ICS 35.240.40
CCS A 11

JR

中华人民共和国金融行业标准

JR/T 0289—2024

金融业开源技术　术语

Open source technology applied in financial industry—Terminology

2024－01－15 发布　　2024－01－15 实施

中国人民银行　　发布

目　　次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则　第1部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中国人民银行科技司提出。

本文件由全国金融标准化技术委员会（SAC/TC 180）归口。

本文件起草单位：中国人民银行科技司、北京金融科技产业联盟、中国工商银行股份有限公司、中国农业银行股份有限公司、中国银行股份有限公司、中国建设银行股份有限公司、上海浦东发展银行股份有限公司、深圳前海微众银行股份有限公司、中国平安保险（集团）股份有限公司、中国银联股份有限公司、网联清算有限公司、北京国家金融标准化研究院有限责任公司、华为技术有限公司、腾讯云计算（北京）有限责任公司、麒麟软件有限公司、阿里云计算有限公司、中信建投证券股份有限公司。

本文件主要起草人：李伟、陈立吾、周祥昆、詹志建、刘帅、潘润红、聂丽琴、胡达川、李寻、余冠宇、孙刚、刘阳、吴冕冠、蔡仕志、闫晓林、刘建珍、王丽静、黄凯、金磐石、李鑫、杨欣捷、弓豪怡、钟燕清、丛洋、龙凯、唐天浩、贺伟平、周继恩、吕伊蒙、弓祎斌、杨阳、郭林、薛松源、吴涛、周夕崇、谢彦丽、张晋钰、李佳凝、薄舜添、白阳、邱成锋、胡正策、耿航、董宾、陈明、王悦良、胡伟琪、王晶昱、许哲。

引　言

近年来，开源技术在金融业各领域得到广泛应用，在推动金融机构科技创新和数字化转型方面发挥着积极作用。在金融业广泛应用开源技术的过程中，一方面由于开源技术部分基础术语缺少标准定义，相关标准制定过程中出现术语重复定义；另一方面，在不同的应用场景下，因金融机构对开源技术部分概念仍存在不同的表述，造成金融机构间沟通交流时对开源技术的实质认知不同，影响开源技术应用和金融业务需求表述。因此，有必要制定统一的金融业开源技术术语标准，形成行业广泛认可的术语定义，促进快速形成共识，为金融业制定开源技术相关标准及规范性文件提供参考。

本文件通过收集现有国家标准、行业标准及国际标准中信息技术、开源等方面的术语，经过分析归纳，结合金融业开源技术应用特点，形成本文件。

金融业开源技术　术语

1　范围

本文件界定了金融业开源技术的常用术语。

本文件适用于金融业中涉及开源技术的相关标准及规范性文件制定和信息沟通等活动。

2　规范性引用文件

本文件没有规范性引用文件。

3　术语和定义

3.1　基础类

3.1.1

开源　open source

将源代码公开提供的一种开放协作形式。

注：在遵守开源许可证（3.2.1）要求的情况下，源代码通常能修改和重新分发（3.2.8）。

3.1.2

软件　software

与计算机系统操作有关的计算机程序、规程、规则，以及相关文件、文档和数据。

[来源：GB/T 11457—2006，2.1469，有修改]

3.1.3

开源软件　open source software

一种可以获取源代码的软件。

注：1.开源软件的著作权持有人通过开源许可证将软件的复制、修改、再发布和商业应用等权利向公众开放。

2.部分开源许可证允许开源软件用于商业目的，存在商业版本和技术支持收费等形式，因此开源软件不等同于免费软件。

3.1.4

开源技术　open source technology

直接引入或间接引入并应用于金融信息系统的开源软件产品及相关技术服务的统称。

注：1.开源软件产品包含开源代码、开源组件、开源软件等，及前述产品的开源许可证（3.2.1）要求开源的衍生品、商业版本开源软件；技术服务包含免费或采购的基于前述产品的服务，例如云服务等。

2.开源软件产品引入方式包含从代码托管平台（3.3.1）、开源社区（3.5.1）、官方网站等渠道直接获取，或通过商业采购（3.6.2）、合作研发（3.6.3）等形式间接获取。

