社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－DPS－128 (19)
2006－GN－ 61 (19)
2006－EIP－ 33 (19)
2006／9／15

権利と責任の観点からみた計算機ソフトウェアの定義に関する一考察

谷口 展郎†

来たるべきユビキタスネットワーク社会において，計算機ソフトウェアの重要性はますます高まると考えられる。しかし，計算機ソフトウェアにまつわる権利と責任の法的取り扱いの現状は，一貫性や均衡を欠いていると言わざるをえない。本稿では，この問題に対する解となる法的枠組みの一試案を示し，該試案の重要な前提の一つである計算機ソフトウェアとは何かという定義を示す。

A Note on a Definition of Computer Software from a Viewpoint of Legal Rights and Liability

Noburou Taniguchi†

In the "Ubiquitous Network Society" of the near future, it is expected that the importance of computer software will become higher and higher. However, we should say that the legal treatment of the rights and liability of computer software lacks both integrity and balance. In this paper, I show a legal framework as one of the solutions to amend this problem, and present a legal definition of "computer software" that the framework assumes.

1 はじめに

近年，社会の情報化が大きく進展し，これに伴って社会における「情報」の重要性に対する認識も高まった。特に，情報の財産的価値をいかに保護するかについては社会的関心が高く，これまでも様々な取り組みが進められてきた。

無体財である情報，とりわけディジタル情報は，物理的な手段のみで保護することが本質的に難しく，法的保護が欠かせない。そこで，ここ二十数年来，プロパテント政策やプロコピーライト政策のような政治的動きも伴いながら，著作権や特許，営業秘密，個人情報などの様々な観点から，情報の法的保護の強化が行われてきた。

中でも近年，社会の情報化の進展と歩みを合わせるように法的保護が強化されてきたのが，計算機ソフトウェア(以下「ソフトウェア」)である。

日本において，明示的にソフトウェアを対象とする法的保護の歴史は，1975年12月に特許庁が発表した「コンピュータ・プログラムの成立性に関する発明についての審査基準(その一)」に遡る。その後，1982年に日立・IBM事件が発生，主にビデオゲームを対象とした不法複製に関する訴訟も相次ぐなか，通産省は産業構造審議会情報産業部会中間答申(1983年12月)でソフトウェアに特化した「プログラム権法(仮称)」の新規立法による保護を，文化庁は著作権審議会第六小委員会中間答申(1984年1月)で著作権法の改正による保護を提言した。この対立は，最終的には，通産省案に対する米国の強い反対もあり，1985年に著作権法が「プログラムの著作物」を取り込むよう改正されるという形で決着した[1][2]。

その後も，パーソナルコンピューター(PC)の普及やインターネットの商用解放などの社会的背景の下，様々な法改正が行われ，現在では著作権法のほか，特許法や不正競争防止法にも，特に「プログラム」(本稿でいうソフトウェア)について記述した条文が存在する。特に特許法では，2002年の改正で，「物」がプログラムを含むことが明記された。近代法体系では長年「物とは有体物」【民法86条】とされてきたことを考えると，これは極めて大きな姿勢転換と言えよう。また直接ソフトウェアを対象とするものではないが，現在，不正競争防止法の営業秘密の保護に関連して，「情報窃盗罪」新設が検討されている[3]。これは従来不可罰とされてきた行為に刑事罰を課すことにつながるため，改正が実際に行われれば，やはり日本の法制史上の転換点の一つとなろう。

このように，近年，ソフトウェアについては，権利保護が強化されてきた。その一方で，ソフトウェアに関する責任に眼を転じると，法的にはほとんど何も変わっていないと考えられる。例えば，責任追及の文脈で，ソフトウェアを対象に明記して記述している法律の条文は，筆者の知る限りほとんどない(少なくとも，「脆弱性」の文脈で「ソフトウェア」について定義し，記述している法令はない[4])。即ち，現在のソフトウェアを取り巻く法的状況は，権利については強化が推進されてきたのに比べて，責任については従来のままという，極めてアンバランスな状況と言えよう。

ソフトウェアが情報化社会において極めて重要な要素であることは明らかだろう。だとすれば，その法的権利だけでなく，法的責任もまた，ソフトウェアの特性を考慮した形で強化されてしかるべきではないだろうか。こうした考えの下，筆者は，その一つの方法として，ソフトウェアの製造物責任を問うことを可能にする法的フレイムワ

