

软件著作权侵权认定的司法实践与典型案例分析

阎涵

引言

随着信息技术的发展，软件著作权保护日益成为企业创新与知识产权保护的重要环节。在实际司法实践中，认定软件侵权依托“接触+实质性相似”原则，通过多维度证据评估被诉软件是否侵害了权利软件。通过典型案例分析，本文归纳软件著作权侵权认定的核心规则、举证责任分配及特殊情形下的判例经验，旨在对相关司法实践进行系统研究与理论探讨。

1. 软件著作权侵权认定的通用规则：接触+实质性相似原则

在我国司法实践中，认定侵害著作权的基本原则通常被概括为“接触+实质性相似”。就计算机软件而言，该原则要求，被告在创作或开发被诉侵权软件时，应存在接触原告权利软件的可能性或实际接触；同时，被诉软件与权利软件在核心内容上存在相同或实质性相似；若被告无法提供合理解释或证据以证明独立创作，则可认定其侵害了原告的计算机软件著作权。

这一原则的核心在于对两个层面的判断，一是接触可能性，即被告是否有机会获得权利软件；二是实质性相似，即双方软件在受保护的表达部分是否高度相似。这两者缺一不可，是司法实践中判断侵权的基本逻辑。

1.1 判断“接触”可能性

“接触”在司法实务中并非要求证明被告一定直接复制或看到过原告软件，而是确认一种合理可能性即可。实践中，法院通常将“接触可能性”分为以下三类：

直接可能性：直接接触权利软件的可能性，如参与开发、测试或销售过程获得源代码或目标程序。例如，（2018）沪73民初32号案中，被告系原告前核心员工，参与软件开发及销售，法院认定其有直接接触可能性。

间接可能性：通过公开渠道或第三方间接接触软件的可能性。例如，（2021）粤 73 知民初 105 号案中，原告的软件先行在官网公开，被告后续使用的涉案软件中出现与原告文件及标识一致内容，法院认定被告存在通过官网下载接触权利软件的合理可能性。

推定可能性：当被诉软件与权利软件出现高度不合常理的相似性，而被告无法提供合理解释时，可从实质性相似反推接触。例如，（2021）最高法知民终 354 号案中，被诉软件与权利软件在文件内容、文件大小及后缀名上存在高度一致性，法院在被告未作出合理解释情况下，直接推定接触可能性。

在司法实务中，“接触”要件的认定门槛相对较低，重在确认被告可能获得权利软件，为实质性相似判断提供前置条件。

1.2 判断“实质性相似”

在计算机软件著作权侵权案件中，“实质性相似”的判断是区分合法借鉴与侵权复制的核心标准。国际司法实践中主要形成了两种具有代表性的分析框架：一是“结构、顺序与组织标准”（Structure - Sequence - Organization，简称 SS0 标准），二是“抽象—过滤—比较”三步测试法（Abstraction - Filtration - Comparison，简称 AFC 测试法）。虽然我国现行法律法规并未直接采用上述术语，但其核心理念已通过司法解释与判例实践被吸收并本土化应用，成为人民法院在认定软件作品相似性时的重要参照。

1.2.1 SS0 标准规则概述

SS0 标准最早由美国宾夕法尼亚东区法院在 Whelan v. Jaslow (1985) 案中确立，其核心判断逻辑是：若两款软件在整体体系结构、功能模块顺序以及系统组织方式上形成高度一致，即便具体代码层面存在差异，亦可能构成著作权法意义上的侵权。该标准实质上将软件的“结构、顺序与组织”视作受保护的表达形式，而非仅限于源代码的逐字相似。

