブである。リバース・エンジニアリングの範囲を現在の互換性を達成することを限定すれば，原ハードウェアの開発者は新製品の開発のため十分の時間（この間にライセンスすることによって利益を受けるかもしれない）がある一方，何時か誰かがこのハードウェアをデコンパイレーションし，その安全システムを解析することによって，ハードウェア開発者の市場に対する一時的な利益が終わりになって，市場の独占が達成できなくなるかもしれない。よって，ハードウェアの開発とソフトウェアの開発のバランスをよく取ることになる。そうしないと，ハードウェアの開発者は新しいハードウェアを開発しても，すぐ解析され，使用ライセンスによる利益がなくなる一方，リバース・エンジニアリングも速く且つ安く開発することができる。結局ハードウェアの開発とソフトウェアの開発のバランスが壊され，ソフトウェアの開発に傾く。

(3) マージ(merge)理論によっても現在の互換性のみを認めるべきである(42)。

マージ理論によって，アイディアと表現が混同する場合には，混同された表現に著作権は成立しない。しかし，本件における沈黙期間に対応するコードには，現在何の機能もなく，アイディアの表現と混同する可能性がないので，この部分は著作権の保護を受けるべき表現である。

（検討）

この判決に対して，ジョージ・ワーシンテォン大学のRichard H. Stern氏から強い反発があった。①裁判所のいうハードウェアとソフトウェアの開発のバランスは誤りである。新しいハードウェアを開発させるのは著作権法の役割ではない。②公正使用の四つの要素の間にバランスを求めるべきである。本件では，Atariの使用によって，Nintendoの「ロック・キー」安全プログラムの利益への影響があるが，公正使用の他の要素も合わせて判断しなければならない。その中第2番目の使用される著作物の性質を考えると，Nintendoの「ロック・キー」の安全プログラムはハードウェアをロックし，自分の利益のために他人のソフトウェアの使用を外すものである。このようなプログラムを著作権法で保護すべき理由は何であろうか。③実務においては将来の互換性は非常に重要である。もし互換されるプログラムの所有者がその関係するインターフェイスを変えるとすれば，このプログラムと互換している他のプログラムの

所有者にとって，重大な経済影響をもたらすことになる。そのため，もしプログラムのある将来の機能がやがて現実の機能になる可能性が大きい場合には，将来の互換性は現在の互換性と同じように重要であり，同じように考えるべきである(43)。

Richard H. Stern氏の第①②の見解は説得力を持つと思う。裁判所は公正使用を適用する時，社会的利益の面（本件においては，ハードウェアの開発にインセンティブを与えること）強調したが，実際には公正使用の第4番目の要素しか判断しなかった。そして，ハードウェアの開発とソフトウェアの開発とのバランスを求めるのは著作権法の公正使用の目的ではないので，裁判所の分析は適切ではなかった。

また，アイディアと表現のマージについては，現在の機能のない，アイディアと表現をマージする可能性がない部分は，必ずしも著作権の保護を受けるべき表現ではない。著作権が保護されるのは，アイディアの表現であり，アイディアがない場合には，その表現がどこにあるのかという疑問が残る。そして本件における沈黙期間に対応するコードが，表現であるとしても，技術的要請に応じる指令であり，創作性が低いと言える。従って将来の互換性の是非は別として，単にこの部分については，著作権法の保護を与えず，自由に利用できるとするのが適切であろう。

しかし，将来の互換性を達成することについて，ある部分が変わる可能性が大きいことに関する立証はあくまでも推測的なものであり，不確実である。そして，変わることが確実であるとしても，どこまで変わるのか（どこまで中間的複製できるのか）も曖昧である。もし，将来の互換性を認めるとすれば，中間的複製の範囲が不明確になったり，拡大されたりする恐れがある。この意味で，裁判所の判決の結論には賛成する。

この判決を通して，互換の製品を開発する場合には，中間的複製できる範囲は現在の互換性を達成するために限定されることが分かる。

第3節　公正使用についての検討

第1款　中間的複製は公正使用に該当するか否かに関する議論

アメリカにおいて，プログラムのリバース・エンジニアリングのための中間的複製が公正使用に該当するか否かについては，従来二つの考え方に分かれていた。

一方の考え方は，プログラムを研究しあるいはプログラムのアイディアやプロセスなど著作権の保護の対象とならない要素を抽出するための中間的複製は，たとえ企業が営利目的で行う場合であっても著作権法第107条の公正使用に該当し適法であるという主張である(44)。その理由としては，①このように解さないと，オブジェクト・プログラムの形態で市場に提供されているプログラムの場合，結果的には著作権法第102条に反し，アイディア等の著作権による保護が及ばない事項が保護されることになってしまう。②著作権法第107条の公正使用に規定されている要素（使用の目的，著作物の性質，使用された部分の量及び実質性，潜在的市場・価格に対する影響）を検討しても，調査・研究ためのリバース・エンジニアリングに伴う中間的複製は公正使用に該当する。

これに対して，中間的複製は著作権法第107条の公正使用に該当しないという主張も行われている(45)。その理由としては，①営利の目的で行われるプログラムのリバース・エンジニアリングに伴う中間的複製は，「研究」とは言え社会全体の利益のためではなく，元のプログラムに競合するプログラムを開発するという自己の営利の目的のために行われるものである。このような営利性はプログラムのアイディアを抽出するというリバース・エンジニアリングの性格により変わるものではない。②著作権法第102条はアイディアそれ自体の保護を否定しているのであり，著作権保護の対象である表現からアイディアを抽出するための複製が第107条の公正使用となることの根拠とはならない。③著作権法第107条の四つの要素から判断しても公正使用とはならない。④半導体チップ法ではリバース・エンジニアリングのための特別の規定

が設けられており，著作権法にはこのような規定が設けられていない以上，プログラムのリバース・エンジニアリングのための複製は違法であると解すべきである。

Sega事件とAtari事件における公正使用の原則適用についても，議論が分かれている。ハーバート大学のMiller氏は，一方の考え方として，中間的複製に対する公正使用の適用に反対した(46)。これに対して，アリゾナ州立大学のKarjala氏は，Sega事件の判決の合理性を認め，中間的複製は公正使用に該当すると主張した(47)。

第2款　公正使用の法理の解釈

中間的複製が公正使用に該当するか否かという問題は，公正使用の法理をどのように解釈するかという問題であると思われる。

1976年アメリカ連邦著作権法で成文化された公正使用の法理については，様々な解釈が試みられた。一般の解釈としては，特定の状況において著作権法が刺激する創作的な活動が妨げられる場合，裁判所はこの法理によって著作権法の融通のきかない適用を回避することが許される，といわれてきた(48)。一部の学者たちは，この法理の融通性を，特に急激な技術変化の時代には，著作権法の「安全弁」と考えた。また，公正使用の法理が不明確，曖昧であると考える学者もあった。