†日本電信電話株式会社 NTT サイバースペース研究所
†NTT Cyber Space Laboratories, Nippon Telegraph and Telephone Corporation

ークを提案している[5]。

このフレイムワークは，単にソフトウェアを製造物責任法の対象とするだけではなく，その特性に鑑み，ソースを公開した場合は免責するという仕組みを取り入れている。こうすることで，ソフトウェアの技術的な進歩をできるだけ阻害せず，ソフトウェアの製作者や流通者(以下「製作者等」)の欠陥減少へのインセンティヴを高めることを目指している。

しかし前稿では解決策を述べるに到らなかったいくつかの課題がある。その主要なものの一つが，「ソフトウェア」とは何か，という定義である。これは例えば，「ソフトウェアと料理のレシピはどう違うのか」という問題である。

本稿では，ソフトウェアの権利と責任のバランスを達成する上で重要な，この「ソフトウェアの定義」の問題について考察する。まず次節で，[5]で示した法的フレイムワークについて再度簡潔に説明する。次に3節でさまざまな法律や法案,あるいは法的議論において，ソフトウェアがどのように定義されているかについて調べた結果について述べる。これを受けて4節で筆者のフレイムワークにおけるソフトウェアの定義を示し，これについて議論する。最後に5節でまとめを述べる。

2 ソフトウェア製造物責任の法的フレイムワーク

情報化が進み，生活のあらゆる場面にコンピューターが組み込まれるような〝ubiquitous computing〟社会では，ソフトウェアには，豊富な機能や利便性だけでなく，高い安全性や信頼性が求められる。特にネットワーク化が高度に進展した世界では，一つのソフトウェアの欠陥により引き起こされた問題が，ネットワークを介し瞬く間に広がって大きな被害をもたらす，というリスクは決して小さくない。従って，ソフトウェアの欠陥を減少させ，安全性や信頼性を高めることは，これからますます重要になってくるであろう。

ソフトウェアの欠陥を減らす最良の方法の一つは，ソフトウェア製作者等自らが欠陥を減らそうとするインセンティヴを高めることである。

これには，例えば政府の情報基盤強化税制[6]などの取り組みのように，欠陥の少ないソフトウェアに褒賞を与えることも有効である。この税制改正では，ISO15408認証を受けたソフトウェア等を用いて情報システムを構築すると，税負担が減免される。ISO15408認証では，IT製品・システムの仕様や設計プロセスが所定のセキュリティ基準を満たしているかどうかを評価する[7]。従って，そこで用いられるソフトウェアにも脆弱性への特段の注意が求められるため，当該税制を活用しようとすると，結果的に脆弱性の少ないソフトウェアの開発が行われることになる。

しかし，日々ソフトウェアが新たに開発/公開されるような情報化社会においては，前述のような正のインセンティヴを与える方法のほかに，欠陥の多いソフトウェアの製作者等に，欠陥によって引き起こされた問題への責任を負わせてペナルティを与えることで，欠陥を減らす努力を促すことも必要であろう。

現行法の下でも，ソフトウェア製作者等は，少なくとも民法上の不法行為責任【民法709～724条】を負う。しかし，この不法行為責任を問おうとする場合，製作者等の故意・過失の存在，および，それらが被害を引き起こしたという因果関係の立証を行わなければならない。ソフトウェアが手許に届くまでの製造・流通過程に故意・過失があったことを，一般的な消費者が証明することは難しい。他方，有体製造物については，製造物責任法により，欠陥の存在と欠陥→損害の因果関係の証明だけで，製作者等の故意・過失の存在を証明することなく責任を問うことができる。しかしソフトウェアは無体物であるため，製造物責任法の対象にはなっていない。

さらに，ほぼ全てのソフトウェアは，ライセンス契約で「無保証」をうたっている。パッケージソフトウェアのいわゆる「シュリンクラップ契約」や，ダウンロード提供されるソフトウェアの利用許諾に同意するボタンをクリックすることで行われる契約(「クリックラップ契約」と呼ばれることがある)の有効性については，疑義も多いと言われている。しかし，多くのソフトウェア製作者等が，こうした利用許諾条件によりソフトウェア欠陥の責任から逃れられる／逃れられるようにしておくべきと考えていることは，明らかであろう。