3.1.5

开源协作　open source collaboration

在开源软件项目建设过程中，多个机构、个人之间通过互联网等渠道协调和配合的开发模式。

3.1.6

源[代]码 source code

以适合于作为汇编程序、编译程序或其他转换程序输入的形式表示的计算机指令和数据定义。
注：源程序由源代码构成。
[来源：GB/T 11457—2006，2.1541]

3.1.7

目标代码 object code

由汇编程序或编译程序输出的格式表示的计算机指令和数据定义。
示例：目标代码的表现形式有软件开发工具包、动态链接库等。
[来源：GB/T 11457—2006，2.1031，有修改]

3.1.8

开源代码 open source code

公开可见、可用的软件源代码。
注：能用于满足开源许可证要求的研究、编辑、传播等行为。

3.1.9

开源基础软件 open source fundamental software

为信息系统应用提供基础服务，通常可独立运行的开源软件。
示例：开源操作系统、开源数据库、开源中间件等。

3.1.10

社区版本开源软件 community edition of open source software

对开源社区的开源代码集成、编译，进行公开或正式发布的可运行的软件。
注：通常能通过开源社区免费获取，但可能缺乏商业配套服务。

3.1.11

商业版本开源软件 commercial edition of open source software

由厂商基于开源软件进行开发发布的软件。
注：1.通常较社区版本功能更丰富，可能增强了部分功能或特性，受商用版权保护。一般由厂商提供收费的配套服务，例如售后服务、技术支持。
2.商业版本开源软件的代码通常也是公开的，使用时遵循开源许可证要求。

3.1.12

长期支持版开源软件 long-term support edition of open source software

可在较长周期中得到持续维护及更新的开源软件。
注：长期支持版（Long-Term Support，LTS）一般被认为是开源软件的最稳定版本，不同开源社区和软件也常使用稳定（stable）、发布（release）等作为表达最稳定版本的标记。

3.1.13

闭源软件 proprietary software

通常不公开提供源代码，需在版权所有者的专有法律权利许可下使用的计算机软件。

注：1.闭源软件一般只给予被许可方在一定的条件下使用该软件的权利，不做其他修改、共享、学习、再分发或逆向工程等用途。
2.部分开源许可证允许开源软件在修改后不再公开提供源代码，形成闭源软件。

3.1.14

商业软件 commercial software

基于闭源软件或开源软件的商业产品。
注：用户为实现商业功能而采购的软件或计算机程序，主要用于提高生产效率或实现具体功能。

3.1.15

停服 end of life；EOL

开源软件发行方停止对软件进行维护支持的处理。
注：开源软件停服后，用户无法获得官方升级和补丁等服务，使用停服的开源软件可能影响应用系统的安全稳定运行。

3.1.16

开源组件 open source component

构成某个开源软件的最小可识别、可评估的组成元素。
注：有时开源软件开发过程中引用的其他开源代码文件或其他开源代码片段可能被识别为某个开源组件，并纳入开源软件物料清单（3.4.8）。

3.1.17

开源工具 open source tool

在开发、测试、运维等环境下作为某种专用工具使用的开源软件。
示例：开发测试工具软件。

3.2 规则类

3.2.1

开源许可证 open source license

用于规范受著作权保护的软件在规定条款、条件下被使用或分发等行为的协议。
注：1.一般指具备广泛认可性的、具有法律性质的协议，也称开源协议，目的是减少作者针对开源软件的使用授权与使用者责任的法律解释成本，常见开源许可证主要包括宽松许可证（3.2.2）、著佐权许可证（3.2.3）和弱著佐权许可证（3.2.4）。
2.开源许可证条款通常包括开源代码的修改、分发、用途、专利授权、使用商标等方面的要求与限制。

3.2.2

宽松许可证 permissive license

对使用开源代码的方式与范围设置了最小限制的协议。
注：1.宽松许可证通常允许用户自由地使用、修改和重新分发开源代码。
2.宽松许可证能用于具有专利的独立软件作品，且该许可证下的衍生品允许闭源。