アメリカ議会は，1976年著作権法に公正使用の例外を成文化する際，特定の使用が公正使用と解釈されるか否かを決定するための特定の基準を定めなかった。その代わりに議会は，裁判所が公正使用の判定を行う際に考慮すべき要因を法定して列挙した。つまり現在アメリカ連邦著作権法第107条に規定されている四つの要素である。議会は，この四つの要素は「いかなる場合にも限定的若しくは決定的なものではなく」，「平衡を保つための若干の標準を提供するものである」ことを了解した(49)。議会は，個々の公正使用の事件をめぐる状況を分析するための一連の融通性のある基準を作り上げ，個々の事件はケース・バイ・ケースの司法的な分析に任せたように思われる。従って裁判所は，公正使用の要素を適用，評価する場合には，かなりの自由を持っていることになる。

公正使用の四つの要素に対する具体的解釈としては，次のようになっている。

第1番目の要素，すなわち使用の目的および性格を評価するに当たって，裁判所は必ずしも常に「営利的な性格」の使用が公正使用の認定を排除すると判示してきたわけではなく(50)，また「非営利の教育」目的が公正使用を認定されると判示してきたわけでもない(51)。しかし，被告が著作物を教育，科学又は歴史学の目的に使用する場合には，その要素に基づく公正使用の抗弁が認められることが多い(52)。

第2番目の要素，すなわち著作物の性質は，それぞれ特定の事件の事実と状況に基づいて考慮されるべきである。例えば未公表の著作物については，公正使用の法理の範囲を狭く解釈してきた(53)。

第3番目の要素，すなわち使用された著作物の部分の量と実質性については，裁判所は量（どれだけの部分が使用されたか）と質（著作物の「核心」又は本質的部分であるか否か）の両面を検討すべきであるとする。裁判所は，原則として，著作物全体若しくはその実質的部分の複製については公正使用を認めなかった。しかし著作物の全体の複製という問題は，1976年法の成立以前は重要な論争の争点であったが，論争の結果は，限られた状況の下での種の複製行為が認められることとなった(54)。

第4番目の要素，すなわち市場への影響については，裁判所は申し立てられた被告の行為を検討し，被告の行為が原告の現在の著作物の潜在的市場または価格に対して実質的に有害な影響を及ぼすか否かを調査してきた(55)。

そして，1991年議会は，著作権の公正使用の問題をかなり詳しく取り上げ，公正使用の法理が，未発行の著作物を無許諾で使用することは許されるかどうかを検討した。結局，1991年，上院は，S. 1035を可決し，著作物が未発行であることは，それ自体としては侵害に対する公正使用の抗弁の

適用可能性を妨げるものではないことを明らかにした。

第3款　Sega事件とAtari事件の判決の合理性及びその問題点

以上の公正使用の法理の解釈についての考察によれば，Sega事件とAtari事件で取り上げたプログラムのリバース・エンジニアリングのための中間的複製に対して，公正使用の抗弁の適用可能性があるのは明白であろう。

1．Sega事件とAtari事件における公正使用の適用について

Sega事件判決は，まず中間的使用と究極的使用にわけて判断した。これは，リバース・エンジニアリングの特徴に合致する。前述したように，リバース・エンジニアリングというのは既存の製品を調査・解析してその構造や製造方法などの技術を探知することであるが，殆どのリバース・エンジニアリングがこのような技術を探知する段階に止まらず，その探知された技術を利用して新しい製品を開発するまでいく場合が多かった。リバース・エンジニアリングと新しい製品の開発は相対的独立の二つの段階であるが，しかし新しく開発された製品とリバース・エンジニアリングされたプログラムとの間に法的問題が起きやすいので，この相対独立している二つの段階は常に繋がっている。これは，具体的な中間的複製が公正使用であるか否かについて判断する際に，最終の製品の様態も含めて判断すべき原因である。この場合には，中間的使用と究極的使用にわけて検討するのが適当であろう。次に，Sega事件判決は，公正使用の四つの要素に照らして判断する際に，著作権法の目的に適うか否かについても配慮した。前述したように，公正使用は著作権法の「安全弁」であり，公正使用の四つの要素は「いかなる場合にも限定的若しくは決定的なものではなく」，「平衡を保つための若干の標準を提供するものである」ので，特に急激な技術変化の時代には，場合によっては，著作権法の目的に照らして判断する必要もあるであろう。そして，Sega事件判決は，プログラムの特性――機能性の面も考慮した。それは，公正使用の第2番目の要素に合致するのみならず，公正使用というケース・バイケースの法理をプログラムという特別な著作物に適用する場合には，検討する仕方として正しいと言えよう。

Atari事件判決は，Sega事件と同じように，プログラムの機能の面も考慮した。そして①著作権法はアイディアや機能などを保護しない，②公正使用は作品のアイディア，プロセスおよび操作方法を理解するために必要な努力をすることを許すと判断した。著作権法の基本的立場として，①は間違っていないが，プログラムの中間的複製に公正使用を適用する根拠とする具体的な説明がなかったので，抽象過ぎるのではないかという気がする。プログラムにおいて，どの部分がアイディアであるか，どの部分が表現であるかは，現時点ではっきり予測することはできないし，Atari事件の判決においても，どの要素が保護されるか，どの要素が保護されないか，という判断も曖昧であった。そして，根拠②は，公正使用の適用の根拠よりも，公正使用についての解釈と言ったほうがよい。

2．中間的複製行為に適用される公正使用の意味について

第1節で考察した公正使用が中間的複製に適用された場合の具体的意味，および1993年Atari v. Nintendo事件の判決に明示された，互換製品を開発する場合には中間的複製できる範囲は現在の互換性を達成するために限定されることを含めて考えて見ると，Sega事件とAtari事件における限られた事実に基づいて，中間的複製行為に適用される公正使用の具体的意味は，合理的な内容を成していると言えよう。

そして，前述したように，公正使用の四つの要素は「いかなる場合にも限定的若しくは決定的なものではなく」，「平衡を保つための若干の標準を提供するものである」ので，特に急激な技術変化の時代には，場合によって，対応しやすい利点もあるであろう。

