DOI:10.15939/j.jujsse.2013.01.017

□民商事法律研究

软件接口代码可著作权性研究
——兼评《著作权法》第三次修改草案“反向工程条款”

张吉豫

[摘 要] 软件接口是计算机程序中用于实现软、硬件各成分之间互联和交互的部分。接口相关代码本身构成表达，属于计算机程序的一部分，但绝大部分接口相关代码是由接口设计、编程规范和习惯等因素决定的有限的表达或常规套路的表达，因此不能受到著作权法保护。软件开发者为获取接口信息，往往要对整个程序实施反向工程。我国《著作权法》第三次修改草案（征求意见稿）允许在一定条件下实施软件反向工程，这对于防止著作权人垄断代码思想、促进软件间的互操作性具有重要意义，符合软件产业发展和社会公共利益的需要，但其中部分表述可能对软件反向工程的合法范围带来不必要的限制。

[关键词] 软件接口；软件著作权；反向工程
[收稿日期] 2011－07－09

[作者简介] 张吉豫，中国人民大学博士后流动站研究人员，理学博士。（北京 100872）

计算机程序中实现软、硬件成分之间互联和交互的部分通常被称为“接口”。系统间互联和交互的能力对信息产业非常重要，在诸多企业从传统的“封闭式创新”走向“开放式创新”尤为如此。知识产权法对接口相关权利的分配及限制，对信息产业发展有着重要影响。目前，人们对接口相关代码是否受著作权法保护、是否可为获取接口信息而进行反向工程等方面还存在一些误解和分歧。2012年3月，由国家版权局公布的《著作权法》第三次修改草案（征求意见稿）（以下简称《著作权法》修改草案）中第43条即针对为获取兼容性信息的目的，在结果管制等条件下，允许对必要的软件部分实施反向工程。本文将在厘清软件接口相关代码属性的基础上，分析软件接口相关代码是否可受著作权法保护，并对该条款进行评论。

一、接口相关代码性质分析

计算机系统各成分间具有多种通过接口进行交互的形式，如函数调用、约定结构的数据传输、约定的文件格式等。与接口相关的代码可以粗略分为两类：一是单纯用来描述存在某种形式接口的语句。以C语言为例，如果程序文件A中希望使用文件B中实现的某一功能①，则可在文件A内声明存在一个外部函数，并且该函数的名称、返回值类型、参数列表中各参数的类型、

① 在常见计算机高级编程语言中，程序具体执行和操作的功能均实现在一个或若干个函数（function）内部。

顺序等均需要与实现该函数的文件 B 中的写法完全一致（此即外部函数声明语句）。如果要向其他文件中的函数传递该函数要求的具有特定结构的数据，则通常会类似地在本文件中声明或定义实质相同的数据结构。二是涉及具体程序执行的语句，包括按照接口规范准备参数及调用函数、传递数据、在文件等存储空间的固定位置按约定的格式或数值写入数据等。为表述方便，本文将上述用来描述接口、不涉及程序执行的代码，简称为接口描述代码；将调用接口函数的语句以及实现接口所必要的代码等，简称为接口实现代码。二者统称为接口相关代码。

学界对于接口实现代码属于计算机程序的一部分通常并无异议，而对接口描述代码是否属于著作权法意义下的计算机程序则有不同意见。有些观点认为，函数声明、数据结构定义等不是函数思想观念的表达，仅仅是思想的一部分要素，因而既不是表达也不是思想；程序的逻辑实现代码才是函数的表达。例如在英特尔诉东进公司一案①引发的“头文件、软件兼容性与版权保护”研讨会中，有专家认为“由于头文件的内容通常是程序函数的声明或常量的定义，没有可执行的代码，因而头文件不是著作权保护意义上的软件本身”[1]。笔者认为这种说法并不恰当。一方面，函数声明、数据结构定义等语句有具体的符号载体，已然构成表达，没有理由因其不是描述函数具体功能的语句，就认为其不构成表达。另一方面，我国《计算机软件保护条例》第 3 条中规定“计算机程序是指为了得到某种结果而可以由计算机等具有信息处理能力的装置执行的代码化指令序列，或者可以被自动转换成代码化指令序列的符号化指令序列或者符号化语句序列”。尽管函数声明、数据结构定义等描述本身并不是代码化指令序列或符号化指令序列，但这些语句属于“符号化语句序列”的一部分，也是将其他相关符号化语句序列正确转换成代码化指令序列所不可缺少的，应属于软件保护条例中规定的“计算机程序”的一部分。文件格式一般不属于计算机程序的一部分。但在计算机程序中可能存在对文件读写的语句和为便利读写而定义相应的数据结构的语句。同上可证，这两类语句也属于表达，是程序的一部分。

