数据自治开放的软件开发和运行环境

吴毅坚[1,2]，陈士壮[1,2]，葛佳丽[1,2]，赵文耘[1,2]
1. 复旦大学计算机科学技术学院，上海 201203；2. 上海市数据科学重点实验室，上海 201203

摘要
数据自治开放是数据开放的一种新形式，能保护数据拥有者的合法权益。数据自治开放这一新的数据开放模式要求软件开发和运行环境能从应用的需求描述、应用开发和调试、应用运行监控等方面进行全方位的支持。给出了支持数据自治开放的软件开发和运行环境的总体设计，阐述了支持软件行为管控的数据自治开放应用开发全过程，从而实现数据自治开放中数据的全生命周期管理，说明了建设数据自治开放的软件开发运行基础设施的可行性。

关键词
数据开放；数据自治开放；软件开发环境；软件运行环境；软件行为管控

中图分类号：TP311.5　　文献标识码：A　　doi: 10.11959/j.issn.2096-0271.2018016

Software development and runtime environment for self-governing openness of data

WU Yijian[1,2], CHEN Shizhuang[1,2], GE Jiali[1,2], ZHAO Wenyun[1,2]
1. School of Computer Science, Fudan University, Shanghai 201203, China
2. Shanghai Key Laboratory of Data Science, Shanghai 201203, China

Abstract
Self-governing openness of data is a new form of the openness of data, which enables protecting interests of data owners. Self-governing openness of data requires full supports of software development and runtime environment in the fields of requirements description, development and debugging, and monitoring applications execution. An overall design of the software development and runtime environment for self-governed openness of data was presented, and the whole application development process for self-governing openness of data that supports software behavior monitoring and control was illustrated. Full life-cycle management of data in self-governing openness of data was enabled and the feasibility of the software development and runtime infrastructure for self-governing openness of data was shown.

Key words
open data, self-governing openness of data, software development environment, software runtime environment, software behavior monitoring and control

1 引言

近年来，许多国家纷纷通过建立数据开放平台推动数据开放和共享，以便公众能够及时、快捷、准确地获取数据资源[1]。通常，这些平台要求数据拥有者以XML、CSV、XLS等格式把数据上传到平台。数据使用者可以根据需求从数据平台下载数据，然后按照数据平台的要求使用数据。然而，是否能完全遵守这类数据使用要求则很大程度上由数据使用者决定，数据拥有者无法控制其提供数据的复制、传播和使用。数据治理权的削弱和丧失很大程度上导致数据拥有者不愿开放数据。

数据自治开放是数据开放的一种新思路和新模式。在数据自治开放环境下，数据拥有者将保持对数据的治理权，拥有的数据将通过系统化的受控机制开放给潜在的数据用户。潜在的数据用户通过开放的渠道获取数据的基本信息，并通过数据自治开放软件平台获取经过封装的数据实体，在本地或云端按照约定的方式受控地使用数据。数据用户可以查看部分数据内容（如果数据拥有者允许），并利用数据开展数据分析和开发数据应用，但无法对数据个体或全体进行复制或传播。

与传统的数据开放不同，数据自治开放环境在支持数据开放的同时，注重保持数据稀缺性和保护数据权益。数据拥有者一方面希望通过持有数据获得数据的潜在价值，另一方面又可能没有能力获取这种潜在价值。这种数据价值困境削弱了数据拥有者开放数据的动机，直接影响了数据潜在价值的挖掘。如何通过技术手段，在保护数据拥有者权益的同时，又能充分发掘数据潜在价值，是面向数据自治开放的软件环境面临的挑战之一。

数据价值的挖掘依赖于软件，因此软件的开发和运行环境是否有足够的能力支持数据使用的全生命周期，直接影响数据自治开放的可行性、有效性和效率。为了提升数据利用的效率，数据拥有者仅提供数据，而读取并处理数据的软件则由数据使用者自行开发（即外部软件）。然而，由于大数据分析处理的创新性和不确定性，难以预先精确限定外部软件的数据处理方式和具体行为，因此需要对外部软件实际访问和处理数据进行必要的管理和监控，才能确保数据不被非法复制和传播，从而保护数据的稀缺性和数据拥有者的合法权益。