しかし，幾つかの欠点も感じられる。要素①についての判断には，営利（リバース・エンジニア

リングされたプログラムの開発者に対する競争利益）と社会的利益（著作権法の目的に適うこと）が同時に存在する場合には，両利益の比例関係を考慮せず，いつでも社会的利益のほうが優先されるのか，それとも，比例関係によって具体的バランスを取るのかについては，判決は説明しなかった。要素②については，Sega事件とAtari事件は共にプログラムの機能性について検討したが，しかし，両事件におけるデコンパイレーションされたプログラムは共に「ロック・キー」のような安全プログラムであり，このプログラムはハードウェアをロックし，自分の利益のために他人のソフトウェアの使用を外すものであり，このようなプログラムが著作権法の目的に適うとは言えないであろう。公正使用の法理を適用する場合，特に著作権法の目的にも照らして判断する際には，このようなプログラムの性質も考慮すべきであろう。要素③については，Atari事件の判決を見ると，保護されない要素とは何にかが曖昧であり，実質的類似性についての判断基準も不明確であった(56)。

そして，上述したように，中間的複製行為は公正使用に該当するか否かを判断する時は，その四つの要素および要素ごとにおける直接と究極の使用様態，著作権法の目的，またプログラムの特性などの事項を考慮しつつ総合的に判断する。しかし，具体的事件においては，どの事項にどの程度，比重をかけるかというのは，問題となるので，おそらく異なる裁判所によってその判断も大きく異なってくるであろう。この点は，公正使用の法理に関する今後の判例の動向に注目すべきと思われる。

第4節　評価

1992年4月米国議会技術評価局は，コンピュータ・ソフトウェアの法的保護の在り方について検討した報告書(57)を公表した。報告書の第1章においては，プログラムのリバース・エンジニアリングを巡る不明確な状況に対処するため，(1)法第102条の改正或いは独自法の制定によりプログラムの保護範囲を明確化する，あるいは(2)立法方式または協調方式によって，法第107条及び第117条を明確化する，などが提言された(58)。

同年の9月と10月に，それぞれAtari事件とSega事件の判決が出された。この両判決は事実上，アメリカにおけるリバース・エンジニアリングを巡る不明確な状況を終結させて，リバース・エンジニアリングの問題を第107条の公正使用の法理により処理するという道を選んだ。

公正使用の法理によりリバース・エンジニアリングの問題が処理されるという道は，リバース・エンジニアリングが絶対的に合法であるとしたものでなく，ケース・バイ・ケースの判断に止まったものである。そのため，リバース・エンジニアリングに対する法的対応として，不明確な面が存在することはいうまでもないであろう。

しかし，この道の選択によって，アメリカにおけるリバース・エンジニアリングに関する判例の流れが決まったばかりでなく，世界的な範囲で，EC指令の後，リバース・エンジニアリングについてのもう一つの処理方法が出てきたとも言え，他の国々に対しても，大きな影響が及ぼすことが予想されるので，積極的な評価をすることができる。

第4章　日本におけるリバース・エンジニアリングの著作権保護の在り方

第1節　日本における議論

日本の著作権法には，リバース・エンジニアリングに関する規定がない。そして，関連判決もない(59)。しかし，これについて活発な研究と討論が行われている。平成6年5月日本文化庁から出された『コンピュータ・プログラムに係る著作権問題に関する調査研究協力者会議報告書―既存プログラムの調査・解析等について―』報告書は，リバース・エンジニアリングについての研究現状が集中的に反映されたものと言える。

この報告書によれば，既存プログラムのリバース・エンジニアリング（調査・解析）に伴う複製または翻案に関する著作権制限規定を設けるべきかどうかについては，以下の四つの意見があった。

(1)　調査・解析一般については，それに伴う複

製または翻案に関する著作権制限規定を設けるべきであり，調査・解析の目的は特定すべきでない。

理由としては，著作権法は表現を保護するが，アイディアを保護しない。著作権法上，公表された著作物の表現を介して著作者のアイディアを知りかつ利用することは禁止されておらず，著作権を，アイディアへのアクセスを妨げるための手段とすることは妥当ではない。プログラムの調査・解析は，プログラムの表現を介してそのアイディアを知覚するための手段であるから，著作権法上合法的な行為として捉えられるべきである。仮にプログラムの調査・解析の過程で複製または翻案があったとしても，その事実をもって直ちに著作権侵害であるとするのは，プログラムの背後にあるアイディアへのアクセスを禁じてしまうこととなり，適当でない。

そして，ハードウェアに対する調査・解析は，社会全体の技術の発展に不可欠であることから特許法および半導体集積回路の回路配置に関する法律によって許容されており，同様の技術的性格を有するプログラムを区別する理由はない。

(2) 調査・解析に伴う複製又は翻案に関する著作権制限規定を設けるべきであるが，特定の目的（調査・解析対象プログラムと市場における競合プログラム開発，その表現が実質的に類似するプログラム（海賊版プログラム）の開発など）の場合には許されないとすべきである。

理由としては，①プログラム中の技術思想の解読を可能とし，産業・文化の発展を促すという要請と先行的な技術開発のインセンティブを守るという要請との調整を図る必要がある。②市場において調査・解析対象プログラムと競合関係に立つプログラムの開発を目的とする調査・解析を許すことは先行的な技術開発インセンティブを減退させる恐れがあるので，その場合には許諾が必要とすべきである。③調査・解析対象プログラムと表現が実質的に類似するプログラム（海賊版プログラム）の開発を目的とする場合には，本来許容されるべき正当な目的とはならないので，このための調査・解析に伴う複製又は翻案は許されないとすべきである（なお，先行的な技術開発の保護のためには，この目的による調査・解析に伴う複製又は翻案のみが許容されないものとすればよく，海賊版プログラム以外の競合するプログラムの開発を目的とする場合は許容してもよいとの意見もあった）。④その他の場合，すなわちバグの修正，接続プログラムの開発，純粋な学問的研究などを目的とする場合には，調査・解析対象プログラムに対して市場における不利な影響を与えるものではなく，商品開発のインセンティブを減退させる結果にはならない。これらの場合には著作権は制限されるべきである。

(3) 調査・解析に伴う複製又は翻案に関する著作権制限規定を設けるべきであるが，許容される特定の目的（インタオペラビリティなどの達成，エラーの修正など）のためのものに限定すべきである。

理由としては，①新たな創作活動へのインセンティブを与えるという著作権法の基本的な目的に照らし，著作権の制限は社会的要請により合理性があると認められる必要最小限の範囲に止めるべきである。②コンピュータ産業の実態としてプログラムの動作する環境についての事実上の標準が存在しており，ユーザーの便宜を考えると，新製品の開発においてはこのような環境下で相互に運用できるインタオペラビリティを達成するために規約を合致させなければならないという要請がある。そのためには既存プログラムの調査・解析によりインターフェイス仕様を抽出する必要があり，これを認めないとコンピュータ市場における公正な競争及び技術の発展を阻害する恐れがある。③インタオペラビリティの達成以外にも，プログラムのエラー修正，著作権侵害の発見などは正当な目的として許容されるべきである。

