Introducció a D-Bus

D-Bus (https://www.freedesktop.org/wiki/Software/dbus) és una API que es pot fer servir en el desenvolupament de qualsevol programa d'usuari per tal d'aconseguir comunicar el seu procés amb altres procesos també compatibles amb D-Bus (bé de forma punt-a-punt o bé multicast) però, i això és important, executant-se tots a la mateixa màquina. És a dir, D-Bus és un mecanisme IPC (Inter-Process Communication) -com podria ser, a un nivell molt més simplificat, una canonada-. O dit d'una altra manera: D-Bus és una manera que tenen els processos per parlar entre sí intercanviant-se ordres, notificacions d'events i missatges en general (tot plegat escrits d'una determinada forma) a través de determinats canals (els "busos") que són oferts per un servei anomenat "dbus".

Generalment hi ha dos busos principals, un anomenat "system" (l'estat del qual es pot consultar amb *systemctl status dbus*) i un altre anomenat "session" (l'estat del qual es pot consultar amb *systemctl status --user dbus*). Tots dos mantinguts pel binari *dbus-daemon* però amb paràmetres diferents.

*El bus "system" és un únic canal per on es poden comunicar totes les aplicacions/serveis del sistema entre sí indistintament de la sessió d'usuari amb la què hagin sigut iniciades. Per exemple, se sol fer servir per monitoritzar quan les tarjes de xarxa es (des)activen, quan els discos externs es (des)endollen, quan la bateria del portàtil està a nivell baix, etc.

*De busos "session" poden haver-hi més d'un creats a la vegada perquè per cada sessió d'usuari diferent iniciada al sistema se'n crea un bus "session" diferent (però només és visible el bus "session" pertanyent a la sessió d'usuari actual). Serveix per comunicar entre sí les aplicacions posades en marxa per l'usuari en qüestió. Per exemple, reproductors de vídeo poden enviar un missatge D-Bus evitant que s'activi el salvapantalles mentre l'usuari està veient una pel·lícula.

NOTA: Un altre mecanisme IPC modern i flexible (entre d'altres que hi ha) però que és molt interessant perquè no necessita cap dimoni funcionant és Varlink (https://varlink.org). Desgraciadament, encara no l'implementen moltes aplicacions

Per tal d'enviar un missatge a un bus D-Bus caldrà primer saber l'adreça de destí. Aquesta està estructurada d'una forma jeràrquica amb els següents elements:

*El nom del **servei** D-Bus (integrat dins de l'aplicació que l'implementi) que rebrà el missatge. Per tal de no solapar noms de diferents serveis, el que se sol fer és anomenar-los simplement revertint el FQDN del domini propietat del desenvolupador i afegint-hi llavors el nom de servei desitjat; per ex: *org.maemo.Alert* o *org.freedesktop.Notifications* D'aquesta manera, diferents desenvolupadors podran anomenar al servei D-Bus de la seva aplicació "Alert" o "Notifications" sense que hi hagi problemes

*Cada servei D-Bus pot contenir un o més "**objectes**", cadascun dels quals proporcionarà una funcionalitat diferent. Per identificar un objecte respecte d'un altre s'usen els "object paths", els quals són simples cadenes que tenen el mateix aspecte que les rutes absolutes de fitxers. En general, si el servei D-Bus només conté un objecte, el seu "path" sol ser el mateix nom del servei però escrit en forma de ruta i, opcionalment, afegint-hi al final un nom específic per l'objecte (és a dir, en el nostre exemple, "/org/maemo/Alert/Alerter" o /org/freedesktop/Notifications/Manager", per exemple).

*Cada objecte D-Bus implementa un conjunt "d'**interfícies**". Cada interfície representa un subconjunt determinat de les diferents ordres concretes que l'objecte pot executar, juntament amb els seus paràmetres i les possibles senyals a les què pot respondre. És possible reutilitzar la mateixa interfície en diferents objectes. El nom d'una interfície segueix les mateixes normes que els noms de serveis; per això per serveis senzills se sol repetir el seu nom pel nom de la interfície més rellevant.

*La darrera part de la direcció d'un missatge D-Bus és el nom del "**membre**" (en el cas de IPCs això se sol anomenar nom del "mètode" i en el cas de senyals, nom "de la senyal"; també hi poden haver "propietats" que són simples parelles clau<->valor editables o no). Aquest nom selecciona quin procediment executar o quina senyal a emetre. Només necessita ser únic dins de la interfície on estigui implementat.