为了有效地管理和监控外部软件行为、支持数据使用者在数据自治开放环境下开发数据分析软件，需要研究支持数据治理权控制的软件开发和运行环境，对数据自治开放中数据获取、外部软件的开发调试和部署运行、数据使用等环节提供数据和软件的全生命周期的支持保障。本文从数据自治开放中数据使用方式变化的角度，分析了面向数据自治开放的软件开发和运行支持的总体需求，阐述了数据封装体的基本设计思想；给出了基于数据封装体的软件开发和运行环境进行架构设计的思路，阐述其主要组成部分的意义和功能，说明了在该环境下数据使用和软件开发的全过程；讨论了构建该环境的关键技术挑战。

2 相关研究

在软件开发平台方面，各大云开发商都在研发相应的大数据开发平台。例如阿里云推出了“数加平台”，提供大数据计算服务、大数据开发套件等大数据应用开发和运行基础设施。然而，这些开发平台主要针对一般性的大数据应用开发，而不关注数据本

身的权益保护和数据稀缺性的保持。

软件行为管控是数据自治开放中的重要环节，通过监控软件访问数据的行为来保护数据不被非法复制和传播。在信息安全领域，有大量的关于软件行为分析的研究，通过动态[2,3]和静态[4-6]的方法追踪敏感数据流，从而防止敏感数据的泄露。还有研究通过对常见的数据源和数据宿进行分类[7]，规定软件的合法行为序列[8]，对比软件的实际执行轨迹，判定软件行为是否符合要求[2]，从而发现是否有数据的非法使用[9]。然而，这些工作主要关注敏感数据泄露，并且大多应用在移动应用的检测上，而并不关注对完整数据集的泄露保护。另外，对软件行为日志的分析也是软件行为和意图分析的一个重要辅助手段[10-13]。

从数据本身的保护方面来看，数据访问控制[14,15]是一种主要的数据泄露防控手段。然而，目前的做法主要是通过技术手段限制数据访问，例如加密、授权、安全策略、信任级别[16]等，内容也主要关注隐私保护[17]，因此这些方法并不完全适用于数据开放环境。

由此可见，要从根本上解决数据开放环境下受控的数据访问，保持数据的稀缺性，必须从访问数据的软件入手，研发针对数据稀缺性保护的软件开发和运行平台，确保软件对数据的访问过程是受控的，并且同时具有一定的灵活性，以支持各类大数据分析和数据密集型应用需求[18,19]。

3 数据自治开放的软件开发和运行环境

3.1 数据自治开放环境的数据访问特点和需求

数据自治开放是控制和限制数据使用者传播或滥用数据的新型数据开放模式。在数据自治开放环境下，数据使用者无法再像传统的数据开放或共享那样，得到完整的数据资源，并对数据资源进行直接的、不受数据拥有者或第三方（如基础平台软件）控制的访问，因此也无法任意复制传播。但同时，数据使用者又能根据自身的需求，在数据拥有者的“授权”（按申请的方式使用数据的权利，但不是处置权或治理权）下自由使用数据。这就要求有一个支持数据使用且对数据使用有适当管控的软件环境，将数据封装和保护起来，同时又具备计算能力满足数据使用者传统的数据使用（处理和分析等）需求。

自治开放中的数据是以带有自主程序单元和内在计算能力的封装体的形式发布和使用的。这种数据封装体是数据自治开放的软件开发和运行环境使用的最基础的数据单元。数据使用者能获得的数据只能存在于数据封装体中，并且只能进行自己声称的数据访问行为。超出预先声称意图的数据访问行为必须被检测、监控、报警或拒绝；必要时，数据封装体中的自主程序单元还可对数据进行销毁（擦除），从而避免数据的非法复制传播。

本文将针对这种数据访问特点和管控需求，给出数据自治开放的软件开发和运行环境的总体设计架构，并论述基于该架构的数据应用开发全过程。

3.2 总体架构

数据自治开放的软件开发和运行环境是一个以数据封装体（带有自主程序单元和内在计算能力的数据存储形式，数据被灌装入数据封装体中）为基本数据组织和运算的开发运行一体化平台，分为相对独立的开发平台和运行平台两个部分。开发平台支持基于数据封装体的外部软件开发

和调试，运行平台支持外部软件的测试、运行和管控，同时两部分又通过数据封装体的基本设计密切结合，一体化地支撑数据密集型应用的开发和运行。

图1展示了基于数据封装体的软件开发和运行环境的总体架构。为了便于阐述，**图1**中还包括了用于管理数据封装体的管理平台。在软件开发和运行环境中，外部软件所需的所有数据可来自于数据拥有者的自主存储或者公有存储，并通过数据管理平台封装到数据封装体中。外部软件的开发人员需要根据自身的业务需求和数据访问需求，通过数据封装体管理平台提出数据访问申请，并由管理平台将数据存储中的数据灌装到数据封装体中，以供程序开发调试使用。具体开发流程将在第3.3节中论述。