(4) 調査・解析に伴う複製又は翻案を認める必要はなく，著作権制限規定は設けるべきではない。

理由としては，①著作権法上著作権者は著作物のいかなる複製又は翻案についても排他的権利を専有することが原則であり，プログラムの調査・解析に伴う複製又は翻案について特別に取り扱う

必要はない。②調査・解析（特にデコンパイレーション）に伴う複製又は翻案を許すと，既存プログラムの連続的な変更による海賊行為に利用されることとなる。③調査・解析は，プログラムの創作に必要な費用及び時間を回避し対象プログラムの著作権者に対する商業的優位を得るために用いられるものであるから，それに伴う複製または翻案を許容すると，著作物の通常の利用を妨げ，著作者の正当な利益を不当に害することとなる。

以上の四つの意見の他に，仮に著作権制限規定を設けることとした場合の規定内容についても，色々な考え方があった。そして，日本の現行法の解釈により調査・解析に伴う複製又は翻案を一定の範囲で許容することの可能性についても，検討した。

第2節　リバース・エンジニアリングが問題となる原因

リバース・エンジニアリングが問題となる原因は二つ考えられる。つまり，コンピュータ業界におけるリバース・エンジニアリングに対する客観的な要請とこれに対する強い反発，及び，著作権法の保護とコンピュータ・プログラムの評価の乖離である。

1．客観的な要請と強い反発

コンピュータ業界においては，一方でリバース・エンジニアリングを客観的に要求している。他方でリバース・エンジニアリングを認めることに対して強い反発もある。これは，コンピュータ・プログラムの性質，コンピュータ産業の状況，プログラムの市場の在り方，およびその他の社会的状況などに関連するものである。

プログラムの性質と利用形態について。一般的にプログラムに完成ということはない。一度できあがったプログラムについても不断のヴァージョン・アップを行い，性能を向上させ或いは機能を追加して初めて実用に耐えうるのである。また，特定のOSを有するハードウェアには，それに適合したアプリケーション・プログラムしか利用できないので，プログラムは単体で取引されることはなく，OSシステムから利用者が仕事に使うアプリケーション・プログラムなどまでの階層構造をなす複数のものとして作成され流通している。従って，プログラムがその機能を果たすためには，コンピュータ・システムを構成する他の要素やプログラムとの互換が可能でなければならないが，その実現には，接続部分において両者が共通の仕様に従うことが条件となる。

近年，通信回線を介したコンピュータ・システム間および端末間での情報伝達の重要性が高まっているが，その実現には，通信プロトコルと呼ばれる一定の手順を守らなければならない。

そして，データの相互利用，プログラムの使用過程でのバグの修正，改良なども普通であり，その実現には，対象プログラムの仕様を知ることが前提となる。

プログラムの市場の在り方について。現在コンピュータ産業では，支配的製品が事実上の標準となる現象が広く見られる。事実上の標準が存在する場合，これに従うことが市場に参入あるいは成功するための重要な要素となる。そして，JISなどにより様々な形で標準化の作業が行われているが，その達成はまだ不十分な状況にある。

その他の社会的状況について。まず，情報やアイディアの重要性が高まっている。それと同時に，情報やアイディアの経済的価値を制度的に保障するよう求める要求が強くなっている。そして，コンピュータ・システムについては，ハードウェアの製造コストの低下に伴い，ソフトウェアの価値が相対的に高まっている。ソフトウェア特にプログラムについては，これを効率的に開発する手段が十分には確立されていないため，その開発費用はプログラムの巨大化により益々増大している。また，情報独占に対する危機意識が存在する。情報の開示については，社会的に合意されたシステムが成立していない。

上述した各事情の中で，コンピュータ・システム間の互換，通信による情報の伝達，データの相互利用，およびプログラムのバグの修正，改良な

どは，対象コンピュータ・システムのインターフェイス情報などを知る必要性およびその手段としてのリバース・エンジニアリングの必要性を示している。

プログラム市場における事実上の標準の存在は，リバース・エンジニアリングを行う企業と行われる企業という二極化をもたらす傾向がある。前者の企業はリバース・エンジニアリングを積極的に支持し，後者の企業は消極的態度を取っている。

ソフトウェアの価値の相対的な高まりおよび作成費用の増大化により，リバース・エンジニアリングは先行者の投資の回収および投資に対する正当な報酬の取得に対する障碍となる。他方，市場独占を防止，競争を促進する手段としては，リバース・エンジニアリングは一つの手段である。

従って，コンピュータ業界におけるリバース・エンジニアリングに対する客観的な要請およびこれに対する反発は，社会的利益（同時に競争メーカーも利益を受ける）と先行開発者の利益の対立を反映していると言えよう。

2．著作権法と保護とプログラムの価値の乖離

著作権法は，表現のみを保護し，アイディアまで保護するものではないという点については，世界的なコンセンサスがある。しかし，コンピュータ・プログラムにおいては，真に重要なのは表現ではなく，その機能である。つまり，プログラムの価値は，具体的なコードの記述からコード記述の基礎となる機能設計，詳細設計，アルゴリズム等に移っている。そして，知的活動としては，アイディアの部分に近いほど高度な活動が行うことが可能となる。そのため，これらの要素の保護を求める要求は絶えず存在する。リバース・エンジニアリングを禁止すべきとする見解は，その一つの現れである。

このような乖離は，技術製品であるコンピュータ・プログラムの保護のための基本的な法制度として，従来，技術保護法の性格を持っていなかった著作権法が選択されたことからもたらされた必然的な結果であろう。他方，著作権法がプログラムの保護のための基本的な法制度として選択された以上，著作権において，このような乖離につき，何らかの特別（伝統的著作物と異なる例外的）な対処をするべきと思われる。

第3節　リバース・エンジニアリングのための複製行為の適法性

リバース・エンジニアリングのための複製または翻案行為が適法であるか否かは，単に著作権法の形式的な解釈だけでは解決できない。これは，情報の自由の原則と知的財産権法的関係，更にアイディアの自由と表現の保護との関係，プログラム著作物の特殊性，及び複製又は翻案行為を認めることが著作権者に与える影響などを検討した上で，考えるべき問題である。