ソフトウェアの欠陥に対する責任を問うべきでないとする人々が主張する理由の一つに，「欠陥責任追及は，自由なソフトウェアの開発・流通を抑制し，技術の進歩を妨げる」というものがある。ソフトウェアは，一般的な有体物の工業製品と異なり，大規模な生産設備や流通設備を必要としない。そのため，こうした設備を持てない一般の人々も含めて，誰もが容易に製作者になれる。これは情報財であるソフトウェアならではの利点であり，これにより活発な開発・流通が促され，技術革新も推進される。しかし，もしソフトウェア製作者等に対して欠陥責任の追及が行われれば，これら製作者等が萎縮してしまい，ソフトウェアの開発・流通は抑制されて，技術進歩も妨げられるだろう，というのが，その主張の骨子である。

これに対し筆者は，ソフトウェアの開発・流通をできるだけ阻害せずに，製作者等に欠陥を減らすよう促す方法の一つとして，1) ソフトウェアを製造物責任法の対象に含める，2) 但しソフトウェアのソースコードが公開(以下「ソース公開」)された場合は製造物責任を免責する，という法的フレイムワークを提案している[5]。

このフレイムワークの下では，まず，ソフトウェアを製造物責任法の対象に含めることで，ソフトウェア欠陥について製作者等の責任追及のハードルが下がるため，製作者等が欠陥を減らすようより真剣に努力することが期待できる。一方で，ソース公開された場合は製造物責任を免責することで，それらのソフトウェアの開発や流通は

阻害されない。

前項では，本フレイムワークの提案を行った後，その得失について様々な観点からの議論を示したが，扱いきれなかった課題も幾つかあった。その重要な課題の一つが，「ソフトウェアとは何か」という定義である。次節以降では，この課題を取り扱う。

3 既存法等におけるソフトウェアの定義

ソフトウェアの製造物責任について反対する主張では，前節で述べた「技術進歩の阻害」以外によく使われる論理の一つに，以下に述べるようなものがある。

- ソフトウェアは料理のレシピと同じように，何かを実施する手順を記述したものである
- 従ってソフトウェアの製造物責任が問えるならば，レシピに従って料理を作った／食べたことにより損害を受けた人は，レシピの作者の製造物責任を問えることになる
- しかし従来法では，そのようなケースで，レシピ作者の製造物責任は問えないというのが定説である
- 従ってソフトウェアの製造物責任を問うことは間違いである

ソフトウェアの製造物責任について議論する際は，この問題に答えるために，「ソフトウェアと料理のレシピはどう違うのか(あるいは，違わないとすれば，レシピの作者の製造物責任を問う合理的な説明は何か)」を示す必要がある。これは即ち，ソフトウェアを法的にどう定義するかという問題である。

この問題については，これまで，ソフトウェアの権利を確定していく過程で，さまざまな法律や法案,法学的議論において，定義が示されている。以下では，それらの幾つかについて見ていくこととする。なお，特に断らない限り，法名は国内法を指す。

■著作権法

日本の法体系では，ソフトウェアはまず著作権法により保護される。著作権法では，ソフトウェアという語は用いずに「プログラム」という語を用い，「電子計算機を機能させて一の結果を得ることができるようにこれに対する指令を組合せたものとして表現したものをいう」【2条1項10の2号】と定義している。

また【10条1項9号】で例示として「プログラムの著作物」を挙げるとともに，【10条3項】では，その例外として，「プログラム言語」:プログラムを表現する手段としての文字その他の記号およびその体系をいう【10条3項1号】，「規約」:特定のプログラムにおける前号のプログラム言語の用法についての特別の約束をいう【10条3項2号】，「解法」:プログラムにおける電子計算機に対する指令の組合せの方法をいう【10条3項3号】，を挙げ除外している。

これらの例外のうち，「プログラム言語」については特に説明は不要であろう。残る二つについては，「規約」は(プログラミング)インターフェイスやプロトコル等を指し，「解法」はアルゴリズムを指すと解釈されている[2]。

■特許法

日本の特許法では，【2条3項1号】で，「物」の発明について，「物」は「プログラム等を含む」と但し書きされている。そして，【2条4項】で，「プログラム等」を，「プログラム」(電子計算機に対する指令であつて，一の結果を得ることができるように組み合わされたものをいう)および「その他電子計算機による処理の用に供する情報であってプログラムに準ずるものをいう」と定めている。なお「プログラムに準ずるもの」についての定義はないが，特許庁では「構造を有するデータ」を例示している[8]。即ち，特許法ではプログラムだけでなくデータ構造も特許対象となっている。