开发平台主要由一个集成开发环境（integrated development environment，IDE）构成。该IDE基于Eclipse架构，扩展了新的面向数据封装体开发的多种设计视图，用于支持数据封装体应用整个开发生命周期中不同开发环境的需求。例如，在数据封装体应用开发的启动阶段，开发人员需要向数据站提交业务需求，此时要用到应用开发需求视图和资源列表视图，便于开发人员书写需求并查看可用的数据封装体目录；而在数据封装体应用开发初期，开发人员可能希望看到数据封装体定义的细节以及一些样本数据，此时则要用到数据封装体视图。各类视图将数据封装体应用开发过程中的需求组织起来，以便开发人员针对不同开发需求自行切换。该IDE还集成了程序运行时开发包，将数据封装体访问、外部数据分析算法以及在数据站端的运行支持接口等集成在开发环境中，方便开发者的程序开发工作。该IDE提供了专门针对大数据应用开发问题而重新设计的调试环境，用于解决由于数据质量参差不齐而出现的实际数据内容与数据封装体标称数据结构产生差异的问题。外部程序开发人员根据获取的样本数据封装体编写和调试面向数据封装体的应用程序，并通过样本数据封装体进行调试，才能在程序中正确处理样本数据封装体中出现的各类问题。调试环境除了提供传统集成调试环

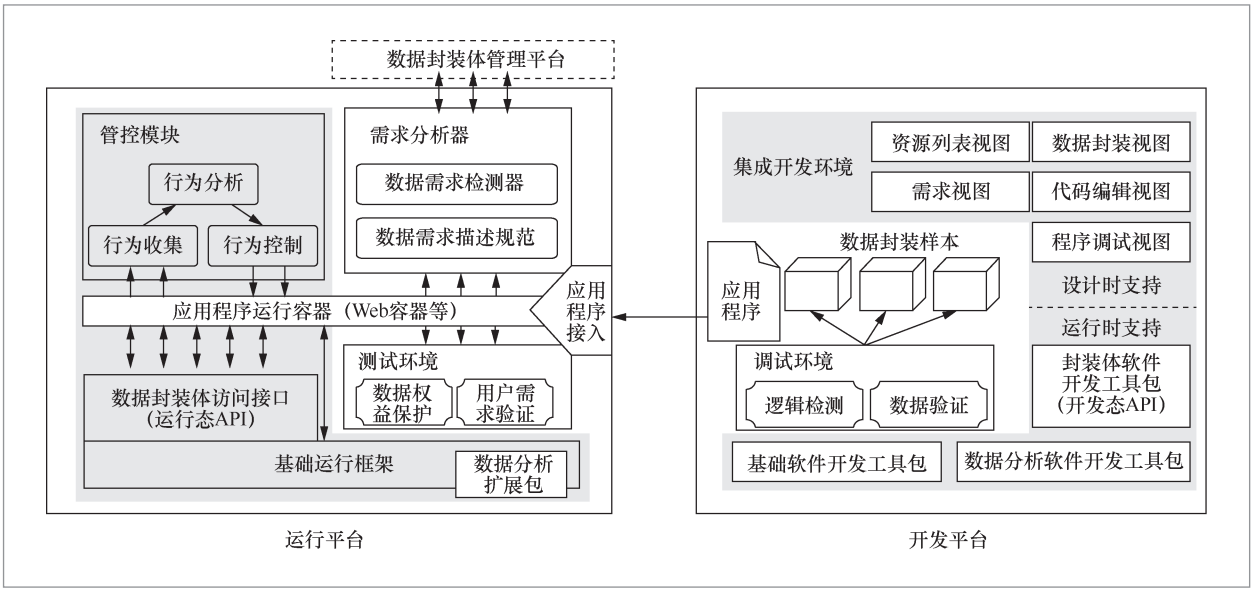


图1 数据自治开放的软件开发和运行环境的总体架构

境以外，还特别针对数据分析过程进行优化，例如对数据计算的中间结果进行检查、根据数据规格说明验证数据封装体的数据实例等；而对于平台而言，可以通过调试对代码进行动态分析，从而对关键数据的处理逻辑进行检查。