1．情報の自由の原則と知的財産権法

知的財産の問題を考える上で，情報の自由，及びその中心としてのアイディアの自由の原則は，問題の核心となる概念と言える。民主主義の社会においては，思想の自由・表現の自由・学問の自由が不可欠であるが，これらの自由の基礎となるものの一つは，情報の自由である。

しかし，情報の自由も他の自由と同様，無制限に貫徹されるものではない。情報には，個人の活動，主として知的活動によって生み出されるものがあり，情報の自由を貫徹することはこのような知的活動を行う意欲を減退させる恐れがある。そこで，知的活動を奨励するために情報の自由を制限し，知的活動を行う者に一定期間又は一定範囲の情報の独占権を与える制度としての知的財産権制度が社会に認められたのである。

情報の自由，即ち情報取得の自由，情報利用の自由，情報表現の自由に対して，情報の独占も，情報取得の独占，情報利用の独占および情報表現の独占が分けられている。これらの独占の様態によって，知的財産権法は色々な分野に分けられる。

例えば，特許法は，アイディアの利用に着目し，これに一定範囲の独占権を与えるが，他面，アイディアの取得，表現は第三者が自由に行うことが

できる。これに対して，著作権法は，アイディアの表現に着目し，アイディアの特別な存在形態としての具体的表現についての独占権を与えている。

つまり，情報の自由およびその中心としてのアイディアの自由の原則が存在し，これに対する例外として，知的活動の奨励のため法制度として著作権法があり，表現という形で知的活動の一部を保護しているということである。従って，著作権法におけるアイディアの自由と表現の保護の関係は，同一レベルで対立する原則として捉えられるものではなく，基本的な原則としてアイディアの自由があり，表現の保護はアイディアの自由の原則に抵触しない範囲で認められるものである。表現とアイディアとが一体化した場合，即ち，アイディアを表現するのに限られた方法しかない場合，または実際的もしくは利用上の考慮からある特定の方法でしか表現できない場合には，保護を与えないとするアメリカ著作権法におけるマージ理論もこの原則から導き出されたものである。また，この原則からすると，学術技芸の進歩という目的に合致し，その目的達成にインセンティブを与える限り，独占権が付与されるが，そうでなければ，例えば自己の独占を確保し，市場から競争を排除するために著作権を行使することは，権利濫用としてその独占権を制限すべきである。

従って，コンピュータ・プログラムの法的保護が著作権法の枠内に取り込まれた以上，プログラムのリバース・エンジニアリングは，アイディア取得の自由の原則から，当然認められるものである。リバース・エンジニアリングを行う過程に複製又は翻案が伴った場合であっても，著作権はアイディア自由の原則に抵触しない範囲で例外的に認められるものに過ぎず，何らの違法でもないことになる。

2．プログラム著作物の特殊性

小説，映画などの伝統的著作物と異なり，プログラム著作物は，有体物である記憶装置に格納され，コンピュータを作動させるという実用的な機能を持ち，直接読んだり，理解したりすることができない。

プログラムは，多くの場合には，オブジェクト・コードの形で，フロッピー・ディスク，ドラム，ROMなどに格納され，流通している。オブジェクト・コードは，コンピュータが理解できる機械語で書かれたものであるから，人間にとっては，特別な技能を有する少数の者には理解可能であるが，普通の者には理解することができない。プログラムがフロッピー・ディスク等に格納されている時，これらを見るだけでは，その内容を理解することはできない。場合によっては，プログラムの通常の使用により，ある程度プログラム自体を理解することができるが（いわゆるブラックボックスアナリシスの分析手段），しかし，この手段は極めて限定的なものであるので，プログラム自体を理解することにとっては，十分ではない。従って，プログラムを理解するためには，それをダンプして，プログラム・リストを作成し解析することが必要となる。その過程においては，対象プログラムの複製又は翻案などの行為が不可欠に伴うことになる。

プログラム著作物のこの特殊性を考えると，複製又は翻案などの行為が認められないとすれば，リバース・エンジニアリングは事実上できなくなり，上述したアイディアの取得自由の原則が存在しなくなる。

3．複製を許容することの著作権者に与える影響

リバース・エンジニアリングのための複製の目的は，プログラムに対する研究及び理解であり，それ自体は，市場における著作物の著作者として享受すべき経済的利益が損なわれることはない。

このような複製行為を許容するとすれば，「既存プログラムの連続的な変更により外形的表現を偽装した実質的な海賊版プログラムの作成に利用されることとなる」という見解がある(60)。しかし，複製行為が伴うリバース・エンジニアリング（特にデコンパイレーション）は極めて困難な作業であるので，一般に海賊版プログラムの作成に利用されるとは考えられないであろう。そして，たと

え海賊版プログラムの作成に利用されたとしても，それはリバース・エンジニアリングのための複製を許容することと関係ないと言える。何故なら，リバース・エンジニアリングの複製を認めなくても，海賊行為に利用される恐れは常に存在するからである。

また，「調査・解析は，プログラムの創作に必要な費用及び時間を回避し対象プログラムの著作権者に対する商業的優位を得るために用いられるものであるから，それに伴う複製または翻案を許容すると，著作物の通常の利用を妨げ，著作者の正当な利益を不当に害することとなる」心配もある(61)。しかし，リバース・エンジニアリング自体も，費用及び時間がかかる非効率的な作業である。そして，リバース・エンジニアリングのための複製の目的は，前述した通り，あくまでもプログラムの内容，構造などについての研究，理解であり，その研究した結果の頒布に繋がらない限り，あるいはこの結果によって直接競合するプログラムを作成しない限り，著作物の通常の利用及び著作者の正当な利益に不利な影響をもたらさないであろう。そして，その結果を利用すれば，場合によって，既存プログラムの著作権者にとって潜在的な経済の影響があるのは否定できないが，リバース・エンジニアリングのための複製及びリバース・エンジニアリングそれ自体の目的を限定すればよいし，そして，このような潜在的な経済の影響はあくまでも，リバース・エンジニアリングの結果の利用によりもたらされたものであり，リバース・エンジニアリングそれ自体および複製行為とは関係ないであろう。

第4節　リバース・エンジニアリングのための複製行為を制限すべき根拠

リバース・エンジニアリングのための複製または翻案行為を著作権の例外として認めるべきである一方，これらの複製または翻案行為に対して，色々な限定をすべきと思われる。前節の問題と同じく，この問題は単に著作権法の形式的な解釈だけで解決できる問題でなく，次の幾つかの要素を考えなければならない。

1．プログラムの保護と著作権制度

本章第2節で述べたように著作権法の保護とプログラムの価値が乖離している。その原因は，プログラムにおいて真に重要なのは表現ではなく，その機能である。つまり，プログラムの保護の本質的な要請は，表現より機能である。