さらに，特許法そのものではないが，実効上法律に近い効力をもつ公文書として，特許庁の審査運用指針がある。ここにおいては，「特定技術分野の審査の運用指針」[9]の中で，「第1章 コンピューター・ソフトウェア関連発明」として，ソフトウェアを扱っているが，ソフトウェア関連用語の定義も示されているので，以下にその一部を示す。

- ソフトウェア…コンピュータの動作に関するプログラムをいう
- プログラム…コンピュータによる処理に適した命令の順番づけられた列からなるもの(ただしプログラムリスト--プログラムの紙への印刷,画面への表示などによる提示そのもの--を除く)
- 手順…所定の目的を達成するため，時系列的につながった一連の処理又は操作
- データ構造…データ要素間の相互関係で表される，データの有する論理構造
- ハードウエア資源…処理，操作，又は機能実現に用いられる物理的装置又は物理的要素

上述の審査基準では，「プログラム」の定義の表現が特許法と異なるが，これについて特許庁は[8]で「技術的意味において実質的に異なるところはない」との解釈を示している。

■プログラム権法(仮称)[2]

日本ではかつて，ソフトウェア保護を著作権法により行うことが確定する前に，ソフトウェアの特性に着目し，著作権法とも特許法とも異なる特別立法でその権利を保護しようという動きがあった。

これが1983年12月の産業構造審議会情報産業部会中間答申に草案として示されている「プログラム権法(仮称)」である。最終的にこの案は，当時既に著作権法でソフトウェアを保護していた米国からの圧力によって葬り去られ，日本でも著作権法による保護を中心とすることになったが，本案はソフトウェアの(それまでの著作物と

は異なる)特性を強く意識している点で非常に重要な参考資料となるため，敢えてここに挙げる次第である。

この答申ではまず，ソフトウェアを，「コンピュータの利用技術であり，その意味ではコンピュータ利用に関する全てのもの〔ソース・プログラム及びオブジェクト・プログラム(以下「プログラム」という。)，フローチャート，マニュアル，アイデア，アルゴリズム等〕を含むと解される〕と定義している。

その上で，保護対象を上述のうち「プログラム」に限定した「プログラム権法」の骨子案を示している。該骨子案では，「プログラム」を「コンピュータにある一定の機能を果させるための一連の指令の組合せ」と定義している。

なお，この特別立法を必要とする理由として，本答申では「第二章　ソフトウェアの特質と法的保護の基本的視点」で，以下のように述べている。

まずソフトウェアの特質として，

1) 計算機で使用されてはじめて価値が発揮される
2) 開発に多額の投資と労力を要するにも関わらず簡単かつ安価に完全なコピーを作り，使うことができる
3) 既存ソフトウェアの利用／改良(バージョンアップ)により高度化され，提供後も保守が必要
4) 同じ仕事をさせる(アルゴリズム等が同じ)プログラムは，独立に作成されても類似したものになりやすいため，同一性/類似性の判断が難しい
5) 技術先端的な商品なので，陳腐化が早く，また今後作成過程(機械によるプログラムの作成など)や形態の変化が予想される

を挙げている。

また法的保護の基本的視点として，

1) コピー容易性やバージョンアップによる高度化，提供後の保守の必要性などソフトウェアの特質を踏まえた取引の実態に即している
2) ソフトウェアが経済財であることを考慮して，権利の保護と利用の促進を二大目的とする
3) 既存ソフトウェアの改良などを通じて高度化するソフトウェアの機能向上を促進する
4) ユーザーの利益への配慮，特に内容(品質)表示が不十分，情報の不足，保守・補修体制の不備などの問題点を考慮する
5) プログラムを作成するプログラムや，プログラム・データベースからのプログラムの検索・取得・作成など，技術進歩に対応しうる
6) 国際性を考慮する

を挙げ，これらの視点に立った制度の構築が必要と述べている。

前段に示したとおり，本答申はユーザーの利益にも配慮しており，プログラム権法(仮称)にも，「(7)ユーザー保護」という項目が設けられている。そこでは，

- 通産大臣がプログラムの取引に関し表示内容等指針を示す
- プログラムの販売者は該指針に基づき一定の内容表示等の義務を負う
- プログラム作成者が保守義務を果たせなくなった場合に備えソースプログラムを登録機関に寄託する制度を設ける