在我国，《计算机软件保护条例》第六条明确规定，软件著作权保护不及于“开发软件所用的思想、处理过程、操作方法或者数学概念等”。该条体现的原

则与 SSO 标准的核心逻辑高度一致，均旨在通过区分思想与表达，防止知识产权保护过度延伸至技术思想层面。

1.2.2 AFC 三步测试法概述

AFC 测试法由美国第二巡回上诉法院在 *Computer Associates v. Altai* (1992) 案中提出，其核心思想在于通过多层过滤以剔除不应受保护的元素，从而更精确地界定软件著作权的客体。该方法包括三个阶段。首先是抽象 (Abstraction)：将软件从具体代码到功能目标分解为不同层级，以区分“思想”与“表达”；其次是过滤 (Filtration)：剔除属于公有领域、功能性必然要求（合并原则）或技术标准约束的表达；最后是比较 (Comparison)：在前述过滤后的剩余部分中，对独创性表达进行逐层相似性比对。

AFC 方法在逻辑上比 SSO 标准更精细，其“抽象—过滤—比较”的层层递进结构与我国著作权保护“仅限于独创性表达”的原则高度契合。尽管在我国法律条文中没有直接体现，且司法判决中鲜有直接引用该术语，但法院在实务操作中已普遍采纳类似的分析思路。

1.2.3 我国司法实践中的综合比对——SSO 标准、AFC 测试法的本土化应用

在确认“接触可能性”后，人民法院对“实质性相似”的审查往往并非拘泥于单一层面的比对，而是形成了兼顾整体结构与表达内容的综合分析模式。这一模式在逻辑上融合了 SSO 标准中对“整体结构、顺序与组织”的宏观比对思路，以及 AFC 测试法对“抽象、过滤、比较”的分层分析逻辑。

具体而言，司法实务中常采用以下几种比对路径：

运行界面与功能流程比对——SSO 的标准规则的实务体现。在终端用户软件（如企业管理系统、办公软件）侵权案件中，法院通常从软件的整体架构、界面布局、功能模块划分及调用顺序等方面入手，对比双方在系统组织与功能逻辑上的相似程度。这种方法实质上对应于 SSO 标准的“结构—顺序—组织”分析思路，通过对整体功能设计与逻辑框架的比对判断两软件是否存在结构性重合。各地法院在多起企业 ERP 系统案件中认定，两软件在功能模块划分及流程调用上高度一致，构成对原告软件结构性表达的实质性复制。

源代码直接比对——AFC 测试法的实务体现。当源代码可被直接获取时，法院通常委托技术鉴定机构对关键代码段进行逐行或功能块对比。

鉴定人员先根据软件功能、模块结构，将源代码划分为不同的功能块或层级，例如界面层、业务逻辑层、算法层、数据库交互层等，对应 AFC 的“抽象”阶段。而后剔除以下不受保护的部分：技术实现中不可避免的表达（例如特定 API 调用、算法流程的固定语句）；通用逻辑代码、开源库代码、第三方组件；行业标准或功能性必然要求的部分（合并原则），对应 AFC 的“过滤”阶段。最后，对过滤后保留的独创性表达进行比对，重点审查是否存在连续、系统性的相似，例如：相同的结构层次或调用顺序；大量功能模块在逻辑上呈现高度一致；变量命名、注释风格、算法组织等非功能性元素的相似，对应 AFC 的“比较”阶段。

在过滤掉非独创性代码、通用算法及标准接口之后，对剩余具有独创性的表达进行实质性相似判断，此类比对具有最高的证明力，是判断软件侵权最具确定性的证据形式。

源代码与目标程序比对——SSO 与 AFC 的交叉应用。当被告拒绝提交源代码时，法院可通过反编译目标程序，并结合运行结果和算法输出特征进行分析。该方法一方面遵循 AFC 逻辑，对反编译代码中非独创性或功能性必然部分进行过滤；另一方面结合 SSO 视角，从程序宏观结构、模块划分及调用顺序上审查两者是否一致。例如，北京一中院审理的（2010）一中民终字第 14256 号案中，法院通过目标程序比对确认双方软件在模块调用和逻辑结构上高度一致，从而形成了推定源代码存在复制关系的高度合理依据。