二、接口相关代码可著作权性分析

程序中函数声明、数据结构定义、文件读写等语句虽然属于表达，亦属于计算机程序的一部分，但接口相关的这些代码是否受到著作权法保护仍需根据著作权法的基本原则来具体分析。我国《著作权法》第 1 条明确了“保护著作权”、“鼓励作品创作和传播”、“促进文化和科学事业的发展”三项目的。笔者赞同当前的主流观点，即著作权法之目的是在激励作者创新和保证公众获得知识及进行再创造之间寻求一种有效的、最具有生产力的平衡，以促进知识、文化发展。在这一目标驱动下，现代著作权法普遍规定，著作权法保护的是有独创性的表达，不延及表达所蕴含的思想。“独创性”是著作权法下作品的一项实质要件；“思想/表达”二分法是著作权法的重要原则，即著作权法保护的是对思想的有独创性的表达，而非表达中蕴含的思想。以这两项基本原则为基础，在著作权法发展过程中逐渐总结出一些原则和分析方法，如思想表达的合并原则、场景原则②（Scènes à faire）等。根据著作权法之立法目的而推演、设计出的这些规则与知识产权体系中不同权利分工相配合，共同勾勒出著作权法保护对象的边界。

对立法目的之追求以及对知识产权保护体系内权利分工的维系，是将著作权法基本概念及原

① 《英特尔公司诉深圳市东进通讯技术股份有限公司计算机软件著作权侵权纠纷案》，广东省深圳市中级人民法院，（2005）深中法民三初字第 328 - 1 号。该案于 2007 年和解。

② 场景原则也称“常规表达套路原则”，最初指对于为了描述特定历史时期所必要的一些标准场景元素，即使这些场景已由在先作品描述，他人以自己的表达描写相同场景不构成侵权。后来被应用于其他方面，如判断游戏规则说明是否侵犯著作权等。

则应用于软件领域时对模糊之处进行解释的基础。我国软件保护条例目的条款指出，软件著作权的设立旨在鼓励计算机软件的开发与应用，促进软件产业和国民经济信息化的发展。接口是软件间交互或兼容的渠道。接口相关代码虽构成表达，但通常在整个程序中只占很小的比例，更重要的是，实现同一接口的代码在大多数情况下仅有非常有限的表达形式。如果这种表达被著作权法赋予至少 50 年的垄断权，将使软件开发者可以在事实上垄断本应通过专利法寻求保护的接口设计思想，将极大限制程序间的兼容和交互，增加重复开发，加重用户的软件更换代价，并可能形成相关技术的后续开发的障碍。[2] 1948－1960 人们运用著作权法的既有原则解释了软件接口代码不构成著作权法保护对象之原因，主要有如下几种结论相近但理由不同的观点：1）操作方法说。此观点认为软件接口属于操作方法，因此不受著作权法保护。例如在 Lotus v. Borland 一案①终审判决中指出，菜单命令层次结构属于操作方法，因此不受著作权法保护。在 2012 年 Oracle v. Google 一案中，Alsup 法官认为 Java 类库和方法的命名以及组织构成了“一个系统的命令结构或者其应用程序编程接口的操作方法”，因此不受著作权法保护 。2）不具独创性说。此观点认为许多接口达不到独创性的要求，故不受著作权法保护。3）思想表达合并说。此观点认为，如果想与某一程序兼容或进行互操作，则必须遵循相同的接口标准，在这一限制下代码编写方式极为有限，思想和表达难于分割，故为了不致使思想被垄断，著作权法对该表达不予保护。②