と規定されている。

この規定は，近年のオープンソース運動を受けた鶴田らの研究[10]に通じるものがある。オープンソースとソフトウェアの法的位置づけを論じたその研究の中では，ソフトウェア産業の現状についての考察が示されたうえで，ユーザー保護の枠組の在り方について，プロプライエタリなソフトウェアでもユーザーがソフトウェアをチェック可能にする，脆弱性情報等についてはパッケージベンダーの権利を制限する，サポートの終了したソフトウェアについてはソースコードをユーザーに引き渡す，などの提案が行われている。

このように，プログラム権法(仮称)においては，「オープンソース」という概念こそ無かったものの，ソフトウェア製作者等の権利だけでなく責任についても一定の配慮が行われていたことがわかる。

■コンピューター・ソフトウェアの保護に関するモデル規定(WIPO)[2][1]

プログラム権法(仮称)に遡ること5年前の1978年，WIPOはソフトウェア保護のモデル規定を発表した。

このモデル規定では，ソフトウェア関連の定義として，以下の定義が示されている。

- 【1条(i)】「コンピュータ・プログラム」とは，機械が読み取ることができる媒体に収納されたときに，情報処理能力を有する機械に特定の機能，任務又は結果を指示させ，履行させ，又は達成させることができる一連の命令をいう
- 【1条(ii)】「プログラム記述」とは，言語，図表その他の形式による全体の手続の表示であって，対応するコンピュータ・プログラムを構成する一連の命令を決定するに十分なほど詳細なものをいう
- 【1条(iii)】「補助資料」とは，コンピュータ/プログラムの理解又は適用を助けるために創作された問題記述，使用者用説明書等の資料(コンピュータ・プログラム又はプログラム記述以外のもの)をいう
- 【1条(iv)】「コンピュータ・ソフトウェア」とは，(i)から(iii)までに掲げる一又は二以上のものをいう

また【4条】には，「この法律に基づく権利はコンピュータ・ソフトウェアの基礎となる概念には及ばない」との記述があり，ソフトウェアへの保護がアルゴリズムまで及ばないことが示されている。

これらの記述から，当モデル規定が通産省答申にも大きな影響を与えていることがうかがえる。

---

[1] ここでは情報入手の都合上，オリジナルではなく[2]に記載された訳文に沿って論じる。

■UCITA(米国)

UCITA (Uniform Computer Information Transactions Act) は，米国の統一州法典の一つである。UCITAはもともと，より大きなUCC (Uniform Commercial Code，統一商法典) の一部として検討されていたが，共同で策定を行う NCCUSL (National Conference of Commissioners on Uniform State Laws) と ALI (American Law Institute) という2つの団体のうち，法案があまりにも業界寄りで消費者の権利が損なわれる，との理由でALIが離脱したため，同団体の賛同を必要としない独立した州法典として策定されるに到った[11]。

前述のように，UCITAはソフトウェア業界寄りの内容を持ち，ソフトウェア製作者等に極めて大きな裁量を認めて，事実上彼等があらゆる法的責任から免れることを可能にするものと言われている。このため，各方面から反対が相次ぎ，州レベルでの採用もなかなか進まなかった。2004年3月には，州レベルでの採用が2州に留まる状況の中，NCCUSLは採用推進を停止した[12]。

UCITAでは，ソフトウェアという用語ではなく，"Computer Information" という言葉を用いている。具体的には【Section102(a)(10)】で，"Computer Information" を，"information in electronic form which is obtained from or through the use of a computer or which is in a form capable of being processed by a computer. The term includes a copy of the information and any documentation or packaging associated with the copy" を意味するものと定義している。これは，ソフトウェアのみならず，ディジタル形式で提供される書物,音楽,映画なども含まれると解される[13]。

その上で，【Section102(a)(12)】において，"Computer Program" を，"a set of statements or instructions to be used directly or indirectly in a computer to bring about a certain result. The term does not include separately identifiable informational content" を意味するものと定義している。直訳すると「直接又は間接にコンピュータ一内で用いられて，何らかの結果をもたらす命令文あるいは命令の組。但し分離して識別可能な情報コンテンツを含まない」となる。