运行平台是外部程序在服务器环境中运行的基础软件支撑，主要包括以下几个方面。

- 基本的应用程序运行容器，如Tomcat等Web应用运行容器或Java运行，提供了应用程序运行的基本环境。
- 基础运行框架：是外部应用程序在运行容器中运行时所需的面向数据封装体的接口支持，如标准的数据封装体访问接口以及集成的标准数据分析方法等。
- 管控模块：基于数据封装体实时访问日志以及运行容器日志，收集外部程序的数据访问行为，并进行实时和准实时的分析，然后根据行为管控策略给出相应的数据行为控制。
- 需求分析器：用于在开发初期分析选择合适的数据封装体以及在运行阶段作为数据访问行为异常的参考。
- 测试环境：任何一个面向数据封装体的外部应用在开发完成后、投入正式运行之前，由应用程序接入模块送入测试环境进行相应的测试，包括用户视角的用户需求验证和平台视角的数据权益保护。

3.3 支持软件行为管控的数据自治开放应用开发全过程

支持软件行为管控的数据自治开放应用开发过程如图2所示，主要分为提交需求、获取数据封装体样本、本地开发程序、调试程序、提交程序、测试程序以及正式运行7个步骤。

(1) 提交需求

与普通的应用开发不同，面向数据封

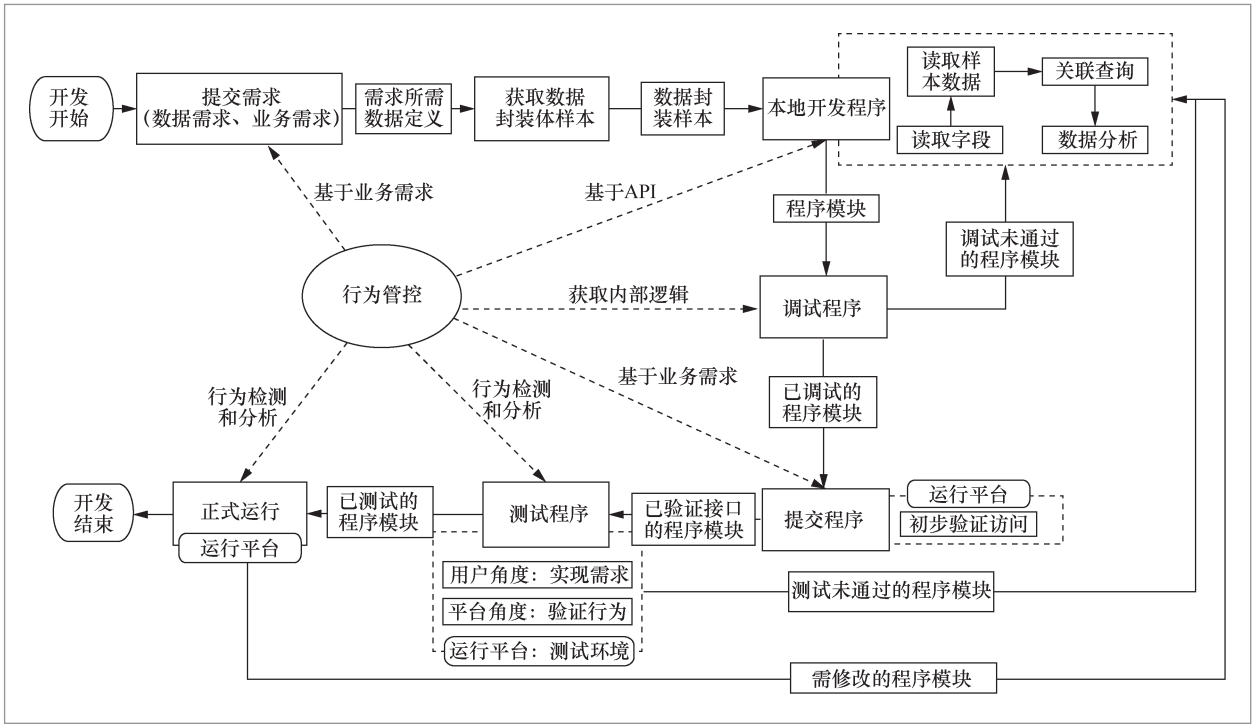


图 2 支持软件行为管控的数据自治开放应用开发全过程

装体的应用开发由于需要对软件的行为进行必要的管控，因此需要将待开发应用的业务需求和数据需求提交给运行平台中的需求分析模块，并在运行平台中对该软件开发任务进行注册。后续的开发过程将同时在开发平台和运行平台中记录。