Eines D-Bus

D-Bus proporciona una comanda oficial per experimentar amb l'enviament de diferents missatges (ja siguin membres de tipus IPCs o de tipus senyal): *dbus-send* . Els seus paràmetres més importants són:

--system	Indica que el missatge s'enviarà al bus "system". Per defecte (o amb el paràmetre *--session*) s'envia al bus "session"
--type=method_call	Indica que es farà una crida IPC (és a dir, una execució d'un mètode) en comptes d'enviar una consulta de lectura -o escriptura- d'una propietat (l'acció per defecte)
--print-reply	S'espera a rebre una resposta de l'execució de la IPC (o de la consulta del valor de la propietat) i mostra aquesta (si n'hi ha)
--dest=nom.servei	Obvi
/ruta/objecte	Obvi
nom.Interficie.nomMembre	Obvi. Noteu com el nom del membre s'ha "d'adjuntar" amb un punt (i sense espais) al final del nom de la interfície.

Un exemple de comanda, que en aquest cas estableix el valor a una propietat, seria aquest: *dbus-send --system --print-reply --dest=org.freedesktop.NetworkManager /org/freedesktop/NetworkManager org.freedesktop.DBus.Properties.Set string:"org.freedesktop.NetworkManager" string:"NetworkingEnabled" variant:boolean:true* on les paraules "string" o "variant" indiquen el tipus de dada que s'està indicant (cadena o calaix de sastre -"variant"- que es concreta en booleà) com a paràmetres de la crida en concret.

Una altra comanda que forma part del paquet oficial "dbus-tools" (igual que *dbus-send*) és *dbus-monitor*, la qual mostra al terminal en temps real tots els missatges D-Bus transferits al bus "*system*" (si s'afegeix el paràmetre *--system*) o bé "*session*" actual (si s'afegeix el paràmetre *--session*).

Una altra eina alternativa per treballar amb serveis D-Bus d'una manera més còmoda que amb *dbus-send* és la comanda *gdbus*, proporcionada pel projecte GTK/Gnome. Aquí es presenta un exemple d'execució RPC, on es poden veure els seus paràmetres més comuns (*--session* , *--dest*, *--object-path* i *--method*):

gdbus call --session --dest org.freedesktop.Notifications --object-path /org/freedesktop/Notifications --method org.freedesktop.Notifications.Notify miApp 42 gtk-dialog-info "Resum" "Text de la notificació" [] {} 5000

L'execució anterior correspon concretament a la del mètode Notify, pertanyent a la interfície org.freedesktop.Notificacion, el qual, tal com es pot veure, ha de rebre vuit paràmetres: nom de qui fa la RPC (cadena), un identificador opcional (número) , nom de la icona dins de "/usr/share/icons/gnome" a mostrar (cadena), títol mostrat a la notificació (cadena), text mostrat a la notificació (cadena), accions a realitzar un cop generada la notificació (no se sol usar; array de cadenes), característiques extra de la notificació (posició, so a emetre, etc; diccionari de cadenes) i el temps en milisegons durant el qual es mostra la notificació (número sencer). Més informació a https://developer.gnome.org/notification-spec

> NOTA: Fixeu-vos que podríem haver aconseguit el mateix simplement amb la comanda *notify-send "Resum" "Text de la notificació"*; de fet, en realitat, l'únic que fa aquesta comanda és realitzar la crida D-Bus a un servei->objecte->interfície->mètode concret (amb uns valors de paràmetres concrets) sense que l'usuari hagi de saber res de com funcionen els canals D-Bus interiorment. Fent servir *gdbus* directament el que estem fent és obviar la comanda *notify-send* i fer el mateix "a mà"

També es poden enviar (normalment a tots els processos que usin D-Bus tot i que també es pot enviar a un procés en concret) senyals (que han d'estar reconegudes per una determinada interfície d'origen per a què es puguin emetre) mitjançant *gdbus emit* . Les senyals serveixen per notificat als programes subscrits de què ha ocorregut cert event.

D'altra banda, també és interessant la comanda *gdbus introspect*, que permet estudiar la composició d'un objecte de forma similar al que fan les aplicació D-Feet (https://wiki.gnome.org/Apps/DFeet) o D-Spy (https://gitlab.gnome.org/GNOME/d-spy) gràficament (la primera de les quals estudiarem als exercicis).