4 製造物責任フレイムワークにおけるソフトウェアの定義

これまで述べてきた議論を鑑みた上で，筆者は，ソフトウェアを，「その時点で利用可能なコンピューティング環境で実行可能，あるいは事実上実行可能であり，実行によって役務を提供するような情報」と定義する。

本定義の特徴は，「(事実上の)計算機可実行性」である。ソフトウェアが実行可能であるためには，まず「コンピューターに読み込み，または事実上読み込み可能(計算機可読)」であることが必要である。ここで，「事実上読み込み可能」とは，直接計算機可読な状態にはなくても，容易に計算機可読な形に変換できることを指す。例えば，紙に印刷されたソースコードを，直接コンピューターで読み込むことは不可能だが，スキャナーとOCRを用いれば，ソースコードとしてコンピューターに取り込むことは容易である。

また，ソフトウェアを「実行する」ということを，ここでは，「何らかの役務を提供する」ことと定義する。例えば，JPEGの画像データは，計算機可読ではあるが，読みこんでそのまま出力されるだけであり，何も役務＝他人のために行う労務やサービス(三省堂『大辞林』第二版より)は提供しない。

さらに，「事実上実行可能」とは，直接実行可能な状態にはなくても，容易に実行可能＝役務提供可能な状態に変換できることをさす。この端的な例はソースコードである。ソースコードは通常そのままでは実行できないが，コンパイラーを用いれば，実行可能なオブジェクトコードに容易に変換できる。

本定義を，前節で示した各種の定義と照らし合わせてみる。

まず，順序は前後するが，「実行によって役務を提供するような情報」という後半の定義について見ていく。この点については，前節に挙げた定義ではどれも，「一の結果を得る」「一定の機能を果たす」「何らかの結果をもたらす」など，表現は違えどほぼ同等な意味を表すものとなっている。

次に「その時点で利用可能なコンピューティング環境で実行可能，あるいは事実上実行可能」という点について見ていく。前節に示した定義ではどれも，多少の表現の相違はあれ，「計算機で実行できる」ことをプログラム(本稿でいうソフトウェア)の要件としており，これについては本定義にも違いはない。

他の定義と異なる本定義の特徴の一つは，「その時点で利用可能な」という文言である。これは，技術開発により，従来ソフトウェアではなかったものをソフトウェアとなることを念頭においている。

例えば，ソフトウェアの分野では，動作の説明に擬似コード(pseudo-code)を利用することがよくある。擬似コードは，実在しないが，(多くの場合)実在する言語に類似したプログラミング言語によって書かれたコードである。それらは第一義的には，ソフトウェアの動作に関する理解を助けるためのものであり，通常は実行を目的としない。従って，例えばウィルスの書き方を指南する文書にこの擬似コードが書かれていても，それはソフトウェアではないため，作者にソフトウェアに関する責任はない。

しかし，そうして擬似コードが提供された後で，その擬似コードを実行できる環境を開発することは技術的に可能である。本定義では，そうした状況も考慮し，作成後に実行可能になった場合もソフトウェアとみなすことで，ソフトウェア製作者等の責任を問えるようにしている。

この定義では，いつか自然言語が実行可能なコンピューティング環境が開発された場合，全ての記述がソフト

ウェアとみなされ，責任を問われるようになってしまうのではないかという懸念があるかもしれない。たしかにその場合定義上は全ての記述がソフトウェアとなるが，筆者のフレイムワークでは「ソース公開」による免責が認められている。ここでいうソースコードは，人間可読性，正確には人間可理解性を提供するものを想定している。従って，自然言語で記述されたものは，通常自動的に免責されると考えてよい。

本定義のもう一つの特徴は，「コンピューティング環境」という語である。これは，現代のソフトウェアが，一台の計算機上だけでなく，複数の計算機がネットワーク上で連携する環境で動作することもあることを考慮したものである。

本定義の特徴の最後は，前の方でも述べた「事実上実行可能」という表現である。これは，プログラムのみならずプログラムリスト的なものもソフトウェアに含むというスタンスを表しているが，こうした定義を採用した理由は，プログラムリストのみを提供する形態をとることで責任を免れようとする悪質ソフトウェア製作者等の存在を考慮したことによる。