（2）获取数据封装体样本

运行平台中的需求分析模块在收到程序开发请求后，通过数据管理平台获得符合相关需求的样本数据封装体，并分发给开发者。样本数据封装体是对所需数据的部分采样，仅供开发人员开发程序时使用。样本数据封装体的行为管控可以设置得较弱，而且开发人员为了充分调试程序，甚至可多次请求样本数据封装体。

（3）本地开发程序

在完成样本数据封装体的获取后，开发人员就可利用传统的开发技术自行开发数据封装体应用。通常，开发人员会首先读取数据封装体的实际数据结构，并按照实际的数据结构尝试读取数据。在熟悉样本数据后，结合实际业务需求，编写数据查询的代码，并完成数据的分析。

（4）调试程序

在程序的开发过程中，开发人员需要利用集成开发环境中的集成调试器进行程序调试。开发人员一般将着重检查和验证开发程序对样本数据处理的正确性。与传统的单纯调试程序不同，在面向数据封装体的应用开发中，平台本身也可通过程序的调试运行获取关键数据的使用逻辑，从而为后续正式程序的软件行为检测和管控收集信息。在调试程序的过程中，当程序的行为与预期不一致时，开发人员随时可以重新修改程序，并再次进行调试。

（5）提交程序

由于最终的程序将在运行平台上执行，所以通常情况下，开发人员在本地开发完成并通过调试的程序，需要通过运行平台的应用程序接入模块加载到运行容器中执行。在此过程中，应用程序接入模块根据需求分析模块记录的开发任务ID，找到程序在提交需求步骤中留下的数据访问信息，并据此信息初步验证程序对数据访问的合法性。

（6）测试程序

在面向数据封装体的应用开发过程中，系统级的测试是必不可少的环节。在此环节中，新开发的程序在隔离的沙箱环境中运行。开发人员可提交必要的测试规则，使得程序在正式大规模高负荷运行之前，仍然有机会在服务器环境下验证其数据处理的正确性。而对于运行平台而言，通过测试一方面可以预先检查数据访问行为是否符合预期，是否与其声称的访问目标一致，另一方面也可预判该程序的正式运行是否会对整个运行平台的稳定运行造成不利影响。程序出现异常或平台判定程序访问数据的言行不一致，都可以导致测试失败，并驱动外部程序返回集成开发环境进行修正。因此，测试程序这一环节，在面向数据封装体的数据自治开放软件开发环境中具有非同寻常的意义。

（7）正式运行

当开发人员和平台都认可测试结果后，程序将进入正式的运行状态。考虑到大数据应用的数据量大、计算方法多样，尽管之前经过了调试和测试，但当程序面临真实数据时，仍然可能产生各种管控问题。因此，运行平台中的管控模块将持续对正式运行的程序进行行为检测和分析，确保程序按既定方案正常使用数据。

4 关键技术挑战

4.1 软件行为管控

外部软件行为管控需要在外部软件

开展正式数据访问之前，通过监控外部软件访问软件的行为，提取软件访问数据的行为特征，并基于这些特征抽象出其高层意图。相关研究被广泛应用在软件行为监测、隐私防护、恶意软件检测方面。

外部软件行为管控是数据自治开放中保障数据权益的重要环节。在数据自治开放环境下，外部软件如何能充分获得访问数据的自由，同时又能保证数据的权益不受侵害，是一个重要的研究挑战。

外部软件行为管控主要分为使用数据前的管控技术和使用数据过程中的管控技术两个方面。使用数据前这一阶段主要包括数据使用者获取数据和获取数据后开发外部软件的若干活动。在获取数据时，数据使用者需要提供使用数据的有关需求和意图，这是保护数据治理权、防止数据滥用的基本条件。数据使用者的需求和意图只有在不侵犯数据拥有者数据治理权的条件下，才可获得批准，并开展后续的数据使用活动。数据使用者获批使用数据后，需要自行开发软件来使用数据（如处理和分析，但是不能复制和传播）。而开发调试软件过程仍然不是正式的数据使用过程，而是仅通过部分或者示例性的样本数据进行。该过程中的软件行为管控主要是在数据使用者初步完成开发软件后，以开发运行环境对软件进行静态代码分析为主、动态轨迹收集为辅的方式，初步验证是否符合（不抵触）之前声称的需求和意图。

