Building a Modern Multi-User Desktop

William Jon McCann <mccann@jhu.edu>
The Johns Hopkins University

GUADEC
17 July 2007

What is a multi-user desktop?

What is a multi-user desktop?

- A system that can support a rich experience for multiple simultaneous user sessions
- A user session that is multi-user aware

Why is this useful?

Why is this useful?

- Over 80% of US households have a shared computer (according to a Microsoft report)
- Even in households/businesses with multiple computers some are more desired than others.
- Computers are idle most of the time.
- Computers are expensive. (cost, heat, space, noise)
- Guests don't want to see your porn collection.
- Why log out?

http://windowsxp.devx.com/presentations/devforawinxpmultiuserpc_110k/default.htm

What forms can this take?

What forms can this take?

- Fast User Switching – share from single location

What forms can this take?

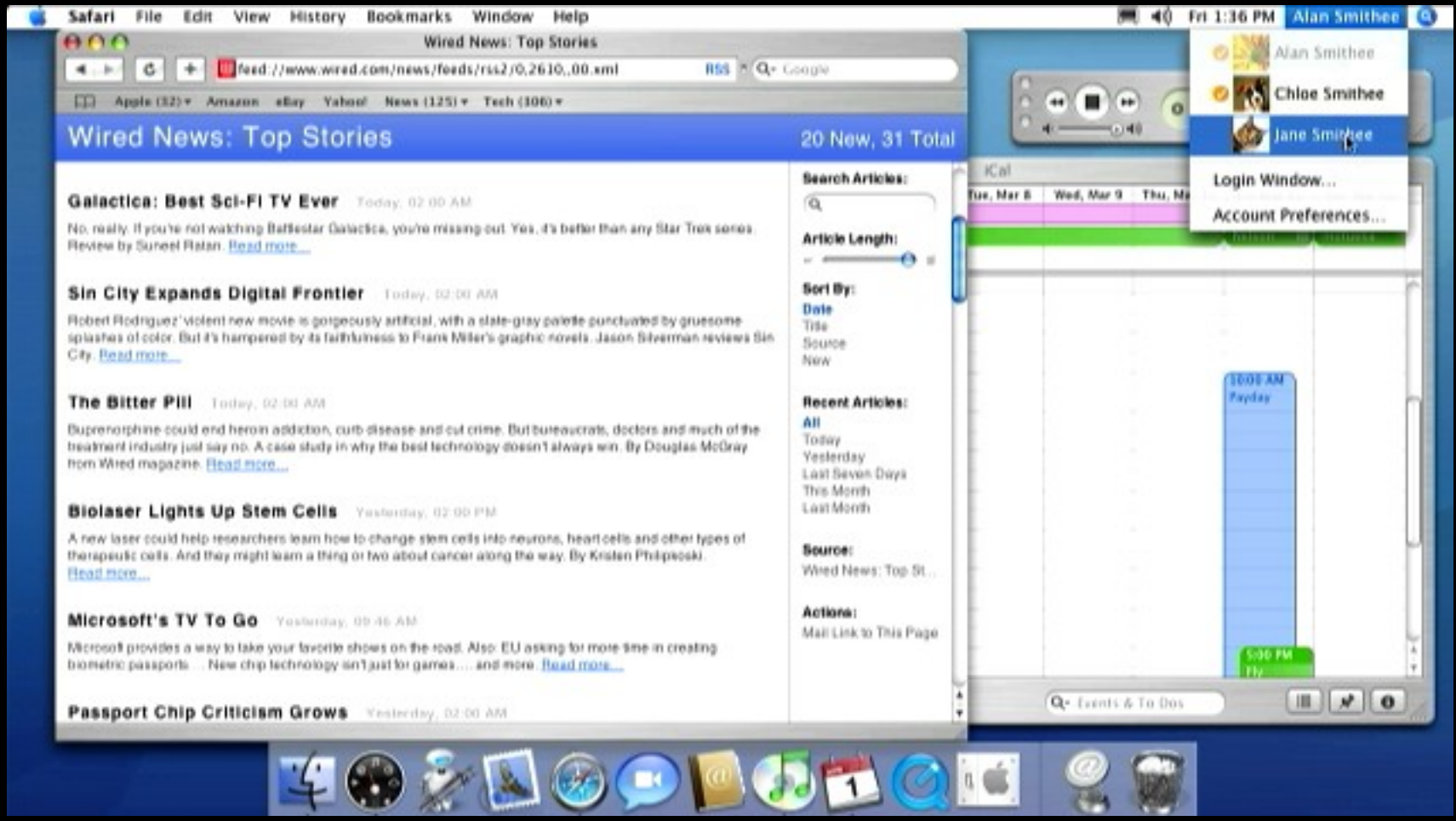
- Fast User Switching – share from single location
- Multi-Seat – share from different locations

What forms can this take?