3.2.3

著佐权许可证 copyleft license

在对源代码的修改、衍生品中必须沿用原开源许可证，并且不得附加其他额外限制的协议。
注：该类许可证通过设置特殊的版权条款，防止开源软件成为闭源软件，用户在遵循该许可证要求下，享有自由复制、分配和修改源代码的权利。

3.2.4

弱著佐权许可证 weak copyleft license

允许在一定条件下对源代码的修改、衍生品中的源代码或部分源代码使用其他开源许可证的协议。

注：与著佐权许可证的条款要求相比，弱著佐权许可证通常对开源软件衍生品的要求较为宽松。

3.2.5

披露要求 notice requirement

开源许可证条款中，对开源软件或其部分代码被分发时应披露相关信息的要求。

注：不同类型开源许可证要求披露的信息有所不同，通常要求分发者对软件是否含有开源代码、原始开源许可证文本、作者信息、版权标记进行披露。

3.2.6

贡献者许可协议 contributor license agreement；CLA

对贡献者向开源软件版权所有人或所属运营管理组织付出劳动时，其产出物的知识产权进行权利主张的协议。

注：贡献者许可协议主要目的为避免知识产权纠纷。

3.2.7

开源软件衍生品 derivative work of open source software

基于开源代码进行再次创作的作品。

注：1.开源软件衍生品包括对全部或部分开源代码进行修改、重写、翻译、注释、组合或与之链接（包括动态链接和静态链接）而形成的作品。

2.通过进程间通信或系统调用源代码的作品通常视为独立作品，不属于衍生品。

3.2.8

分发 distribution

对开源软件进行传播的行为。

注：传播的行为通常为公开发布、介质转移等，仅一方内部使用而不提供给外部其他人或组织时，通常不被视为分发。

3.2.9

公平合理非歧视原则 fair reasonable and non-discriminatory；FRAND

一种对开源软件专属权所有者的权利作出限制的承诺条款。

注：遵守FRAND承诺即同意公平、合理和无歧视地将开源软件必要专利授权许可给专利实施者使用，以防止垄断及滥用许可，维护合理竞争。

3.3 技术类

3.3.1

代码托管平台 code hosting platform

主要面向开源项目存储、管理、维护源代码，促进项目协同开发的网络托管平台。

注：通常也具备供非开源代码托管的功能。

3.3.2

代码仓库 code repository

用于组织软件项目的仓库。

注：代码仓库能存放软件项目的文件、代码等数据，代码托管平台包含多个代码仓库。

3.3.3

代码托管 code hosting

将开源软件项目交由代码托管平台管理的过程。

3.3.4

制品仓库 artifact repository

用于存储和管理由源代码编译、打包或由第三方提供的制品的仓库。

注：制品仓库可帮助使用方构建最终可执行的软件，通常包含制品的基本信息、引入方及使用方等信息。

3.3.5

星标 project star

对开源软件代码仓库进行标记，表达用户对该项目的关注、支持及赞赏的功能。

注：通常以星号进行标记，可以用于表示开源软件项目的受欢迎情况，在一定程度上反映该项目的质量。

3.3.6

关注 watching

接收开源软件项目动态，实时跟进项目代码提交申请、议题、评论等信息的功能。

注：对项目进行关注的数量能一定程度上反映出开发者对项目的关注程度。

3.3.7

复刻 fork

用户在代码托管平台上复制一份原软件代码仓库所有内容副本的操作。

注：1.复刻主要用于用户的自行开发、演进，以创建不同的项目，或者为了贡献代码到被复制的原项目。

2.复刻也称分叉，分叉数可反映项目对开发者的吸引度。

3.3.8

代码拉取 code pull

用户从代码托管平台上拉取最新代码的过程。

注：用于将其他开发者已提交至代码仓库的最新代码更新到使用场景中。

3.3.9

拉取请求 pull request

将复刻代码仓库中的变更申请拉取至被复刻的原始代码仓库的操作。

3.3.10

合并请求 merge request

将分支中的变更申请合并至另一分支的操作。

3.3.11

主干 master

源代码的主要控制版本。

注：主干也称为主分支，用于保存和记录整个开源项目已完成的功能，以及部署生产环境，开发人员通常不直接在主干上修改代码。

3.3.12

分支 branch

基于主干创建的代码副本。