笔者认为，操作方法说仅仅说明接口属于操作方法，并不能很好地解释表达这一操作方法的代码不受著作权法保护的理由，正如我们不能因为一个作品蕴含了思想、操作方法或处理过程就否定对其表达的保护一样。更何况对于计算机程序是为达到某个结果而采取的运算步骤这一层面而言，每个计算机程序都可被看作“处理过程”或对计算机的“操作方法”。[3] 3302 但若从这一角度来解释则几乎无法为软件提供著作权保护。不具独创性说首先依赖对独创性的定义。传统上在版权体系和作者权体系之间，独创性具有不同的含义和标准。欧盟软件指令希望能够统一欧盟国家对计算机程序独创性的定义，明确规定程序具有独创性的含义为“是作者自己的智力创造”，但目前欧盟各国在司法过程中适用的独创性标准仍有差异。[4] 891－902 不管怎样，不具独创性说确实可以说明一部分软件接口不受著作权法保护的原因。例如大多数编程人员都会认同在代码中使用“StudentID”来表示“学号”并不具有独创性。但许多软件系统接口是庞大而复杂的组合。例如在 Oracle 诉 Google 一案中，涉及 6 000 多个方法（即函数），这些方法被划分为若干类，并被进一步划分为 37 个类库。在大部分国家对独创性的解释下，这样庞大数目的方法名称和组织方式的选择是具有独创性的。思想表达合并说可以较清晰地解释大部分接口代码不能受到著作权法保护的原因。著作权法不禁止他人独立开发可以与已有计算机软件进行交互的软件或具有相同功能的软件。然而他人在编写接口相关代码时，往往会根据接口的客观要求编写出与已有软件中接口相关部分非常相似的代码。笔者认为，可进一步细分为表达由接口设计直接决定和间接决定两种情况进行讨论。表达由接口设计直接决定的情形包括由接口函数名称决定等。例如，调用接口函数时必须使用相同的函数名和类型、顺序一致的参数列表。此时软件接口的实现方式极为有限，甚至很多时候是唯一的。根据思想—表达的合并原则，这类接口代码不受著作权法保护。最高人民法院在一份批复函中认定，程序运行参数属于软件编制过程中的构思③。我们可以认为，其实

① Lotus Development Corp. v. Borland International，516 U. S. 233（1996）.

② Oracle America，Inc. v. Google Inc.，2012 U. S. Dist. LEXIS 75896. Alsup 法官似乎担心人们对这一结论质疑。他在论述了属于操作方法因而不受保护之后，还补充说明从兼容性的要求来看，这些代码不受著作权法保护就更加清晰了。

③ 《最高人民法院关于深圳市帝慧科技实业有限公司与连樟文等计算机软件著作权侵权纠纷案的函》，（〔1999〕知监字第 18 号函）。

际所指的是这种构思和表达合并了。这里所说的程序运行参数即属于软件对外接口。表达由接口设计间接决定的情况主要是由于程序编写的需要和存在业界广泛采用的编程规范。例如，虽然文件格式不属于软件，但在程序中可能存在对遵循固定格式的文件进行读写的语句，该部分代码可能采用一般程序员常用的读写语句，或依据格式规范而编写的数据结构。例如，如果接口约定在文件起始第三个字节处写入数值 “1”，基于功能需要、书写方便和程序运行效率等考虑，加上在工程培训中养成的关于文件读写操作的常规写法的习惯，大部分程序员都会独立地采用相似的表达。这种构成可以应用合并原则和场景原则的一种实例。并且我们也可以论证这种常规表达套路不具有独创性，不能受到著作权法保护。

综上所述，绝大部分接口相关代码是由接口设计、编程规范和习惯等因素决定的，不能受到著作权法保护。需要说明的是，如果存在某种特殊的接口相关代码具有独创性且不会导致思想表达的合并，其仍可能受到著作权法的保护；并且在任何大型程序中均存在大量函数声明、数据结构定义等语句，大部分是为实现程序内部功能、为程序内部函数之间、模块之间相互调用、通讯所需要的，只有很少部分构成对外的接口。我们不应简单认定所有声明和定义语句均不是著作权法保护对象，否则会使软件著作权人的合法权利得不到应有的保护。

三、接口信息获取与软件著作权限制

接口信息及大部分接口相关代码不受著作权法保护，但能否为了获得接口信息而对软件进行反向工程，是著作权法中与软件接口密切相关的一个问题。计算机软件往往以二进制目标代码的形式向公众发布。人们可以通过观察来了解程序界面，但对于函数接口规范、数据传输格式等不受著作权法保护的思想，如果软件开发者不将其公开，则往往只能通过反向工程来获得。广义上，反向工程指从人造物中分析并获取技艺和知识的过程。[5] 1577 软件反向工程指通过对目标软件系统进行分析，从而识别该系统的各组成成分及其相互关系以及创造系统的另一种形式的表达或更高抽象层次表达的过程。[6] 15 在知识产权领域讨论较多的软件反向工程技术，主要是通过反汇编、反编译①等方式对软件的二进制目标代码（即由机器指令构成的程序）进行解析，还原软件的汇编代码或源程序，进而得出该软件的接口信息、组织结构、算法流程等设计要素。软件反向工程是理解计算机程序的一项重要途径，并不包含开发新程序或相同、相似程序的过程。