- Fast User Switching – share from single location
- Multi-Seat – share from different locations
- Hot desking / Session migration - combination

What is Fast User Switching?

OS X Fast User Switching

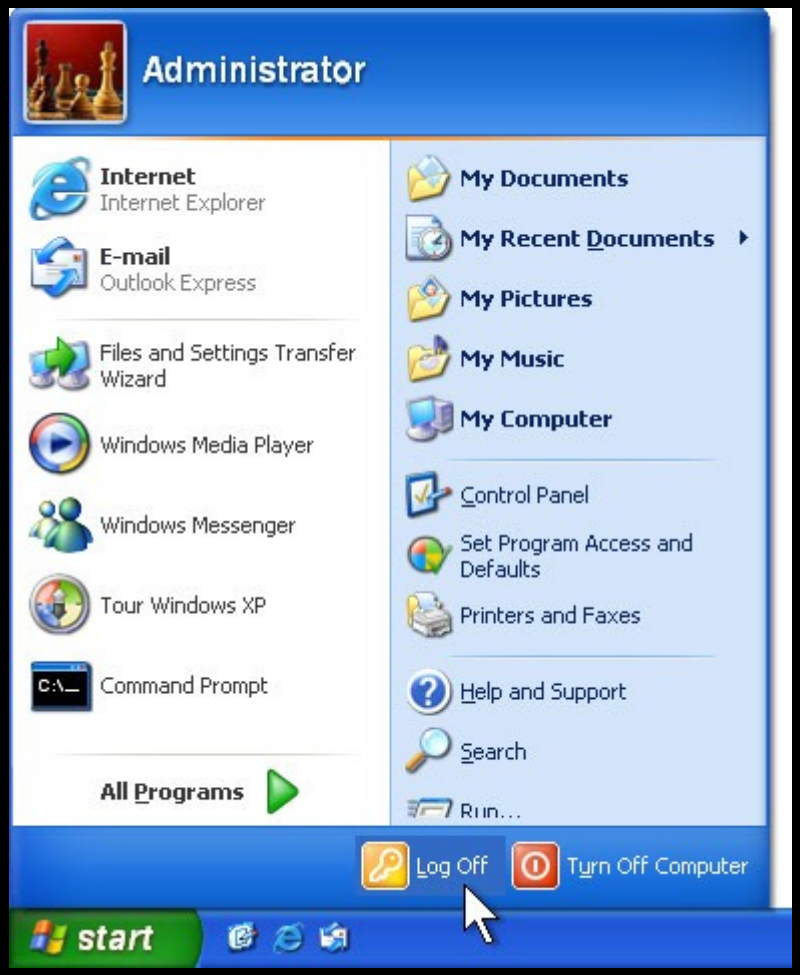


OS X Fast User Switching