> NOTA: Si no poguéssim usar D-Feet, per conèixer la llista de serveis D-Bus disponibles podríem executar per exemple la següent comanda: *dbus-send --session --type=method_call –print-reply --dest=org.freedesktop.Dbus /org/freedesktop/DBus org.freedesktop.Dbus.ListNames*

A més de la implementació oficial ("libdbus") i la que ofereix Gnome ("gdbus"), existeix una tercera alternativa per poder treballar amb D-Bus des del terminal: la proporcionada per Systemd ("sd-bus") en forma de comanda *busctl*. En concret les possibilitats bàsiques que proporciona aquesta darrera comanda són aquestes (per més informació llegiu l'article http://0pointer.net/blog/the-new-sd-bus-api-of-systemd.html):

busctl {--system | --user } call nom.Servei /nom/Objecte nom.Interfície mètode [signatures param1 param2 ...]
busctl {--system | --user } get-property nom.Servei /nom/Objecte nom.Interfície propietat
busctl {--system | --user } set-property nom.Servei /nom/Objecte nom.Interfície propietat signatura valor
busctl {--system | --user } introspect nom.Servei /nom/Objecte [nom.Interfície]

on "signatura" vol dir el tipus de paràmetre indicats, que pot indicar-se amb una "**i**" si és de tipus sencer amb signe, una "**u**" si és de tipus sencer sense signe, una "**s**" si és de tipus "string", "**b**" si és de tipus booleà, "**v**" si és de tipus "variant" (és a dir, calaix de sastre, útil per números decimals entre altres), "**ai**" si és un array de sencers, "**as**" si ho és de cadenes, etc. Per saber tots els tipus possibles, es pot consultar https://dbus.freedesktop.org/doc/dbus-specification.html#type-system

> NOTA: En el cas de tenir una signatura amb un array, abans de la llista dels valors dels seus elements caldrà indicar el número d'aquests. És a dir, així: *as 3 hola adeu pepe*
> NOTA: En el cas de tenir una signatura amb un valor "variant", això vol dir que el tipus del valor forma part del propi valor. Això a la pràctica significa que abans del valor en sí caldrà indicar el tipus específic d'aquest ("i", "u", "s", "b"...i, aquí sí, "d" si és un número decimal). És a dir, així: *v d 0.7*
> NOTA: Per defecte *busctl* empra sempre el canal de sistema (*--system*); per emprar el canal particular de l'usuari que l'invoca cal indicar explícitament el paràmetre *--user*

D'altra banda, es pot obtenir la llista d'objectes continguts en un determinat servei D-Bus amb la comanda *busctl tree nom.Servei*, i, a partir d'aquí, fer una instrospecció. D'altra banda, si s'executa *busctl list* o sense paràmetres (*busctl*), es mostrarà la llista de processos i serveis que fan ús de D-Bus

---

Per a què un servei D-Bus sigui reconegut com a disponible per les diferents aplicacions del sistema, aquest servei s'ha de definir en un arxiu "*.service" dins de la carpeta "**/usr/share/dbus-1/services**" o bé dins de "**/usr/share/dbus-1/system-services**". Les interfícies, d'altra banda, s'han de definir en arxius "*.xml" dins de la carpeta "**/usr/share/dbus-1/interfaces**". Per més informació es pot consultar *man dbus-daemon*. De totes maneres, la manera més fàcil de conèixer tots els serveis D-Bus disponibles al sistema o a la sessió juntament amb els objectes, interfícies i membres que implementen és fent ús, tal com ja hem dit, del programa gràfic **D-Fee**t.

> NOTA: Si es vol saber com desenvolupar un programa que publiqui un determinat servei D-Bus personalitzat fent servir la convenció Systemd, es pot llegir http://0pointer.net/blog/the-new-sd-bus-api-of-systemd.html o també https://zignar.net/2014/09/08/getting-started-with-dbus-python-systemd o https://leonardoce.wordpress.com/2015/03/11/dbus-tutorial-using-the-low-level-api

EXERCICIS:

1.-a) Instal.la a una màquina virtual qualsevol però amb entorn gràfic el programa D-Feet i utilitza'l per esbrinar, un cop posis en marxa el reproductor VLC (si no el tens instal·lat, llegeix la "nota" següent), quin és el nom del servei D-Bus que aquesta aplicació obre (al canal de sessió), quins són els objectes que implementa aquest servei, quines interfícies contenen aquests objectes i quins mètodes, propietats i senyals estan incloses en cadascuna de les interfícies.