在软件反向工程过程中，通常不可能在开始前就明确软件目标代码中具体哪些段与接口相关，因此不可避免地涉及将软件目标代码的全部或大部分转换为适于人类理解的描述，并在此基础上分析接口信息。不管该工作是自动还是手动进行，这种转换过程均涉及复制。如果不对软件著作权进行相关限制，则势必导致软件著作权人可以对软件接口信息获得事实上的垄断，影响软件产业的发展和社会利益。欧盟及美国等地区和国家在立法和司法实践中纷纷允许了一定条件下的反向工程。欧盟在 1991 年《欧盟计算机软件保护指令》中专门设立了“反编译”条款，允许为实现软件兼容的目的，在限定条件下进行反向工程。美国著名的 Sega v. Accolade 一案②中，Accolade 公司为了获知 Sega 公司的游戏终端接口，对 Sega 公司的软件进行了反汇编和反编译。法院认为，Accolade 的复制只是为了获得不受版权法保护的兼容性信息，并且这是 Accolade 公司获得兼容性信息的唯一途径，依法构成合理使用。1998 年美国颁布的《数字千年版权法》明确

① 反汇编指将软件的二进制目标代码转换为人可以理解的汇编代码的过程；反编译指将二进制目标代码或汇编代码转换为高级语言源程序的过程。

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

规定了禁制规避技术措施的例外包括反向工程：合法获得计算机程序使用权的人，为识别和分析与其他程序进行互操作所必需的程序要素，可在《版权法》允许的范围内，规避及开发技术手段来规避技术设施。关于是否允许反向工程、应具备怎样的约束条件等尚需进行深入而系统的研究。本文仅针对为获取兼容性信息而进行反向工程这一问题，就我国《著作权法》第三次修改草案的相关规定进行反思，并为著作权法的完善提出建议。

在2012年3月公布的《著作权法》修改草案征求意见稿中，增加了一条针对反向工程的条款，即第43条“计算机程序的合法授权使用者在通过正常途径无法获取必要的兼容性信息时，可以不经该程序著作权人许可，复制和翻译该程序中与兼容性信息有关的部分内容”。“适用前款规定获取的信息，不得超出计算机程序兼容的目的使用，不得提供给他人，不得用于开发、生产或销售实质性相似的计算机程序，不得用于任何侵犯著作权的行为。”该条款对于防止著作权人垄断代码思想、促进软件间的互操作性具有重要意义，但具体表述仍存在值得商榷之处。

首先，该条款允许在一定约束条件下，“复制和翻译该程序中与兼容性信息有关的部分内容”。一方面，著作权中包含“翻译权”，即“将作品从一种语言文字转换成另一种语言文字的权利”。笔者认为第43条中的“翻译”实际指类似于从二进制代码到汇编代码或高级语言的转换，而非从中文到英文的语言翻译。虽然这种转换也可以称为一种翻译，但为避免在法律条文中一词多义，建议采用“形式转换”的表述。另一方面，条款中的“兼容性信息有关的部分”范围并不明晰。如果理解为仅指软件接口相关代码，根据本文在第二部分中的分析，通常那些代码并不能成为著作权法保护的对象，对其进行复制本就应该是自由的。既然本无权利，就更无从提起权利的限制。而软件反向工程中问题在于，人们无法在一开始就从目标文件（二进制代码）中区分哪些部分与兼容性信息有关，因而往往对与兼容性信息无关的、受著作权法保护的部分也进行复制和转换，之后进一步分析出与兼容性信息直接有关的部分。因此笔者认为，修改稿中“复制和翻译与兼容性信息有关的部分内容”应进行调整，例如改为“可以为了获得该程序的兼容性信息而在必要限度内对该程序实施反向工程”[7]531。在43条第2款中对反向工程获得的结果已经进行了严格规制，因此这样修改虽然授权程序的合法使用者为获取兼容性信息在必要时对程序进行整体复制和转换，但该复制仅在实施反向工程过程中被允许，并没有授权实施反向工程者在自己开发的程序中复制他人软件中受保护部分，对著作权人的合法权益不会造成侵害。