综上所述，SSO 标准与 AFC 测试法在软件著作权侵权认定中各有侧重，但在实践中具有互补性。SSO 标准侧重于软件的整体结构、模块顺序及系统组织，通过宏观层面的比较识别可能的表达重合，解决“软件整体是否相似”的问题；而 AFC 方法通过抽象、过滤和逐层比较，更精细地剔除非保护性元素，集中于独创性表达的微观分析，解决“哪些部分应当受到保护”的问题。二者并非相互排斥，而是分析层次与适用范围不同、可交叉运用的关系。

2. 软件源代码侵权认定的典型案例分析

2.1 全局相似度低，同样可能构成侵权-（2022）最高法知民终 1605 号

2.1.1 案件事实

原告深圳某科技公司系“商品运营管理系统”软件的著作权人，被告黄某、陈某、叶某系原告前核心技术人员，离职后共同设立杭州某科技公司，并在短期内推出母婴出口电商平台。原告认为，被告在短时间内完成的软件功能架构与其高度一致，存在复制原告权利软件的行为。

本案的争议焦点在于：当被控软件整体结构及大部分代码与权利软件差异显著，仅部分文件存在相同或高度相似的源代码时，是否仍可认定构成“部分复制”侵权。

2.1.2 法院裁判逻辑与核心观点

经一审法院委托鉴定，双方软件中存在 9 个源代码文件（合计约 63KB）内容相同或实质性相似，占权利软件总代码量约 1.085%。一审法院据此认为相似比例过低，且分布分散，不足以构成实质性相似，驳回原告诉讼请求。

二审中，最高人民法院撤销原判，明确指出：

“复制比例并非判断侵权性质的决定性因素，著作权法并未对‘部分复制’设置数量或比例门槛。判断关键在于被复制部分是否属于具有独创性的表达。”

若被复制的部分在功能上具有相对独立性、在表达上具备独创性，即可构成受保护的“表达单元”；数量虽少，但如属于实现核心功能的模块、算法或关键逻辑，即应认定为侵权。

法院认为，本案中涉案的 9 个文件分别对应独立功能模块，具有独创性特征。被告未能证明该部分源于开源代码或表达受限，亦未能合理解释其短期内形成同类代码结构的原因，结合被告核心成员与权利软件开发团队存在直接接触关系，法院据此认定其复制了原告软件的核心表达内容，构成侵权。最高法同时指出，复制比例虽不影响侵权性质认定，但影响责任范围的衡量。因此在确认侵权成立的基础上，法院酌情考虑复制比例较低、侵权获利有限等因素，判令被告赔偿经济损失及合理开支共计 25 万元。

2.1.3 案例启示

在软件著作权侵权诉讼中，“全局相似度低”不能成为有效的抗辩理由。司法实践的关注点已从简单的代码数量对比，转向对软件中核心功能模块源代码质量的审视。只要被复制的部分构成了软件中具有独创性的相对独立功能模块，即使数量很少，同样可能被认定为侵权。

2.2 权利人初步举证后，举证责任可转移至对方—（2020）最高法知民终 209 号

2.2.1 案件事实

原告君意东方公司主张其享有著作权的计算机软件被被告东方瑞利公司非法复制并使用。由于被告掌握涉案软件的运行环境与源程序，原告无法直接取得双方软件的源代码进行比对。原告遂提交了包括软件目标程序、运行界面、数据库结构、运行参数等在内的证据材料，以证明两款软件在功能与表现上存在高度一致性。

被告在庭审中拒绝提供其软件源程序供司法鉴定机构比对，亦未提出合理解释。争议的核心在于：当权利人已完成初步举证，而关键证据掌握于对方时，法院能否将举证责任转移并适用“举证妨害”规则？

2.2.2 法院裁判逻辑与核心观点

最高人民法院在终审判决中明确指出，在软件著作权侵权诉讼中，举证责任应当根据证据掌握的客观状况进行动态分配。

法院认为，虽然权利人原则上负有证明双方软件“构成实质性相似”的举证责任，但这一责任并不要求其必须提供源代码。若权利人能够通过目标程序表现、用户界面、运行逻辑、数据库结构等外部可观察特征初步证明被控软件与权利软件在功能、结构或操作流程上存在高度重合，即已达到“初步举证”的标准。在此基础上，法院进一步援引《最高人民法院关于民事诉讼证据的若干规定》第 111 条的精神，指出：