> NOTA: Per instal.lar el VLC a l'Ubuntu només cal executar, com sempre, *sudo apt install vlc* però a Fedora, per un tema de llicencies, no està disponible ni als seus repositoris oficials i és per això que cal afegir uns repositoris extra-oficials anomenats "RPMFusion (free)" i "RPMFusion (non-free)" per poder-lo descarregar i instal.lar. Bàsicament el que cal fer és:
>
> *sudo dnf install https://download1.rpmfusion.org/free/fedora/rpmfusion-free-release-$(rpm -E %fedora).noarch.rpm*
> *sudo dnf install https://download1.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-$(rpm -E %fedora).noarch.rpm*
> *sudo dnf install vlc*

aII) ¿Quina informació obtens executant la comanda *busctl --user tree* i, més en concret, *busctl --user tree org.mpris.MediaPlayer2.vlc*? A partir d'aquí, ¿quina informació obtens executant la comanda *busctl --user introspect org.mpris.MediaPlayer2.vlc /org/mpris/MediaPlayer2* ? ¿ I la comanda *busctl --user introspect org.mpris.MediaPlayer2.vlc /org/mpris/MediaPlayer2 org.freedesktop.DBus.Properties* ?

b) Amb el VLC obert, digues què fa *busctl --user call org.mpris.MediaPlayer2.vlc /org/mpris/MediaPlayer2 org.freedesktop.DBus.Properties Set ssv "org.mpris.MediaPlayer2.Player" "Volume" d 0.7*

> NOTA: En el cas d'haver fet servir la comanda *gdbus*, l'últim valor s'ha d'escriure entre < i > perquè les dades "Variant" s'indiquen sempre amb < i > (i no caldrà indicar el tipus "d")

c) Reprodueix un vídeo qualsevol amb el VLC i a continuació omple els buits de la següent comanda per tal d'aconseguir pausar la reproducció: *busctl --user call ... ...* Pista: troba la resposta a D-Feet

d) ¿Quina comanda *busctl* similar a l'anterior hauries de fer servir ara per reprendre la reproducció pausada? Pista: troba la resposta a D-Feet

e) Executa el mètode adient amb *busctl* per tal de tancar el reproductor VLC. Pista: troba la resposta a D-Feet

Systemd fa servir D-Bus (el canal de sistema si és PID nº1 o el corresponent canal de sessió si és via paràmetre *–user*) per tal d'oferir la possibilitat de gestionar els serveis que administra a possibles tercers programes. De fet, la comanda *systemctl* el que fa en realitat és simplement realitzar les crides pertinents a aquest/s canal/s i no res més que això; és a dir, és simplement un "client" D-Bus (però especialitzat i molt convenient per l'usuari, això sí). Per demostrar-ho, l'exercici següent emula la funcionalitat d'aquesta comanda llençant de forma directa crides a D-Bus tal com ho faria ella.

2.-a) A la mateixa màquina virtual anterior (de fet, si no es diu res, tots els exercicis següents es faran en aquesta mateixa màquina), executa *sudo busctl call org.freedesktop.systemd1 /org/freedesktop/systemd1 org.freedesktop.systemd1.Manager StopUnit ss "cups.service" "replace"* ¿Què fa? (comprova-ho).

> NOTA: No facis cas del segon paràmetre del mètode StopUnit...és un valor intern que sempre valdrà "replace"

aII) ¿Per què passa el que passa si executes la comanda de l'apartat anterior sense escriure *sudo*? Pista: recorda el contingut i significat de l'arxiu "/usr/share/polkit-1/actions/org.freedesktop.systemd1.policy" vist a l'assignatura de Seguretat. D'aquí pots deduir que les peticions Polkit viatgen també pel canal D-Bus....

b) Després de llegir https://www.freedesktop.org/wiki/Software/systemd/dbus i amb l'ajuda de D-Feet, digues què fa la comanda *sudo busctl call org.freedesktop.systemd1 /org/freedesktop/systemd1 org.freedesktop.systemd1.Manager DisableUnitFiles asb 1 "cups.service" false* i quin és el significat dels seus paràmetres. Comprova-ho (pots fer-ho ràpidament mitjançant *systemctl is-enabled cups*)

> NOTA: Existeix un mètode contrari a *DisableUnitFiles* anomenat *EnableUnitFiles*, el qual té, a més dels dos mateixos paràmetres que aquell, un tercer paràmetre extra de tipus booleà i que sol valer sempre *true*