其次，该条款中规定“在通过正常途径无法获取必要的兼容性信息时”，可以进行复制和翻译。这种基于反向工程必要性的规则本身非常可能为后继软件开发者带来陷阱，并且条款中“正常途径”这一表述也具有含糊性，可能引起双方争议，增加诉讼成本或妨碍反向工程的进行。一是，这一规则可能使得在先软件开发商在提供兼容性信息时附加各种条件，从而在事实上可能使得后继软件开发者只能付出不希望的代价来获取兼容性信息，而不能进行反向工程。例如，如果在先软件开发者以高昂的费用提供必要的兼容性信息，是否就构成了一种“正常途径”？如果成立，则后继软件开发商要么花费大量财力，要么放弃获取信息，不能合法进行反向工程。再如，在Sega v. Accolade一案中，签订合同即可直接获取Sega的接口信息，但条件是Accolade必须停止为其他平台编写游戏。权衡之下Accolade选择了进行反向工程来获取接口信息。Sega的这种带有条件的许可又是否属于此处所指的“正常途径”？二是，关于哪些兼容性信息是“必要”的，双方可能存在争议。原软件著作权人可能公开或许可一部分兼容性信息，但后继软件开发者可能认为并不充分，仍有一些必要信息需要通过反向工程来获得。这可能只能通过昂贵的诉讼来进行裁定。有学者曾从效益角度论证过欧盟软件指令中“不能轻易获取”兼容性信息这一限制的合理性。[8]121的确，法律引入这一对反向工程的限制规则应有其明确目的和理

由。笔者认为，此处限制的初衷在于，一方面进一步明确法律允许实施的反向工程的目的在于获得兼容性信息，另一方面希望激励软件开发商主动公开或以低于反向工程成本的合理价格出售接口信息，便利计算机软件之间的兼容和互操作性的实现，促进产业发展，并且可以减少不必要的反向工程，从而减少社会财富和社会劳动力的浪费。如果确实可以在合理的代价下获得兼容性信息，除非有除获得信息之外的目的，否则后继软件开发者也无理由选择进行昂贵且费时的软件工程，因此不需要此条款来辅助进行对反向工程目的的约束和判断。笔者建议，将此限制去掉或修改为“在此前并未获得兼容性信息时”，以减少对软件反向工程带来不合理的限制。这样修改后，如果在先软件开发者确实以比采用反向工程更加合理的价格或条件提供兼容性信息，并且为后继软件开发者所接受，则后继软件开发者依然不能再进行反向工程，否则其目的显然不是为获取兼容性信息，有违目的约束。

最后，修改草案仅在第 67 条中规定了四种可规避技术保护措施的例外，缺少更细致或更灵活的规定。如前所述，美国《版权法》规定了为获取互操作信息而进行的反向工程，可以规避技术措施及开发所需技术手段。如果没有这一例外，对于采取了技术保护措施的软件，仍然不能通过反向工程来获得接口信息。这将在许多情况下架空 43 条。修改草案中应对此进行完善。

软件的技术性和功能性与著作权法保护的其他对象有显著区别，并且软件的“复制”具有更多的目的和形式，这引起了关于软件著作权保护范围和权利限制进行特别关注的必要。本文聚焦于接口相关代码，从其可著作权性及相关权利限制角度进行了研究。主要结论如下：1）接口相关代码构成表达，属于计算机程序一部分。2）绝大部分接口相关代码是由接口设计、编程规范和习惯等因素决定的有限的表达或常规套路的表达，根据著作权法的基本原则，不能受到著作权法保护。3）尽管软件对外接口不应受著作权法保护，但为获取接口信息，往往要对整个目标文件实施软件反向工程，并且这一过程中涉及复制。为避免使本来不受著作权法保护的接口信息在事实上被软件著作权人垄断，有必要在著作权法中设置相关限制。我国《著作权法》第三次修改草案（征求意见稿）中，增加了在一定条件下允许软件反向工程的条款，同时明确管制对反向工程结果的不正当使用，符合软件产业发展需要，但其中部分表述值得商榷，并且缺少为获取兼容性信息进行反向工程可以避开技术保护措施的规定，建议予以完善。

参考文献

[1] 朱泉峰 《该不该保护“头文件”》，《计算机世界》，2005 年 6 月 27 日，第 A14 版。

[2] Samuelson P. Are patents on interfaces impeding interoperability? *Minnesota Law Review*，2009，93（6）：1943 – 2019.

[3] Mahajan A. Intellectual property，contracts，and reverse engineering after ProCD：A proposed compromise for computer software. *Fordham Law Review*，1999，67（6）：3297 – 3335.

[4] Mylly U. Harmonizing copyright rules for computer program interface protection. *University of Louisville Law Review*，2010，48（4）：877 – 909.

[5] Samuelson P，Scotchmer S. The law and economics of reverse engineering. *Yale Law Journal*，2002，111（7）：1575 – 1663.

[6] Chikofsky E，Cross J. Reverse engineering and design recovery：A taxonomy. *IEEE Software*，1990，7（1）：13 – 17.

[7] 《十二国著作权法》翻译组 《十二国著作权法》，北京：清华大学出版社，2011 年。

[8] 曹伟 《计算机软件版权保护的反思与超越》，北京：法律出版社，2010 年。

责任编辑：高 玥 李佳欣