そのため現行の著作権制度は，コンピュータ・プログラムの基本的法制度として適当でなく，新しい法律を作ってプログラムを保護するか，或いは現行の著作権制度を変容すべきかについて，数年にわたって検討が続けられてきたが，しかし最近はこのような議論は前より沈静化している。その根本的原因は，EC指令及びアメリカの公正使用のように，プログラムの保護が著作権制度の例外として特別の保護を与えられたからである(62)。

つまり，著作権制度の例外的規定により，プログラムの機能をある程度保護することが期待されていると言えよう。

リバース・エンジニアリングのための複製または翻案行為については，本章第3節で分析したように，著作権制度の本来の趣旨（情報・アイディアの自由の原則に対する例外）に照らして，このような複製または翻案行為は認められるべきであるが，しかし，プログラムの保護の本質的な要請——機能の保護を全く無視してはならない。そうでなければ，既存プログラムの開発のインセンティブを減退させることになり，結局著作権制度の目的——文化の発展に寄与することに反することになってしまう。

従って，著作権制度の本来の趣旨とプログラムの保護の本質的な要請の間の，バランスを取るために，適法とするリバース・エンジニアリングのための複製行為に一定の限定を加えるべきであろう。

また，著作権制度の本来の趣旨により，このような複製または翻案行為が認められるべきであるとしても，それはあくまでも現行の著作権制度の中の例外的規定であり，基本的原則ではない。基

本的原則に比べると，例外的規定は必要な最小限の範囲に限定すべきであるのは，当然のことである。これに反すれば，著作権制度自体が混乱することになる。

2．プログラムの保護と著作権法の保護範囲

アイディアの表現のみを保護し，アイディア自体を保護しないことは，著作権法の基本的立場である。これによって，著作権法の保護範囲は限定される。しかし，翻案という概念を著作権制度に導入するのは，表現形式を外面的形式と内面的形式とに分け，外面的形式は異なっていても内面的形式が同一であれば翻案に該当する。そうすると，「翻案はアイディア保護に一歩踏み込んだものであることは否定しえないし，そのことは，著作権保護一般については好ましいことである」(63)。この内面的形式という考え方を前提にすると，プログラムのような機能的著作物の場合においても，「ある程度表現それ自体を超えた部分にまで保護範囲を広げざるをえない」(64)。

つまり，著作権の保護それ自体は，具体的に記述された表現にとどまらず，より抽象的なレベルのものまで保護していると言える。

そして，プログラムのような機能的なものを著作権法で保護する場合は，「形式的には表現を保護していると言っても，現実には機能保護の側面を否定することはできない。」(65)。

従って，著作権法それ自体が，プログラムの機能部分にまで保護範囲を広げるのは現実であり，問題は，ただプログラムの機能部分のどこまでに保護範囲を広げることができるかということである。これによって，プログラムのある機能を保護するため，リバース・エンジニアリングのための複製行為に対して許容される範囲を限定することは当然なことであろう。

3．社会的利益と先行開発者の利益とのバランス

本章第2節で考察したように，コンピュータ業界においてはリバース・エンジニアリングの合法化を客観的に要求している。リバース・エンジニアリングは，個々の企業にとって将来の技術開発のために重要であるということだけではなく，社会全体にとっても，無用な二重投資を避けるために重要である。他方，既存プログラムの開発者は，投資の回収および開発による利益を確保するために，リバース・エンジニアリングに強く反対した。これは，社会的利益（同時に競争メーカーも利益を受ける）と先行開発者の利益の対立を反映しているのは明らかである。このような社会的利益と先行開発者の利益とのバランスを取ることを考えると，リバース・エンジニアリングのための複製行為を制限すべきである。

第5節　リバース・エンジニアリングのための複製行為を制限すべき要素

EC指令とアメリカの公正使用を参考すると，少なくとも，次の要素について制限するべきと考える。

1．主体の限定

リバース・エンジニアリングを行う主体は，対象プログラムの複製物の合法的所持者でなければならない。

2．最終目的の限定

リバース・エンジニアリングは，あくまでも中間的行為であり，最終目的によって，その性質が変わる場合がある。例えば，最終の目的が海賊版プログラムを作成することである場合には，リバース・エンジニアリングそれ自体は合法的研究であっても，その結果の利用は，著作権法に反するものであるので，結局リバース・エンジニアリング自体も認められないであろう。従って，著作権法の趣旨及び目的に照らして，リバース・エンジニアリングの最終目的に対して，制限を加えるべきである。

具体的にどのような目的が適法であるかについては，現時点では，判例が少なく，リバース・エンジニアリングに係る多くの概念，例えば競合，互換，インタオペラビリティ，標準化などにつき

法的に確立された定義がないので，網羅的に規定することは困難であるが，次に掲げるものの全部または一部とすることが考えられる。

①純粋な学術研究のため。

②著作権侵害を発見するため。

③エラー修正のため。

④プログラムの間，またはプログラムとハードウェアの間とのインタオペラビリティを達成するため。

⑤著作権法の目的に適い，且つ対象プログラムの著作権者に不利な影響を与えない等のその他の目的。

3．行為様態の限定

限定された目的を達成するために，リバース・エンジニアリングの過程における不可欠な限度の複製または翻案，翻訳に限定すべきである。

4．行為対象の範囲の限定

限定された目的を達成するために，必要の情報，および対象プログラムの必要な部分のみに限定すべきである。

5．取得された情報の利用の限定

限定された目的の達成するためのみに利用できることに限定すべきである。

第6節　結論

以上の検討によって，プログラムのリバース・エンジニアリングの著作権保護についての基本的な考え方としては，著作権法において，著作権に対する例外として，一定の範囲でリバース・エンジニアリングを認めるべきである。

具体的な法的対応は，EC指令の取った方法とアメリカの公正使用における方法の二つのがある。シビル・ローのアプローチを反映したEC指令は，リバース・エンジニアリングを行う者の権利につき明確な規定を作る一方，コモン・ローのアプローチを反映したアメリカの公正使用は，ケース・バイ・ケースで，強力な公正使用の抗弁を提供している。具体的内容としては，EC指令の規定は限定的且つ明確であり，公正使用の法理は技術の変化に対する対応性が強いである。

日本の場合は，上述したプログラムのリバース・エンジニアリングの著作権保護についての基本的な考え方に沿って，シビル・ローのアプローチを反映したEC指令の取り扱い方を取るべきと思われる。