El dimoni *systemd-timedated* se sol emprar per atendre sota demanda peticions fetes des de la comanda *timedatectl*, les quals solen consistir en canviar manualment l'hora del sistema (o també, depenent del paràmetre emprat, la zona horària preestablerta) en absència d'un client NTP. Tots els paràmetres de la comanda *timedatectl* en realitat l'únic que fan és una crida D-Bus a un determinat mètode d'un servei->objecte-interfície determinat. L'exercici següent emula la funcionalitat de la comanda *timedatectl* fent la crida al D-Bus directament. Més informació: https://www.freedesktop.org/wiki/Software/systemd/timedated

3.- a) Busca a D-Feet el servidor D-Bus oferit pel dimoni *systemd-timedated* i digues com es diu

b) Sabent que el servei "org.freedesktop.timedate1" inclou l'objecte "/org/freedesktop/timedate1" que ofereix la interfície "org.freedesktop.timedate1" que conté la propietat "Timezone", escriu la comanda *busctl* adient per tal d'obtenir el seu valor al terminal

c) Sabent que el servei "org.freedesktop.timedate1" inclou l'objecte "/org/freedesktop/timedate1" que ofereix la interfície "org.freedesktop.timedate1" que conté el mètode "SetTimezone()" per canviar la zona horària del sistema, escriu la comanda *busctl* adient per tal de realitzar aquest canvi (i comprova tot seguit que s'hagi fet)

Un altre dimoni que proporciona un servei D-Bus és *systemd-hostnamed*. Aquest dimoni se sol emprar per atendre sota demanda peticions (via D-Bus) fetes des de la comanda *hostnamectl*, les quals solen consistir bàsicament en demanar el nom actual de la màquina o bé en voler canviar-ho. En altres paraules: la comanda *hostnamectl* en realitat no és més que un client D-Bus que realitza les peticions mitjançant aquesta via (ja siguin de consulta o d'actualització de dades) al servei *systemd-hostnamed*, el qual es posa en marxa només quan és rep la petició (no està tota l'estona funcionant). Això significa que la mateixa funció que podem fer amb la comanda *hostnamectl* es pot realitzar mitjançant qualsevol eina que permeti fer crides al canal D-Bus, com ara *dbus-send*, *gdbus* o la pròpia de Systemd, *busctl*. Més informació: https://www.freedesktop.org/wiki/Software/systemd/hostnamed

4.- Observa amb D-Feet (o *busctl tree/busctl introspect*) amb quin objecte, interfície i mètode necessitaries comunicar-te per canviar el nom actual de la màquina i utilitza la comanda *busctl* per construir la crida adient per fer-ho. Comprova-ho

Un altre dimoni (en aquest cas, no relacionat amb Systemd) que també proporciona un servei D-Bus és Upower. Aquest servei, sempre que estigui encés, permet enumerar els dispositius existents en una màquina capaços de proporcionar-ne energia (bateries, adaptadors AC, etc), a més de romandre a l'escolta d'events (endollament/desendollament, etc) i acumular historial i estadístiques de càrrega, consum, etc. La seva API D-Bus es troba documentada a https://upower.freedesktop.org/docs/ref-dbus.html

5.-a) Amb l'ajuda de D-Feet, escriu una comanda *busctl* que faci ús de l'API D-Bus proporcionada pel servei UPower (instal.la, si cal, el paquet "upower" per disposar d'aquest servei i inicia'l) per esbrinar si el teu sistema està rebent energia des d'una bateria o des d'un endoll.

b) ¿Quina seria la comanda *busctl* per esbrinar el nivell de càrrega de la bateria (suposant que s'estigués fent-la servir aquest tipus d'alimentació)?

Un altre dimoni (en aquest cas, no relacionat amb Systemd) que també proporciona un servei D-Bus és l'anomenat Accounts-daemon (https://github.com/freedesktop/accountsservice). Aquest servei permet obtenir i manipular informació sobre els usuaris del sistema mitjançant aquest canal

6.-a) Executa la comanda *sudo busctl --system call org.freedesktop.Accounts /org/freedesktop/Accounts org.freedesktop.Accounts CreateUser ssi usuari2 "Usuari de prova" 1* i comprova tot seguit, la sortida de la comanda *id usuari2* ¿Què veus i per què?