当关键证据掌握于对方控制之下，而该方拒不提供且无合理理由时，人民法院可以认定其构成“举证妨害”，并据此推定对方当事人主张的事实成立。

因此，本案中被告东方瑞利公司负有提供源程序以供比对的便利条件与法定义务，却经法院释明与保全措施后仍拒不提交。其行为已构成证明妨害。法院据此推定原告所主张的实质性相似成立，认定被诉软件构成对原告权利软件的复制侵权。

最高人民法院同时强调，举证责任的转移并非免除原告的举证义务，而是基于证据控制能力差异的平衡性调整机制。该机制旨在防止当事人利用证据优势逃避责任，从而保障著作权司法保护的实效性。

2.2.3 案例启示

本案确立了我国软件著作权侵权诉讼中“举证责任动态分配”与“举证妨害推定”的实务标准。权利人并非必须掌握源代码，只需提供足以使人合理相信双方软件高度相似的证据，即可视为完成初步举证，当被告拒绝提供其控制下的关键证据时，法院可依举证妨害规则推定权利人主张成立。

该案在制度意义上体现了我国法院在知识产权诉讼中对“证据掌控不对称”的平衡思维，推动了软件侵权案件中举证责任分配从静态向动态、从形式向实质的转型，为“接触+实质性相似”规则的适用提供了程序保障。

2.3 软件包含开源代码，核心源代码被复制仍可认定侵权-（2018）京民终471号

GNU 通用公共许可证（GNU General Public License，简称 GPL）是一种典型的“强开源”许可协议，其核心特征在于“传染性”（Copyleft）条款。该机制要求：任何在 GPL 许可下获得的软件或其衍生作品，只要在分发或公开时仍包含 GPL 许可代码，即必须以同样的 GPL 协议开放源代码。换言之，GPL 不仅保障原作者代码的自由使用和再传播权，也通过传递义务确保派生作品继续遵循相同的开放规则。

2.3.1 案件事实

原告数字天堂公司系“HBuilder”软件及其多个功能插件的著作权人，相关插件包括“代码输入法功能插件”“真机运行功能插件”“边改边看功能插件”，均已完成著作权登记。被告柚子科技公司及柚子移动公司（以下合称“柚子公司”）共同开发并对外提供“APICloud”软件平台。

数字天堂公司主张，APICloud 软件在核心功能插件中大面积复制了 HBuilder 的源代码。经司法鉴定，三组插件的源代码重合度极高：

代码输入法插件：30 个文件中 29 个具有同一性；

真机运行插件：23 个文件中 18 个具有同一性；

边改边看插件：56 个文件中 44 个具有同一性。

柚子公司辩称，涉案插件包含大量开源代码或第三方代码，相关相似内容属于 GPL 协议所许可的自由使用部分。案件焦点在于：在软件包含开源代码的情形下，是否仍可认定对核心源代码的复制构成侵权，以及 GPL 开源协议的适用

2.3.2 法院裁判逻辑与核心观点

本案的裁判逻辑集中于两点：一是 GPL 协议适用范围的限定，二是在开源混同条件下的实质性相似认定。

（1）关于 GPL 协议的“传染性”边界

法院经审查认为，原告主张权利的三个功能插件分别以独立文件夹形式存在，且在 HBuilder 软件及插件目录中均未附带 GPL 协议或引用 GPL 开源组件。换言之，涉案插件并非 GPL 协议下的衍生作品或修订版本。

法院明确指出，判断 GPL 协议是否适用，不能仅依据功能调用关系或软件整体逻辑关联，而应关注相关模块在物理存储位置、代码耦合程度、独立运行能力等方面的事实特征。该判定思路实质上确立了对开源协议“物理独立性”的审查标准，限制了 GPL “传染性”条款的泛化适用。