使用数据过程中的管控主要是对正式上线运行的外部软件的访问数据等行为进行收集、分析、决策和控制。通过监控外部软件访问数据的行为，提取出软件访问数据的行为特征，并基于这些特征抽象出其高层意图，与数据使用者预先声称的需求和意图进行比较，从而决定是否允许后续的数据访问行为。主要通过程序动态分析让软件在受控环境中运行，收集软件运行时对数据实际的处理操作，并分析得到软件的整体行为意图。该过程需要同时考虑管控的效果和效率，并且在实现中根据实际管控需求作出相应的权衡。

4.2 高层业务需求描述与建模

对外部程序的数据需求和业务需求的描述和建模是软件行为管控的基础。在面向数据封装体的软件开发开始前和完成后，都需要尽可能准确地描述开发程序对数据的实际需求以及程序的高层业务需求。而现实中，准确定义需求已经是公认的软件工程的难题，并且由于软件开发任务的高度复杂性和现实环境快速变化等多种因素的影响，需求的易变性也是导致难以准确描述业务需求的一大障碍。然而，在数据自治开放环境下，由于数据需求可以相对清晰、较好地体现软件开发者的设计思路，因此，充分利用数据需求和领域知识对数据使用者的意图进行描述，是一个重要的突破口。

外部软件访问开放的数据资源时，应当表明其访问数据资源的高层意图。例如某外部软件声称为了追踪病症A的治疗和患者愈后情况，需要访问该病症的所有医疗数据，那么根据这一意图，对与病症A“概念相关”的数据资源的访问（可能）都是符合其意图的。这种概念相关性依赖于特定业务领域知识模型以及对开放数据资源的语义标注。在外部软件访问开放数据资源时，对其所有数据访问行为和曾访问过的数据资源语义进行分析，对外部软件访问数据资源的实际意图进行建模，并通过实际意图与其生成的高层意图的比较来评价数据访问行为的风险。

4.3 面向数据封装体的软件开发工具包的设计

软件开发工具包(software development kit，SDK)是数据封装体应用开发的基础。由于大数据应用开发的复杂性和不确定性，一些软件实现算法往往无法内置于开发环境中，而只能由开发者自行编写。这就产生了如何将外部编写的算法逻辑安全可靠地运行在服务器端，并确保数据权益不受外部程序损害的问题。因此，需要一个基础开发框架为外部软件的开发提供足够的可扩展性。同时，对于一些常见的数据分析算法，则希望有一些内置的实现可以给开发人员方便地使用。另外，数据封装体本身的访问也需要在开发平台中实现，但开发平台仅处理样本数据封装体，因此需要数据封装体访问接口的开发态和运行态两个不同的权限控制版本。可见，对基础开发框架、可扩展的数据分析算法的实现以及对数据封装体本身的访问，都是开发运行平台需要解决的问题。

(1)基础开发框架SDK

基础开发框架SDK提供了外部程序在本地和运行平台中运行的基础接口类。这些基础接口类是外部应用程序代码逻辑的重要扩展点。当外部应用开发者需要在应用开发时使用自有的特定算法时，应当将这些算法根据基础开发框架SDK中规定的框架进行设计编码，实现相应的数据处理接口，从而实现从特定结构输入数据到特定结构输出数据的转化。

之所以需要规定特定的数据输入和输出接口，是出于对数据保护的考虑。当外部程序用自身算法进行计算时，涉及的原始数据都是对应用程序开放的。然而，在这种情况下，缺乏对应用程序的行为管控及对数据权益的保护是非常危险的。因此，在基于数据封装体的外部应用开发中，应用开发模块对数据的读写都必须通过给定的接口完成，从而通过控制数据流入和流出情况来保护数据。

(2)数据分析SDK

数据分析SDK以外部软件包的形式提供常用的数据分析算法。这些开发包对于整个开发环境而言是开放的，可以由任何第三方根据基础开发框架的标准要求进行扩展。与数据封装体内置的基本分析接口不同，数据分析SDK可以提供更加复杂的计算逻辑，并且需要通过基础开发框架SDK和数据封装体SDK才能访问数据封装体内的数据。尽管复杂的计算逻辑可以通过外部程序开发者根据基础框架SDK自行开发，但内置的数据分析SDK提供了常用的分析算法实现，可简化外部程序开发人员的投入。