アメリカの公正使用の法理は，色々な利点特に技術の変化に対する優れた対応性があるが，しかし，それはあくまでもコモン・ローの法システムの国々に適うものであり，日本を含むシビル・ローの法システムの国々に直接適合するかどうかは，慎重に検討する必要があるであろう。公正使用の法理は，個々の事案毎にケース・バイ・ケースで判断する指針であるので，全ての事案に機械的に適用可能な基準はありえず，常に曖昧さが残ることはやむを得ない。しかし，シビル・ローの法システムの場合には，個別的事案の解決の集積が法理の柔軟性に役に立つとしても，それと同時に裁判所の判断が不安定になることを一定範囲で抑制する必要がある。また，プログラムのリバース・エンジニアリングの保護は現在既に重要な問題となっており，明確な基準が不存在であることは，結果の予測を困難にし，企業行動を不当に制約することになる。法的安定性のためには，裁判規範としてまたは行為規範としての有効な基準が確立されるべきであろう。

従って，判例が未だない，しかも法規定が必要である現在の日本にとっては，種々の関連要素およびケースを考慮し，その上に，公正な基準を発見し，詳細な規定を作るべきと考える。

そしてその際には，EC指令の仕方を取ると同時に，その欠点（目的の狭い限定性）を考慮し，アメリカの公正使用のような一般的な限定条項（著作権法の目的に適うもの）も設けるべきであろう。

注

（1）　ディスアセンブルは，機械語で記述するプログラムをアセンブラ言語で記述するオブジェクト・

コードに変換するツールであり，デコンパイレーション（デコンパル）は，オブジェクト・コードを高級言語で記述し，人間が理解できるソース・コードに変換するツールである。市場で販売されているプログラムは，機械語で書かれたものがほんどであるので，現在，プログラムをデコンパイレーションするいずれの作業も，まず，ディスアセンブルから始められる。そして，デコンパイレーションという用語は，政策論争においては，ディスアセンブルを含めて使われる場合もある。例えば，「デコンパイレーション」は，しばしば，「機械可読」のコードを「人間可読」のコードに変換するために利用される技法一切と見なされる。但しプログラムが「ディスアセンブル」されようと，「デコンパイレーション」されようと，法律上の問題は同じである。従って本稿では，以下単に「デコンパイレーション」という用語を使う。

(2) Council Directive of 14 May 1991 on the legal protection of computer programs, 91/250/EEC

(3) Completing the Internal Market, COM (85) 310

(4) Green Paper on Copyright and the Challenge of Technology-Copyright Issues Requiring Immediate Action, COM (88) 172 final (Brussel, 7. Jun. 1988)

(5) Proposal for a Council Directive on the Legal Protection of Computer Programs, COM (88) 816 final-SYN 183

(6) Amended Proposal for a Council Directive on the Legal Protection of Computer Programs, COM (90) 509

(7) Bridget Czarnota & Robert J. Hart, Legal Protection of Computer Programs in Europe--A Guide to the EC Software Directive (1991), 173, 174頁。

(8) 同160，161頁。

(9) Official Journal of the European Communities, No. C91 (1989. 4. 12.), 16頁参照。

(10) M. Lehmann & C.F. Tapper, A Handbook of European Software Law, 1993, 47～52頁。

(11) 同上。

(12) 同54頁。

(13) 同58～59頁。

(14) 同68，71頁。

(15) 高橋三雄『コンピュータ用語辞典』(1991年) 428頁参照。

(16) 前掲，注(7)69頁。

(17) 財団法人ソフトウェア情報センター，第3回『コンピュータ・ソフトウェアの法的保護に関する国際シンポジウム議事録』，平成4年7月，297頁。

(18) 前掲，注(10)66，72頁。

(19) 同73頁。

(20) EC委員会，Twentieth Report on Competition Policy (1991), CM 50 91 410 Part II,

(21) 同78頁参照。

(22) 同17～18頁。

(23) 前掲，注(7)84～86頁参照。

(24) 前掲，注(10)77頁の注^{2}。

(25) 同77頁。

(26) 同80頁。

(27) 同20頁。

(28) 山中伸一，「コンピュータ・プログラムのリバース・エンジニアリングについて」(下)，ジュリスト No. 1020 (1993.4.1.)，143頁。

(29) 前掲，注(10)80頁参照。

(30) 中山信弘，『ソフトウェアの法的保護』(新版)，100～105頁。有斐閣，1988年

(31) 1989年ニューヨーク市弁護士協会の「リバース・エンジニアリングと知的財産権法」というレポートでは，七つのリバース・エンジニアリングを例にとって，EC指令の規定をこれらの例に適用し検討したところ，これらの七つの例のいずれに対しても，明確な結論が導かれるとしている。このことについて，Robert J. Hartは「Legal Protection of Computer Programs: Decompilation, Reverse Engineering and the EC Directive」(The International Computer Lawyer, V. 1, No. 3, February 1993. 21頁参照) という論文で，EC指令第5条，第6条に規定されているリバース・エンジニアリングに関する規定がかなり明確である，と評価してい

る。

(32) 前掲，注(30)20頁。

(33) Segar Enterprises Ltd. v. Accolade Inc., 977 F. 2d 1510 (9th Cir. 1992)

(34) Atari Games Corp. v. Nintendo of America Inc., 975 F. 2d 832 (Fed. Cir. 1992)

(35) この問題は本稿で取り上げたリバース・エンジニアリングの合法性の問題と直接関係がないため，ここでは検討しない。

(36) Sega Enterprises Ltd. v. Accolade Inc., 785 F. Supp. 1392 (N.D. Cal. 1992), at 1397～1399

(37) 10 U.S.P.Q. 2d 1177 (N.D. Cal. 1989)

(38) 前掲，注(36)，at 1527

(39) Atari Games Corp. v. Nintendo of America Inc.,
1993 U.S. Dist. LEXIS 8183 (April 15, 1993)
1993 U.S. Dist. LEXIS 6786 (May 17, 1993)

(40) 1993 U.S. Dist. LEXIS 8183 (April 15, 1993), at 20.

(41) 同，21～25頁。

(42) 1993 U.S. Dist. LEXIS 6786 (May 17, 1993), at 5.

(43) Richard H. Stern, Reverse Engineering for Future Compatibility: Atari v. Nintendo, (4 European Intellectual Property Review 178～180 1994

(44) Last Frontier Conference Report on Copyright Protection of Computer Software, (Jurismetrics, vol. 30, No. 1, Fall 1989, at 24-25.)

(45) W. Patry, The Fair Use Privilege in Copyright Law, (at 399-401, 1985)

(46) Arthur R. Miller, Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?, (106 Harvard Law Review 1014-1032, 1993.)