（2）关于核心源代码的独创性与相似性认定

柚子公司辩称涉案插件中包含大量开源或第三方代码，但第二次鉴定结果显示：被复制的主要文件并非来自开源代码库，而是数字天堂公司独立创作的原创部分，与第三方代码不存在对应关系或仅存在极少重合。法院据此认定，被告复制的对象为原告软件的核心功能模块，其独创性表达应受著作权法保护。

法院强调，即便软件整体建立在开源框架之上，只要权利人对该框架进行了独立设计、功能创新或结构优化，其新增部分仍属于受保护的“表达层”内容。换言之，开源基础并不排除独创性代码的存在，被告不得以开源为由规避对原创部分的复制责任。

2.3.3 案例启示

本案确立了我国在开源与著作权保护冲突问题上的重要司法标准。在涉及 GPL 协议的混合代码结构中，只要权利人能够证明其贡献部分具有独创性表达，即可受到著作权法保护。虽然 GPL 协议具有开放性，但这并不赋予使用者无限制复制权，所谓“传染性”应以代码在物理上是否独立及与其他功能模块的耦合程度为判断标准。

2.4 根据双方软件运行过程存在的共同缺陷及个性化特征，推定构成实质性相似—（2007）苏民三终字第 0018 号，2015 年最高法指导案例 49 号

2.4.1 案件事实

原告石鸿林开发了“S 型线切割机床单片机控制器系统软件 V1.0”，并完成著作权登记。被告华仁电子资讯有限公司生产、销售的线切割机床控制器使用了与原告软件功能、界面及运行特征高度相似的系统软件。

由于被告控制器采用加密芯片固化程序，源代码无法直接提取比对，导致传统的“源代码层比对”路径不可行。原告遂提交了双方软件在运行中出现相同异常反应、界面结构、功能逻辑及使用说明书的比对资料，主张被告在无法独立开发的情况下复制其软件核心结构。

2.4.2 法院裁判逻辑与核心观点

江苏省高级人民法院在二审中明确指出，在无法进行源代码比对的情况下，软件运行特征与行为表现的同一性可以作为判断实质性相似的重要依据。法院采用了“从外部表现反推内部结构”的思路，通过比对软件运行过程中的异常现象与功能响应，认定两者具有一致的内部逻辑设计。

鉴定意见表明，原、被告软件在运行过程中均出现完全相同的设计性缺陷——当加工零件的连续程序段超过 2048 条时，系统均发生异常，且异常时蜂鸣器发出的声响相同。这种缺陷反映出底层程序在算法或数据处理逻辑上的一致性，非独立开发中可能偶然出现的情形。

同时，法院还比对了双方软件在操作界面、参数设置、菜单结构等方面的整体布局，发现二者在若干功能模块的排列顺序、命令名称和显示方式上均高度一致。这些细节体现了开发者个性化设计选择，具有独创性特征。

综合上述因素，法院认为原告虽无法提供源代码，但已提交足以形成完整证明链条的比对证据。被告未能说明其软件系独立开发，也未能合理解释上述相同缺陷及特征的来源，因此法院认定被诉软件与原告软件构成实质性相似，侵权成立。

2.4.3 案例启示

本案确立了在无法直接比对源代码的情况下，可通过软件运行过程中的缺陷及异常现象推定实质性相似。双方在运行中出现的相同缺陷成为关键比对依据，有效反映代码层面的一致性，法院依此认定侵权事实。本案的裁判方法被最高人民法院在指导案例第 49 号中采纳，成为实践中广泛引用的判例。

3. 结语

综上所述，我国软件著作权侵权认定的司法实践，已在“接触+实质性相似”原则的框架下，逐步形成一套兼具体系化与灵活性的分析模式。在实质性相似判断中，法院普遍吸收了域外的 SSO 标准与 AFC 三步测试法理念，通过结构化比对、层级过滤与综合分析，实现了对软件独创性表达的精准识别与合理保护。

从典型案例看，司法实践正从“代码数量比”向“表达独创性”转变，从“静态比对”向“动态运行逻辑”拓展，在事实与技术条件之间寻求平衡。这一模式兼顾技术特点与司法可操作性，为软件著作权保护提供了明确路径。