(3)数据封装体SDK

数据封装体SDK提供了外部程序访问数据封装体中数据的基本操作规范，即数据访问接口。在数据自治开放环境中，当数据站中的任何数据提供给外部程序使用时，均会灌装到数据封装体中；而外部程序对数据封装体中数据的访问，必须通过数据封装体SDK提供的数据封装体访问接口。

数据封装体SDK将提供受控的体内数据访问能力和基本的数据分析能力。数据访问能力是指，当外部程序需要获取数据封装体内的原始数据时，能通过数据封装体提供的接口调用数据SDK中的数据获取方法，读取原始的数据记录。这种原始数据的读取能力受到数据封装体内置的安全机制的限制，例如，当原始数据的读取超过预设的限制时，数据访问请求将被拒绝。基本的数据分析能力主要包括简单

的数据统计操作，如求指定字段取值的最大值、最小值、算术平均值、中位值、方差等。同样，数据分析能力也受到封装体内安全机制的限制，用于保护原始数据不被过度泄露。

4.4 离线开发调试和在线运行验证

为了便于开发人员使用与业务需求相关的各类数据，数据封装体提供了数据的基本规格说明。数据使用方利用数据的规格说明在本地编写适用于获得的数据封装体的软件代码。用户开发集成环境内置数据封装体访问接口，供数据用户进行本地调试使用。为了方便开发和保护数据权益，在开发前，运行平台生成一些带有部分样本数据的数据封装体，使得开发人员可以利用数据封装体开发接口进行开发调试。

这种离线开发调试的方式能处理一部分数据质量问题。但是，在大数据应用环境下，数据质量不高、数据内容与规格说明不一致等问题层出不穷，因此即便本地调试通过的程序，仍然很可能在真实运行环境中发生错误而需要停机调试。

在正式运行中，为了提升处理停机问题的效率，需要一种既保护数据不被意外泄露，又能方便用户在本地调试的实现方案。数据封装体的开发态应用程序编程接口（application programming interface，API）的离线版本提供了一种重要的在线本地调试方式，当在线数据使用发生问题、产生错误时，可通过本地接口获得出错数据的样本，并且配置一定的容错性，使接口能够一次尽可能多地返回可能的数据异常的问题。与此同时，这一特性还需要和数据防泄露机制配合，防止因运行时故障导致原始数据泄露的情况发生。

5 结束语

面向数据封装体的软件开发与运行环境，是支持数据自治开放的软件开发和运行的重要软件基础设施。本文给出了这种新型软件开发与运行环境的总体架构，并阐述了基于数据封装体的数据自治开放软件开发过程。数据自治开放的软件开发要求开发环境和运行环境的无缝衔接，进一步突出了开发、运行一体化的要求。数据封装体作为支持数据自治开放的新型载体，要求软件开发和运行环境提供相应的支撑，包括解决数据需求描述、数据访问行为管控、数据封装体应用的本地开发和远程运行等一系列的关键技术挑战。在后续工作中，将逐步对面向数据封装体的软件开发和运行环境的各个技术挑战开展进一步的研究，建立支持数据自治开放的软件研发基础设施。

参考文献:

[1] 周大铭. 我国政府数据开放现状和保障机制[J]. 大数据, 2015, 1(2): 19-30.
ZHOU D M. Status and safeguard mechanisms of Chinese government data opening[J]. Big Data Research, 2015, 1(2): 19-30.

[2] CLAPP L, ANAND S, AIKEN A. Modelgen: mining explicit information flow specifications from concrete executions[C]// ISSTA, July 14-17, 2015, Baltimore, USA. New York: ACM Press, 2015: 129-140.

[3] XU H, ZHOU Y F , GAO CY , et al. SpyAware: investigating the privacy leakage signatures in app execution traces[C]//ISSRE, November 2-5, 2015, Gaithersbury, USA. Piscataway: IEEE

Press, 2015: 348-358.

[4] LI L, BARTEL A, BISSYANDE T F, et al. IccTA: detecting inter-component privacy leaks in android Apps[C]// ICSE, May 16-24, 2015, Florence, Italy. Piscataway: IEEE Press, 2015: 280-291.

[5] KLIEBER W, FLYNN L, BHOSALE A, et al. Android taint flow analysis for app sets[C]//The 3rd ACM SIGPLAN International Workshop on the State of the Art in Java Program Analysis (SOAP'14), June 9-11, 2014, Edinburgh, UK. New York: ACM Press, 2014: 1-6.