NOTA: El darrer paràmetre de la comanda *busctl* anterior indica si l'usuari creat és estàndard (valor 0) o administrador (valor 1). És per això que l'usuari "usuari2" té com a grup secundari el grup "sudo" (a Ubuntu) o "wheel" (a Fedora)

b) Executa la comanda *sudo busctl --system call org.freedesktop.Accounts /org/freedesktop/Accounts/User1001 org.freedesktop.Accounts.User SetPassword ss "$(openssl passwd -6 1234)" ""* (suposant que l'usuari creat a l'apartat anterior hagi obtingut l'UID 1001...si no, canvia aquest valor pel què correspongui) i comprova, tot seguit, la sortida de la comanda *sudo grep usuari2 /etc/shadow* ¿Què veus i per què?

c) Executa la comanda *sudo busctl --system get-property org.freedesktop.Accounts /org/freedesktop/Accounts/User1001 org.freedesktop.Accounts.User HomeDirectory* ¿Què veus?

d) Executa la comanda *sudo busctl --system get-property org.freedesktop.Accounts /org/freedesktop/Accounts/User1001 org.freedesktop.Accounts.User AutomaticLogin* ¿Què veus? ¿I si tornes a executar aquesta comanda després d'executar aquesta altra: *sudo busctl --system call org.freedesktop.Accounts /org/freedesktop/Accounts/User1001 org.freedesktop.Accounts.User SetAutomaticLogin b true* ? ¿Què estàs aconseguint amb això? Comprova-ho reiniciant la màquina. Un cop fet, però, torna a iniciar sessió amb l'usuari de treball.

Un altre dimoni (en aquest cas, no relacionat amb Systemd) que també proporciona un servei D-Bus és Udisks (*systemctl status udisks2*) el qual pertany al projecte global "Storaged". Aquest servei permet conèixer les propietats (i, en alguns casos, també manipular-les) de discos i dispositius d'emmagatzematge en general, tant a nivell de bloc com de taula de particions, sistemes de fitxers i punts de muntatge. La seva API D-Bus està documentada a http://storaged.org/doc/udisks2-api/latest

7.-a) Executa la comanda *udisksctl -b /dev/sda* ¿Què mostra? ¿I la comanda *udisksctl -b /dev/sda1*? ¿I la comanda *udisksctl -b /dev/sr0?* ¿I la comanda *udisksctl dump* ?

NOTA: Existeixen altres comandes interessants, com ara *udisksctl status*, *udisksctl monitor*, *udisksctl mount -b /dev/disp1 -t {ext4|vfat|...} -o opcions* (el punt de muntatge es crea dinàmicament sota "/run/media"; la gràcia d'aquesta comanda és que no cal ser "root" per fer el muntatge perquè funciona a través de Polkit), *udisksctl unmount -b /dev/disp1*, etc. També permet bloquejar/desbloquejar discos xifrats amb *udisksctl lock/unlock*, crear dispositius "loop" amb *udisksctl loop-setup*... Interessant mirar el contingut del fitxer de configuració "/etc/udisks2/mount_options.conf" i "/etc/udisks2/udisks2.conf"

El dimoni NetworkManager també ofereix una API D-Bus per permetre gestionar la configuració de les tarjes de xarxa del sistema d'una forma dinàmica (de fet, tant la comanda *nmcli* com el panell de control gràfic de Gnome la fan servir).

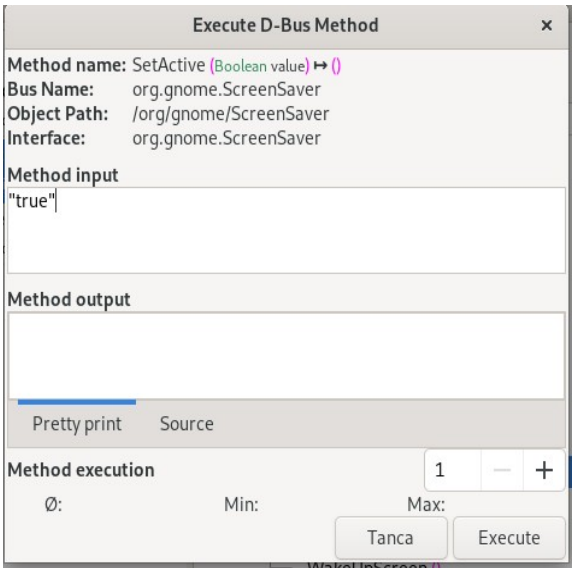
NOTA: Systemd-networkd també proporciona una API D-Bus però no és tan completa (encara)

8.-a) Descarrega l'arxiu zip disponible a la web del centre i descomprimeix-lo. Veuràs que conté quatre scripts que fan ús (mitjançant la comanda *dbus-send*) de l'API D-Bus proporcionada per NetworkManager. Prova cadascun d'ells i observa què fan. ¿Quin són els serveis, objectes, interfícies i mètodes que fan servir?