注：通过创建分支与主干分离的形式进行新功能的开发、测试、修复，可防止多方协作开发时互相干扰。

3.3.13

议题 issue

用于追踪开源软件代码的提问、反馈、任务或缺陷的互动功能。

3.3.14

代码提交 code commit

用户将新增或变更的代码提交到本地代码仓库，使本地代码仓库与本地最新进度同步的操作。

3.3.15

代码推送 code push

用户将本地代码仓库的变更推送到远程代码仓库，使远程代码仓库与本地代码仓库同步的操作。

注：代码推送到远程仓库后其他用户即可通过代码拉取获得代码。

3.3.16

代码冲突 code conflict

用户执行代码提交时，所修改的内容与代码仓库中其他用户已提交内容存在冲突的情况。

3.3.17

代码审核 code audit

由某主体借助某种工具对源代码进行的独立审查。

注：代码审核目的包括验证其是否符合软件设计文件和程序设计标准，并对正确性和有效性进行估计。

[来源：GB/T 11457—2006，2.219，有修改]

3.3.18

代码评审 code review

将软件代码呈现给项目人员、管理人员、用户、客户或其他感兴趣的人员，用于评价、审议等用途的过程。

[来源：GB/T 11457—2006，2.224，有修改]

3.3.19

代码发布 code release

从托管平台上拉取发行版本代码的功能。

3.3.20

标签 tag

标记某次代码提交或发布的功能。

注：通过标签功能，能快速、精准定位并查看代码提交记录，常用于开源软件版本发布记录管理。

3.3.21

特征 feature

用于向用户展示开源软件某一版本发布时，相比前版本在内容、特色等方面变化的描述。

3.3.22

安全漏洞 security vulnerability

开源软件在设计、编写、配置过程中存在的缺陷。
注：安全漏洞能使系统或应用数据的保密性、完整性、可用性、可控性等面临威胁。

3.3.23

后门 backdoor

绕过安全性控制而获取对程序或系统访问权的程序方法。

3.3.24

内源 inner source

借鉴开源软件开发模式在机构内部建立的跨部门、跨岗位的软件协作开发机制。
注：内源模式下，源代码的使用或分发等行为一般不通过开源许可证进行规范，以机构内部规章制度进行对应管理。

3.4 评估类

3.4.1

社区成熟度 community maturity

衡量开源社区常见发展模式下达到的阶段，用于表示该社区所涉及的技术随着时间的发展，社区价值总量的膨胀和收缩情况。

3.4.2

项目成熟度 project maturity

衡量开源项目整体发展水平、资源完备程度的综合评价指标。
注：综合评价指标包括社区成熟度、企业投入、商业模式、市场情况（例如供应商与商业解决方案）等。

3.4.3

开发活跃度 development activity

衡量开源社区或项目开发活动的活跃程度的综合评价指标。
注：综合评价指标包括开发者数量、合并次数、代码提交次数、修复缺陷数、新功能增长数、分支数量、咨询解答数量等，可用于预测社区或项目的发展潜力。

3.4.4

社区贡献度 community contribution

衡量个人或组织对开源项目贡献量的评价指标。
注：社区贡献度通常能通过代码贡献量、代码贡献占比、解答咨询次数等指标进行量化。

3.4.5

评估指标 assessment indicator

用以衡量、评估或判断事物质量、过程能力是否满足其规定准则、特性的测度。
[来源：GB/T 11457—2006，2.90，有修改]

3.4.6

评估工具 assessment tool

在评估中所用的一个或一组工具。
注：用以帮助评估者评价过程性能或能力、处理评估数据和记录评估结果。
[来源：GB/T 11457—2006，2.92，有修改]

3.4.7

开源软件成分分析 open source software composition analysis

识别开源软件构建详情及供应链关系的方法。
注：通过扫描开源软件程序、二进制文件、生成开源软件物料清单等方式，展示开源软件的最小可识别项、组件间依赖性、安全漏洞、开源许可证等信息，分析评估开源软件安全与识别风险。

3.4.8

开源软件物料清单 open source software bill of material

展示构成开源软件最小可识别信息的文件。
注：开源软件物料清单可提供开源软件中依赖的其他组件、版本、作者、开源许可证等信息，主要用于开源软件成分分析、评估，以降低开源软件供应链风险。