[6] FENG Y, ANAND S, DILLIG I, et al. Apposcopy: semantics-based detection of android malware through static analysis[C]//The 22nd ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE 2014), November 16-22, 2014, Hong Kong, China. New York: ACM Press, 2014.

[7] RASTHOFER S, ARZT S, BODDEN E. A machine-learning approach for classifying and categorizing Android sources and sinks[C]// Network and Distributed System Security Symposium, February 23-26, 2014, San Diego, USA. [S.l:s.n], 2014.

[8] KRKA I, BRUN Y, POPESCU D, et al. Using dynamic execution traces and program invariants to enhance behavioral model inference[C]// ICSE, May 2-8, 2010, Cape Town, South Africa. New York: ACM Press, 2010: 179-182.

[9] GAFNY M, SHABTAI A, ROKACH L, et al. Detecting data misuse by applying context-based data linkage[C]//The 2010 ACM Workshop on Insider Threats, October 8, 2010, Hyatt Regency, USA. New York: ACM Press, 2010: 3-12.

[10] 张兴科. 数据挖掘在Web日志分析中的应用[J]. 微处理机, 2009, 30(3): 80-83.
ZHANG X K. The study on data mining algorithm and application in Web log analysis[J]. Microprocessors, 2009, 30(3): 80-83.

[11] 赵龙, 江荣安. 基于Hive的海量搜索日志分析系统研究[J]. 计算机应用研究, 2013, 30(11): 3343-3345.
ZHAO L, JIANG R A. Research of massive searching logs analysis system based on Hive[J]. Application Research of Computers, 2013, 30(11): 3343-3345.

[12] 朱金清, 王建新, 陈志泊. 基于APRIORI的层次化聚类算法及其在IDS日志分析中的应用[J]. 计算机研究与发展, 2007, 44(s3): 326-330.
ZHU J Q, WANG J X, CHEN Z B. Apriori-based hiberarchical clustering algorithm and its application on IDS log analysis[J]. Journal of Computer Research and Development, 2007, 44(s3): 326-330.

[13] ZAMORA J, MENDOZA M, ALLENDE E. Query intent detection based on query log mining[J]. Journal of Web Engineering, 2014, 13(1-2): 24-52.

[14] YANG K, JIA X H, REN K. Secure and verifiable policy update outsourcing for big data access control in the cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(12): 3461-3470.

[15] YANG K, JIA X H, REN K, et al. DAC-MACS: effective data access control for multi-authority cloud storage systems[J]. IEEE Transactions on Information Forensics and Security, 2013, 8(11): 1790-1801.

[16] ROOS A, DRÜSEDOW S, HOSSEINI M I, et al. Trust level based data storage and data access control in a distributed storage environment[C]//The 8th IEEE International Conference on Mobile Cloud Computing (MobileCloud), June 27-July 2, 2015, New York, USA. Piscataway: IEEE Press, 2015: 169-176.

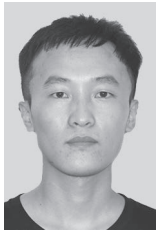
[17] 方滨兴, 贾焰, 李爱平, 等. 大数据隐私保护技术综述[J]. 大数据, 2016, 2(1): 1-18.
FANG B X, JIA Y, LI A P, et al. Privacy preservation in big data: a survey[J]. Big Data Research, 2016, 2(1): 1-18.

[18] 高丰. 开放数据: 概念、现状与机遇[J]. 大数据, 2015, 1(2): 9-18.
GAO F. Open data: concept, current

situation and opportunity[J]. Big Data Research, 2015, 1(2): 9-18.
[19] 朱扬勇, 熊赟. 大数据是数据、技术, 还是应用[J]. 大数据, 2015, 1(1): 71-81.
ZHU Y Y, XIONG Y. Defining big data[J]. Big Data Research, 2015, 1(1): 71-81.

作者简介

吴毅坚(1979-), 男, 博士, 复旦大学副教授, 主要研究方向为软件维护与演化、大数据应用开发平台。



陈士壮(1991-), 男, 复旦大学硕士生, 主要研究方向为软件工程、数据开放共享。

葛佳丽(1993-), 女, 复旦大学硕士生, 主要研究方向为软件工程、数据开放共享。

赵文耘(1964-), 男, 复旦大学教授, 主要研究方向为软件工程、企业应用集成、软件开发平台。

收稿日期: 2018-01-26
基金项目: 上海市科技发展基金资助项目(No.16JC1400801)
Foundation Item: Shanghai Science and Technology Development Fund (No.16JC1400801)