(47) Dennis S. Karjala, Copyright Protection of Computer Software, Reverse Engineering, and Professor Miller, (19 University of Dayton Law Review 975, 1010-1018, 1994)

(48) Harper & Row, Publishers, Inc. v. National Enterprises, 471 U.S. 539 (1985)参照

(49) House of Representatives, Rep. No. 1476, 94th Congress, 2d Session 65 (1976)

(50) 前掲，注(48)

(51) Marcus v. Crowley, 695 F. 2d 1171 (9th Cir. 1983)

(52) Italian Book Corp. v. American Broadcasting Cos., 458 F Supp. 65 (S.D.N.Y. 1978)

(53) Salinger v. Random House, Inc, 811 F. 2d 90 (2d Cir. 1987)

(54) William Party, The Fair Use Privilege in Copyright Law (Washington, DC: The Bureau of National Affairs, 1985) 449-450

(55) Universal City Studios, Inc v. Sony Corp. of America, 480 F. Supp. 429 (D.C. Cal. 1979), rev'd, 659F 2d 963 (9th Cir. 1981), rev'd, 464 U.S. 417 (1984)

(56) Stephen B. Maebius,「公正使用（フェア・ユース）の新しい解釈」，AIPPI (1994) Vol. 39 No. 5, 242-244頁参照。

(57) U.S. Congress, Office of Technology Assessment, Finding a Balance: Computer Software, Intellectual Property, and the Challenge of Technological Change, OTA-TCT-527 (Washington, DC: U.S. Government Printing Office, May 1992)

(58) 中山信弘監修『ソフトウェアと知的財産権』，39-42頁。㈶ソフトウェア情報センター翻訳，日本評論社，1993年。

(59) マイクロソフト対秀和事件（東京地裁昭和62年1月30日判決）が，日本におけるリバース・エンジニアリングに関する最初の判決と解するものはある(D.S.カージャラ，椙山敬士，『〔日本―アメリカ〕コンピュータ・著作権法』33頁、，日本評論社1989.11.30.)が，しかしこの判決は「単に他人のオブジェクト・コードを逆アッセンブルして出版する行為は著作権侵害である，という点を明らかにしただけのものであり」，「リバース・エンジニアリングの可否については白紙であると考えるべき」という

反対の解釈もある(中山信弘,「基本プログラムを逆アッセンブルして出版した事件」,判例タイムズ,No. 634, 1987.7.15. 46-49頁)。筆者は後者に賛成する。

(60) 本章第1節参照。

(61) 同上。

(62) 前掲,注(10),214頁参照。

(63) 中山信弘『ソフトウェアの法的保護』(新版),110頁。有斐閣,1988年。

(64) 同上。

(65) 同上,21頁。

欧文の題名

(1) Council Directive of 14 May 1991 on the legal protection of computer programs, 91/250/EEC

(2) Completing the Internal Market, COM (85) 310

(3) Green Paper on Copyright and the Challenge of Technology-Copyright Issues Requiring Immediate Action, COM (88) 172 final (Brussel, 7. Jun. 1988)

(4) Proposal for a Council Directive on the Legal Protection of Computer Programs, COM (88) 816 final-SYN 183

(5) Amended Proposal for a Council Directive on the Legal Protection of Computer Programs, COM (90) 509

(6) Bridget Czarnota & Robert J. Hart, Legal Protection of Computer Programs in Europe--A Guide to the EC Software Directive (1991), 173, 174頁。

(7) Official Journal of the European Communities, No. C91 (1989. 4. 12.),

(8) M. Lehmann & C.F. Tapper, A Handbook of European Software Law, 1993.

(9) The Commission: Twentieth Report on Competition Policy (1991), CM 50 91 410 Part II,

(10) Robert J. Hart, Legal Protection of Computer Programs: Decompilation, Reverse Engineering and the EC Directive, The International Computer Lawyer, V. 1, No. 3, February 1993

(11) sega Enterprises Ltd. v. Accolade Inc., 977 F. 2d 1510 (9th Cir. 1992)

(12) Atari Games Corp. v. Nintendo of America Inc., 975 F. 2d 832 (Fed. Cir. 1992)

(13) Sega Enterprises Ltd. v. Accolade Inc., 785 F. Supp. 1392 (N.D. Cal. 1992),

(14) 10 U.S.P.Q. 2d 1177 (N.D. Cal. 1989)

(15) Atari Games Corp. v. Nintendo of America Inc.,

1993 U.S. Dist. LEXIS 8183 (April 15, 1993)
1993 U.S. Dist. LEXIS 6786 (May 17, 1993)
1993 U.S. Dist. LEXIS 8183 (April 15, 1993)
1993 U.S. Dist. LEXIS 6786 (May 17, 1993)

(16) Richard H. Stern, Reverse Engineering for Future Compatibility: Atari v. Nintendo, (4 European Intellectual Property Review 178～180 1994)

(17) Last Frontier Conference Report on Copyright Protection of Computer Software, Jurismetrics, vol. 30, No. 1, Fall 1989, at 24-25.

(18) W. Patry, The Fair Use Privilege in Copyright Law, at 399-401 (1985)

(19) Arthur R. Miller, Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?, (106 Harvard Law Review 1014-1032, (1993.)

(20) Dennis S. Karjala, Copyright Protection of Computer Software, Reverse Engineering, and Professor Miller, 19 University of Dayton Law Review 975, 1010-1018 (1994)

(21) Harper & Row, Publishers, Inc. v. National Enterprises, 471 U.S. 539 (1985)

(22) House of Representatives, Rep. No. 1476, 94th Congress, 2d Session 65 (1976)

(23) Marcus v. Crowley, 695 F. 2d 1171 (9th Cir. 1983)

(24) Italian Book Corp. v. American Broadcasting Cos., 458 F Supp. 65 (S.D.N.Y. 1978)

(25) Salinger v. Random House, Inc, 811 F. 2d 90 (2d Cir. 1987)

(26) William Party, The Fair Use Privilege in Copyright Law (Washington, DC: The Bureau of National Affairs, 1985) 449-450

(27) Universal City Studios, Inc v. Sony Corp. of America, 480 F. Supp. 429 (D.C. Cal. 1979), rev'd, 659F 2d 963 (9th Cir. 1981), rev'd, 464 U. S. 417 (1984)

(28) Stephen B. Maebius,「公正使用（フェア・ユース）の新しい解釈」, AIPPI (1994) Vol. 39 No. 5, 242-244頁参照。

(29) U.S. Congress, Office of Technology Assessment, Finding a Balance: Computer Software, Intellectual Property, and the Challenge of Technological Change, OTA-TCT-527 (Washington, DC: U.S. Government Printing Office, May 1992)