NOTA: Tens més exemples relacionats amb la gestió de NetworkManager (i on es fa servir igualment *dbus-send*) a http://cheesehead-techblog.blogspot.com/2012/09/dbus-tutorial-fun-with-network-manager.html

b) ¿Per què la següent comanda no funciona, tot i estar ben escrita i executar-la com a administrador: *busctl set-property org.freedesktop.NetworkManager /org/freedesktop/NetworkManager org.freedesktop.NetworkManager "NetworkingEnabled" b true ?* ¿Què hauries de fer per aconseguir el mateix objectiu? Pista: observa les característiques d'aquesta propietat mitjançant D-Feet.

9.-a) D-Fleet no només serveix per fer introspecció de membres. També és capaç d'executar mètodes. Fes doble clic sobre el mètode: "org.gnome.ScreenSaver" -> "/org/gnome/ScreenSaver" -> "org.gnome.ScreenSaver" -> "SetActive" del canal de sessió, escriu el paràmetre que necessita (és de tipus booleà) així: *"true"* (tal com mostra la imatge de sota) i clica sobre el botó "Executa". ¿Què passa?



NOTA: Existeix un altre mètode similar: "org.gnome.Shell.ScreenShield" -> "/org/gnome/ScreenSaver" -> "org.gnome.ScreenSaver" -> "SetActive" que funciona igualment i un altre, sota l'objecte "org.freedesktop.ScreenSaver", que no està implementat

aII) ¿A quina comanda *busctl* ... seria equivalent l'execució de l'apartat anterior? Prova-la per confirmar-ho

b) Fes doble clic sobre el mètode: "org.freedesktop.portal.Desktop" -> "/org/freedesktop/portal/desktop" -> "org.freedesktop.portal.Screenshot" -> "Screenshot" del canal de sessió, escriu una cadena buida com a valor de cadascun dels dos paràmetres que necessita (separats per una coma), així: "", "" i clica sobre el botó "Executa". ¿Què passa?

NOTA: El primer paràmetre del mètode utilitzat en aquest apartat serveix per indicar l'identificador de la finestra que es vol capturar: caldria, llavors, esbrinar-lo d'alguna manera però si es vol capturar tota la pantalla no cal indicar res en aquest paràmetre (per això s'escriu la cadena buida). El segon paràmetre serveix per indicar diverses opcions a l'hora de fer la captura, però també el deixarem buit.
NOTA: Antigament es podien fer captures de pantalla amb el mètode "org.gnome.Shell.Screenshot" -> "/org/gnome/Shell/Screenshot" -> "org.gnome.Shell.Screenshot" -> "Screenshot", el qual té tres paràmetres: el primer indica si es vol que aparegui el cursor del ratolí (o no) a la captura; el segon indica si es farà "flash" (o no) a la captura i el tercer indica la ruta del fitxer imatge a generar (si s'indica una ruta relativa, aquesta ho serà respecte la carpeta personal de l'usuari actual). No obstant, a les darreres versions de Gnome aquest mètode no és permés per una qüestió de seguretat

bII) Sabent el que ja saps de fer l'apartat anterior, si executes la següent comanda ¿què creus que passarà?:
busctl --user call org.freedesktop.portal.Desktop /org/freedesktop/portal/desktop org.freedesktop.portal.Screenshot Screenshot sa{sv} "" 0 ¿Què significa aquest darrer "0" indicat a la comanda anterior?

Fer crides de mètodes remots sobre D-Bus és només una part de les capacitats de D-Bus: D-Bus també admet un mètode de comunicació de difusió (és a dir, "broadcast"), igualment asíncron. Aquest mecanisme s'anomena "senyal" (en la terminologia D-Bus) i és útil quan cal notificar a molts receptors un canvi d'estat que els pugui afectar. Alguns exemples en què els senyals podrien ser útils són la notificació de molts receptors si s'està apagant el sistema, s'ha perdut la connectivitat de xarxa, etc. D'aquesta manera, els receptors no necessiten interrogar l'estat d'una determinada propietat o cridar a un determinat mètode contínuament.

