用新型D-BUS与Linux桌面应用程序通讯 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/145/2021_2022__E7_94_A8_E6_96_B0_E5_9E_8BD_c103_145350.htm D-BUS 是一个大有前途的消息总线和活动系统，正开始深入地渗透到 Linux&reg. 桌面之中。了解创建它的原因、它的用途以及发展前景。 D-BUS 本质上是 进程间通信（inter-process communication）（IPC）的一个实现。不过，有一些特性使得 D-BUS 远远不是“ 只是另一个 IPC 实现 ”。有很多不同的 IPC 实现，因为每一个都定位于解决特定的明确定义的问题。CORBA 是用于面向对象编程中复杂的 IPC 的一个强大的解决方案。DCOP 是一个较轻量级的 IPC 框架，功能较少，但是可以很好地集成到 K 桌面环境中。SOAP 和 XML-RPC 设计用于 Web 服务，因而使用 HTTP 作为其传输协议。D-BUS 设计用于桌面应用程序和 OS 通信。 桌面应用程序通信 典型的桌面都会有多个应用程序在运行，而且，它们经常需要彼此进行通信。DCOP 是一个用于 KDE 的解决方案，但是它依赖于 Qt，所以不能用于其他桌面环境之中。类似的，Bonobo 是一个用于 GNOME 的解决方案，但是非常笨重，因为它是基于 CORBA 的。它还依赖于 GObject，所以也不能用于 GNOME 之外。 D-BUS 的目标是将 DCOP 和 Bonobo 替换为简单的 IPC，并集成这两种桌面环境。由于尽可能地减少了 D-BUS 所需的依赖，所以其他可能会使用 D-BUS 的应用程序不用担心引入过多依赖。 桌面/操作系统通信 术语“ 操作系统 ”在这里不仅包括内核，还包括系统后台进程。例如，通过使用 D-BUS 的 udev（Linux 2.6 中取代 devfs 的，提供动态 /dev 目录），当设

备（比如一个 USB 照相机）插入时会发放出一个信号。这样可以更紧密地将硬件集成到桌面中，从而改善用户体验。
D-BUS 特性 D-BUS 有一些有趣的特性，使其像是一个非常有前途的选择。 协议是低延迟而且低开销的，设计得小而高效，以便最小化传送的往返时间。另外，协议是二进制的，而不是文本的，这样就排除了费时的序列化过程。由于只面向本地机器处理的使用情形，所以所有的消息都以其自然字节次序发送。字节次序在每个消息中声明，所以如果一个 D-BUS 消息通过网络传输到远程的主机，它仍可以被正确地识别出来。 从开发者的角度来看，D-BUS 是易于使用的。有线协议容易理解，客户机程序库以直观的方式对其进行包装。 程序库还设计用于为其他系统所包装。预期，GNOME 将使用 GObject 创建包装 D-BUS 的包装器（实际上这些已经部分存在了，将 D-BUS 集成入它们的事件循环），KDE 将使用 Qt 创建类似的包装器。由于 Python 具有面向对象特性和灵活的类型，已经有了具备类似接口的 Python 包装器。 最后，D-BUS 正在 freedesktop.org 的保护下进行开发，在那里，来自 GNOME、KDE 以及其他组织的对此感兴趣的成员参与了设计与实现。 D-BUS 的内部工作方式 典型的 D-BUS 设置将由几个总线构成。将有一个持久的 系统总线（system bus），它在引导时就会启动。这个总线由操作系统和后台进程使用，安全性非常好，以使得任意的应用程序不能欺骗系统事件。还将有很多 会话总线（session buses），这些总线当用户登录后启动，属于那个用户私有。它是用户的应用程序用来通信的一个会话总线。当然，如果一个应用程序需要接收来自系统总线的消息，它不如直接连接到系统总线 不过，它可以

发送的消息将是受限的。一旦应用程序连接到了一个总线，它们就必须通过添加 匹配器（matchers） 来声明它们希望收到哪种消息。匹配器为可以基于接口、对象路径和方法进行接收的消息指定一组规则（见后）。这样就使得应用程序可以集中精力去处理它们想处理的内容，以实现消息的高效路由，并保持总线上消息的预期数量，以使得不会因为这些消息导致所有应用程序的性能下降并变得很慢。对象 本质上，D-BUS 是一个对等（peer-to-peer）的协议 每个消息都有一个源和一个目的。这些地址被指定为 对象路径。概念上，所有使用 D-BUS 的应用程序都包括一组 对象，消息发送到或者发送自特定对象 不是应用程序 这些对象由对象路径来标识。另外，每个对象都可以支持一个或多个 接口（interfaces）。这些接口看起来类似于 Java 中的接口或者 C 中的纯粹的虚类（pure virtual classes）。不过，没有选项来检查对象是否实现了它们所声明的接口，而且也没有办法可以调查对象内部以使列出其支持的接口。接口用于名称空间和方法名称，因此一个单独的对象可以有名称相同而接口不同的多个方法。消息 在 D-BUS 中有四种类型的消息：方法调用（method calls）、方法返回（method returns）、信号（signals）和错误（errors）。要执行 D-BUS 对象的方法，您需要向对象发送一个方法调用消息。它将完成一些处理并返回一个方法返回消息或者错误消息。信号的不同之处在于它们不返回任何内容：既没有“信号返回”消息，也没有任何类型的错误消息。消息也可以有任意的参数。参数是强类型的，类型的范围是从基本的非派生类型（布尔（booleans）、字节（bytes）、整型（integers））到高层次数据结构（字符串（strings）、数组

（arrays）和字典（dictionaries））。 服务 服务（Services）是 D-BUS 的最高层次抽象，它们的实现当前还在不断发展变化。应用程序可以通过一个总线来注册一个服务，如果成功，则应用程序就已经 获得 了那个服务。其他应用程序可以检查在总线上是否已经存在一个特定的服务，如果没有可以要求总线启动它。服务抽象的细节 尤其是服务活化 当前正处于发展之中，应该会有变化。 用例 尽管 D-BUS 相对较新，但是却迅速地得到了采用。如前所述，可以构建具有 D-BUS 支持的 udev 以使得当热插拔（hot-plug）设备时它可以发送一个信号。任何应用程序都可以侦听这些事件并当接收到这些事件时执行动作。例如，gnome-volume-manager 可以检测到 USB 存储棒的插入并自动挂载它；或者，当插入一个数码相机时它可以自动下载照片。 一个更为有趣但很不实用的例子是 Jamboree 和 Ringaling 的结合。Jamboree 是一个简单的音乐播放器，它具有 D-BUS 接口，以使得它可以被告知播放、到下一首歌、改变音量等等。Ringaling 是一个小程序，它打开 /dev/ttyS0（一个串行端口）并观察接收到的内容。当 Ringaling 发现文本 “ RING ” 时，就通过 D-BUS 告知 Jamboree 减小音量。最终的结果是，如果您的计算机上插入了一个调制解调器，而且电话铃响，则音乐音量就会为您减小。这 正是计算机所追求的! 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com