最後に，ここで示した定義を用いて「レシピとソフトウェアの違い」を考えてみる。レシピは，それが紙に書かれていようとWebページに記載されていようと，コンピューターデータとして読みこむことが事実上可能である。しかし，通常それは単なるデータであり，コンピューターを用いて実行する＝役務を提供させることはできない。ただし，当該レシピが，例えば何らかの調理用コンピューターで実行可能な言語で記述されているのであれば，それはソフトウェアである。例えば，もしそのレシピを(調理用コンピューター上で)実行して作った料理について，レシピの欠陥がもとで例えば食中毒が起き，これによりさまざまな損害が生じたとする。この場合，当該レシピの製作者等は，筆者のフレイムワークの下では，製造物責任を問われることになる。

5 まとめ

本稿では，安全・安心なネットワーク社会の基盤となるべきソフトウェアについて，その脆弱性低減のインセンティヴを強化する手段の一つとして，ソフトウェアの特性を考慮したうえで，1)ソフトウェアに製造物責任を課す一方で，2)ソースコードが公開された場合は製造物責任を免責可能とする，法的フレイムワークについて述べた上で，当該フレイムワークの対象となるソフトウェアの定義の問題に対して，「ソフトウェアとは，その時点で利用可能なコンピューティング環境で実行可能，あるいは事実上実行可能であり，実行によって役務を提供するような情報」との定義を提案した。

今後の研究方針としては，まず本稿では触れられなかった既存法におけるソフトウェアの定義，例えば不正競争防止法やEUの特許指令などについて，今後調査を進めていく必要があると考えている。

これらを踏まえた上で，ソフトウェア製造物責任の法的フレイムワークをさらに精緻化し，より妥当なソフトウェア製造物責任を考えて行くため，さまざまな立場の方々と議論を深めていきたいと考えている。

参考文献

[1] 中山信弘.『ソフトウェアの法的保護に伴う国際問題』NIRA経済政策研究シリーズ13 企業の多国籍化に伴う法的諸問題-4. 総合研究開発機構. 1985.

[2] 中山信弘.『ソフトウェアの法的保護(新版)』. 有斐閣. 1988.

[3] 経済産業省 産業構造審議会 知的財産政策部会 第3回不正競争防止委員会 配布資料. 2002. http://www.meti.go.jp/report/downloadfiles/g20904bj.pdf

[4] 経済産業省 セキュリティホールに関する法令等の国内外調査委員会:『セキュリティホールに関する法律の諸外国調査』報告書 付録A 各国法令等整備状況一覧表. 2003. http://www.meti.go.jp/policy/netsecurity/Foreign_Law_AppendixA.pdf

[5] 谷口展郎. ソフトウェアの製造物責任とオープンソースソフトウェアに関する一考察. 情処研報 EIP-28-4. 2005.

[6] 財務省: 平成18年度税制改正 情報基盤強化税制. 2006. http://www.mof.go.jp/jouhou/syuzei/zeisei06/html/contents/02/index.html#02b

[7] 独立行政法人 情報処理推進機構(IPA): ISO/IEC 15408 ITセキュリティ評価及び認証制度. 2006. http://www.ipa.go.jp/security/jisec/documents/pamph_0605.pdf

[8] 特許庁: 特許法改正(平成14年4月)に伴う審査基準「コンピュータ・ソフトウエア関連発明」の適用について. 2004. http://www.jpo.go.jp/shiryou/kijun/kijun2/pu-kijun_cs-tekiyou.htm

[9] 特許庁: 特許・実用新案審査基準. 第VII部 第1章 コンピュータ・ソフトウェア関連発明. 2006. http://www.jpo.go.jp/shiryou/kijun/kijun2/pdf/tjkijun_vii-1.pdf

[10] 鶴田祐紀, 薬 剛. オープンソースによるソフトウェア利用者保護の枠組み構築に関する考察. SITE2004-19 信学技報 Vol.104 No.392. pp:37-42. 2004.

[11] マーク・ミナシ 著. 植木不等式 監訳.『いつまでバグを買わされるのか』. ダイヤモンド社. 2000.

[12] National Conference of Commissioners on Uniform State Laws: UCITA STANDBY COMMITTEE IS DISCHARGED. http://www.nccusl.org/nccusl/DesktopModules/NewsDisplay.aspx?itemID=56

[13] 曽野裕夫. UCC第2B編(コンピュータ情報取引)起草作業のその後. 2000. http://www.law.kyushu-u.ac.jp/~sono/softic99.htm