Cada senyal és un membre de la interfície que té el seu propi nom i "arguments". En el cas del senyal, la llista d'arguments és només una llista d'informació que es transmet al llarg del senyal (i que no s'ha de confondre amb els paràmetres de crides a mètodes, tot i que tots dos es lliuren de la mateixa manera).

Els senyals no "retornen", és a dir, quan s'envia un senyal D-Bus, no es rebrà cap resposta ni s'esperarà. Si l'emissor de senyal vol assegurar-se que el senyal s'ha enviat, cal construir mecanismes addicionals per a això (D-Bus no els inclou directament). De fet, un senyal D-Bus és molt similar a la majoria de protocols de xarxa basats en datagrames, per exemple UDP.

Systemd ofereix la comanda *busctl { --system | --user } emit [--destination nom.Servei] /nom/objecte nom.Interfície nomSenyal signatura paràmetres* per emetre senyals (amb els seus corresponents paràmetres) al canal indiscriminadament (tot i que, opcionalment es pot indicar un determinat servei per enviar-li només a ell la senyal).

10.-a) Després de llegir els paràgrafs blaus anteriors, obre un terminal a la màquina real i executa la comanda necessària per emetre la senyal anomenada "ActiveChanged" (la qual té un únic paràmetre de tipus booleà) pertanyent a la interfície "org.gnome.ScreenSaver" pertanyent a l'objecte "/org/gnome/ScreenSaver", presents al canal D-Bus de sessió. ¿Què estàs fent en realitat en executar aquesta comanda?

> NOTA: Existeix un altre servei anomenat "org.freedesktop.ScreenSaver" que també proporciona una senyal anomenada "ActiveChanged" (pertanyent a la interfície "org.freedesktop.ScreenSaver" de l'objecte "/org/freedesktop/ScreenSaver") però no funciona

b) Obre un altre terminal a la màquina real i executa-hi la comanda *busctl --user monitor | grep -B 1 -A 5 "org.gnome.ScreenSaver"* . Torna a executar la comanda que vas provar a l'apartat anterior i observa què es veu ara en aquest segon terminal que acabes d'obrir. ¿Què és? ¿Què passa si tornes a executar la mateixa comanda *busctl monitor* però treient la comanda *grep*?

bII) Es pot filtrar una mica la sortida de la comanda *busctl monitor* sense haver de fer servir *grep* gràcies al seu paràmetre *--match* .Sabent això, repeteix l'apartat anterior però ara executant la comanda *busctl --user monitor --match interface=org.gnome.ScreenSaver* ¿Què veus ara?

> NOTA: Per més informació dels valors que es poden escriure en el paràmetre *--match*, veieu https://dbus.freedesktop.org/doc/dbus-specification.html#message-bus-routing-match-rules (i també el següent apartat)

bIII) Si executes el següent script i tot seguit actives el salvapantalles del Gnome (per anar ràpid, això pots fer-ho simplement pulsant SUPER+L) i el tornes a desactivar, ¿què veus a pantalla? (raona el per què observant el codi de l'script i també la sortida que genera la comanda *busctl* indicada)

```
#!/bin/bash
busctl --user monitor --match "interface='org.gnome.ScreenSaver',type='signal'" |  while read linia; do
    case "$linia" in
      *"BOOLEAN true"*) echo "Salvapantalles activat"
    ;;
      *"BOOLEAN false"*) echo "Salvapantalles desactivat"
    ;;
    esac
done
```

> NOTA: En lloc d'escriure un missatge a pantalla, quelcom més útil seria, per exemple, habilitar/deshabilitar els dispositius USB mentre el salvapantalles està activat (mitjançant l'ús adient de la comanda *usbguard*, per exemple), entre altres idees

c) Tanca totes els terminals i ara obre un de nouc a la màquina real i executar-hi la següent comanda: *busctl --user capture --match interface=org.gnome.ScreenSaver > fitxer.pcap* Mentre aquesta comanda es queda funcionant, activa el salvapantalles del Gnome (recorda, amb SUPER+L). Finalment, atura la comanda *busctl* ... anterior i obre el fitxer generat ("fitxer.pcap") amb el programa Wireshark (que deus tenir instal·lat ja al sistema de l'aula). ¿Què se't mostra?