3.4.9

兼容性 compatibility

a）当2个或2个以上系统或部件共享相同的硬件或软件环境，执行其所要求的功能时可得到同样结果的能力。
b）2个或2个以上系统或部件交换信息的能力。
c）同一软件不同版本可共存的能力。
[来源：GB/T 11457—2006，2.249，有修改]

3.4.10

许可证兼容性 license compatibility

代表某一种开源许可证下，代码组合、合并的能力。
注：合并代码时遵循各个代码所应用的开源许可证要求，但一些开源许可证包含与其他开源许可证不兼容的权利条款，通常来说，著佐权许可证兼容性低，宽松许可证兼容性高。

3.4.11

开源许可证遵从性 open source license compliance

代表用户、开发者对开源软件的应用、分发、贡献及维护等行为遵守开源许可证规定条款、条件的程度。

3.4.12

可靠性 reliability

与预期行为和结果相一致的特性。
[来源：GB/T 29246—2017，2.62]

3.4.13

可扩展性 extensibility

系统或部件能修改以增加其存储或功能的能力情况。
[来源：GB/T 11457—2006，2.594，有修改]

3.4.14

可维护性 maintainability

a）通过修改软件系统或部件达到排除故障、改进性能或其他属性来适应变更环境的容易程度。
b）硬件系统或部件可保持或恢复到能履行其功能的状态的容易程度。
[来源：GB/T 11457—2006，2.903，有修改]

3.4.15

易用性 usability

软件产品在指定条件下运行时，其被用户理解、学习、使用的能力。

3.4.16

代码更新频率 code update rate

开源项目在一定时间段内，其代码进行增加、删除、修改的行数、次数的数据值。
注：代码更新频率能衡量该项目的开发效率、进展情况。

3.4.17

依赖性 dependency

软件系统或部件正常运行需要调用其他软硬件系统的功能或服务而与之产生的关联关系。
注：1.能通过可替换性、隔离能力及程度，确定依赖性强弱。
2.依赖性能放大开源软件供应链风险的影响。

3.4.18

脆弱性 vulnerability

可能被一个或多个威胁利用的资产或控制的弱点。
[来源：GB/T 29246—2017，2.89]

3.4.19

开源软件供应链 open source software supply chain

为满足开源软件产品和服务的供应关系，通过投入资源、参与开发或应用等过程，将供方、需方、参与方等相互链接的网链结构。
注：多方之间根据开源许可证、服务协议或其他约定建立连续的供应关系。
[来源：GB/T 36637—2018，3.4，有修改]

3.4.20

开源软件供应链风险 open source software supply chain risk

利用开源软件存在的开源许可证变更、停服、访问限制、安全漏洞等脆弱性，导致开源许可证违约、稳定性低、无法访问代码托管平台等供应链安全合规事件的可能性，及其由此对组织造成的影响。
[来源：GB/T 36637—2018，3.5，有修改]

3.5 生态类

3.5.1

开源社区　open source community

项目开发的组织形式。

注：开源社区是由所有参与开发和改进开源项目的用户组成的社群，在网站、邮件群组等形式组成的虚拟社区、代码托管平台中进行交流、协作开发及成果分享。

3.5.2

上游社区　upstream community

维护开源软件主干版本的开源社区。

3.5.3

开源社会组织　open source social organization

为推动开源软件高质量发展而组建的社会团体、基金会等组织。

注：常见的开源社会组织主要为产业联盟、开源基金会等非营利性组织，承担中立性运营管理职责，主要职责包括建立开源社区、提供代码托管、法务及财务支持等。

3.5.4

项目管理组织　project management organization

负责保证开源社区运转良好、决定与社区相关的重大事项并制定相关规章的开源项目核心管理团队。

注：1.通常以成立项目管理委员会的形式开展有关工作，并为下属组织给出指导意见。

2.部分大型开源社区的项目管理组织通过下设技术委员会、品牌委员会、生态委员会等组织进行细化运营治理。

3.5.5

技术管理组织　technical management organization

负责项目的技术路线规划、指导、监督、决策和技术资源协调的开源项目核心管理团队。

注：1.通常以成立技术委员会、技术监督委员会或技术指导委员会的方式开展有关工作。

2.根据开源社区运营治理模式与发展阶段的不同，部分开源社区会将项目管理组织职责整合纳入技术管理组织。

3.5.6

贡献者　contributor

对开源软件以某种方式做出贡献行为的个人或法人。

注：贡献行为包括但不限于问题解答、撰写文档、提交代码、捐款。

3.5.7

维护者　maintainer

提交者　committer

拥有对代码审核决策的权限、负有审查和合并来自贡献者的贡献内容的职责并负责项目运维的人员。

注：当开源社区发展成熟后，维护者与提交者通常承担不同职责，例如维护者为项目的规划和设计者，提交者为项目的实际开发者，其提交的代码需要经过维护者的审批才能被合并到代码仓库。

3.5.8

第三方机构　third party institution

不直接参与源代码开发、维护或修改，但可围绕该产品提供衍生服务、产品的机构。

注：例如提供配套设备、咨询服务、人员支持的机构。

3.6 其他类

3.6.1

软件获取 software acquisition

从其他组织通过文档化的协议获得软件的过程，或从其他组织获取软件产品的一组活动。
[来源：GB/T 30998—2014，3.1.25]

3.6.2

商业采购 commercial procurement

以合同方式有偿取得软件或其相关工程与服务的行为。
注：有偿获取方式包括购买、租赁、委托等。

3.6.3

合作研发 collaborative research and development

研发主体通过协议的形式与其他主体共同参与产生智力成果创作的研发模式。
注：1.研发主体及其他主体通常需对项目的某一个关键领域分别投入资金、技术、人力等资源，共同完成项目研发。
2.合作研发共同完成的知识产权，其归属由协议约定。

3.6.4

选型 lectotype

根据实际需求对实施方案、软硬件及所涉及的技术进行评估与选择的活动。

3.6.5

交付 delivery

软件研制周期中，将产品提交客户、预定用户或计划中的用户，并由其使用或接受的阶段。
[来源：GB/T 11457—2006，2.430，有修改]

3.6.6

风险管理 risk management

指导和控制相关风险的协调活动，识别、控制、消除或最小化可能影响系统资源的不确定因素的过程。
注:典型的风险管理包括风险评估、风险应对、风险容忍及风险交流（在决策者和承担者之间交换和分享风险信息）。
[来源：GB/T 25069—2022，3.168，有修改]

3.6.7

软件生命周期 software life cycle

软件产品从概念构思到退役的演变。
注：典型软件生命周期包括需求阶段、设计阶段、实现阶段、测试阶段、安装和验收阶段、操作和维护阶段、退出阶段。
[来源：GB/T 8566—2022，3.1.26，有修改]

3.6.8

开源软件生命周期管理 open source software life cycle management

金融机构应用开源软件时从引入、使用、更新、退出以及过程中进行评估、维护及风险管理的时间周期。

3.6.9

可信来源　trusted source

经评估认为值得信赖，具有可靠性、安全性的实体或资源渠道。

注：可信来源通常具备一定的信息过滤和验证措施，所提供的产品或信息能够被确认为合法安全，一般指经过认证的网站、经合法授权的软件服务商、机构内部自建代码仓库、经过安全扫描的镜像源等渠道。

参 考 文 献

[1] GB/T 5271.14—2008 信息技术 词汇 第14部分：可靠性、可维护性与可用性
[2] GB/T 8566—2022 系统与软件工程 软件生存周期过程
[3] GB/T 11457—2006 信息技术 软件工程术语
[4] GB/T 15733—1995 金融电子化基本术语
[5] GB/T 25069—2022 信息安全技术 术语
[6] GB/T 29246—2017 信息技术 安全技术 信息安全管理体系 概述和词汇
[7] GB/T 30998—2014 信息技术 软件安全保障规范
[8] GB/T 35273—2020 信息安全技术 个人信息安全规范
[9] GB/T 36637—2018 信息安全技术 ICT供应链安全风险管理指南
[10] ISO/IEC 5230:2020 信息技术 开放链规范（Information technology—OpenChain Specification）