OS X Fast User Switching

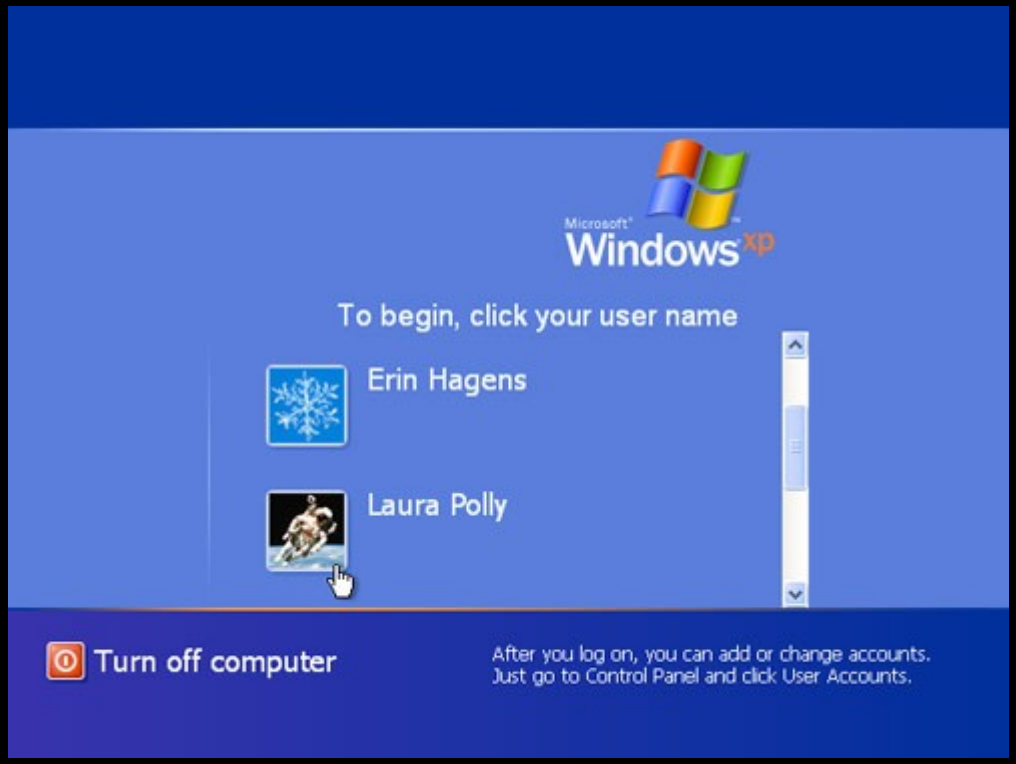
OS X Fast User Switching

Windows XP Fast User Switching



Windows XP Fast User Switching

Windows XP Fast User Switching

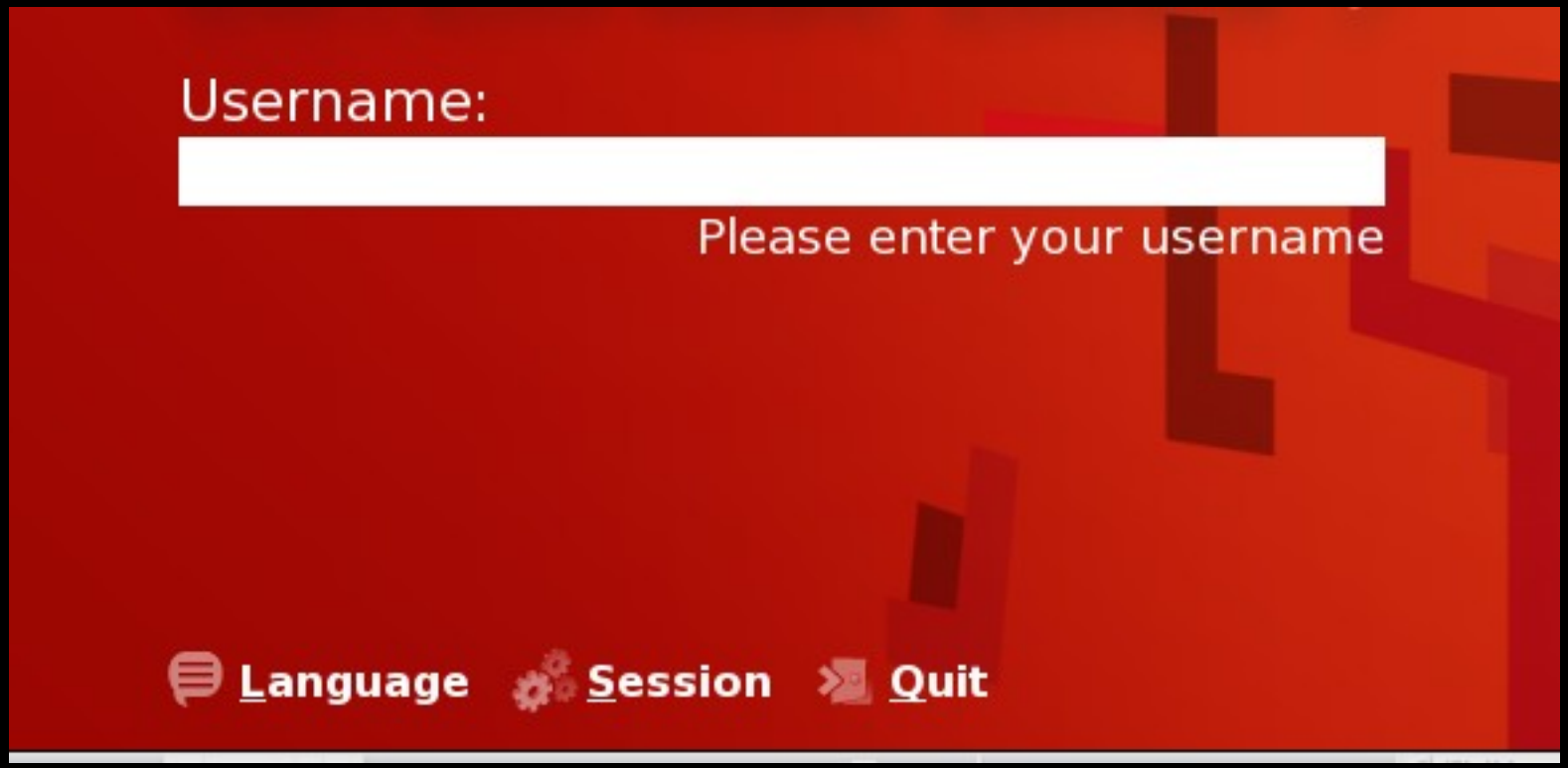


Can't I do that already?

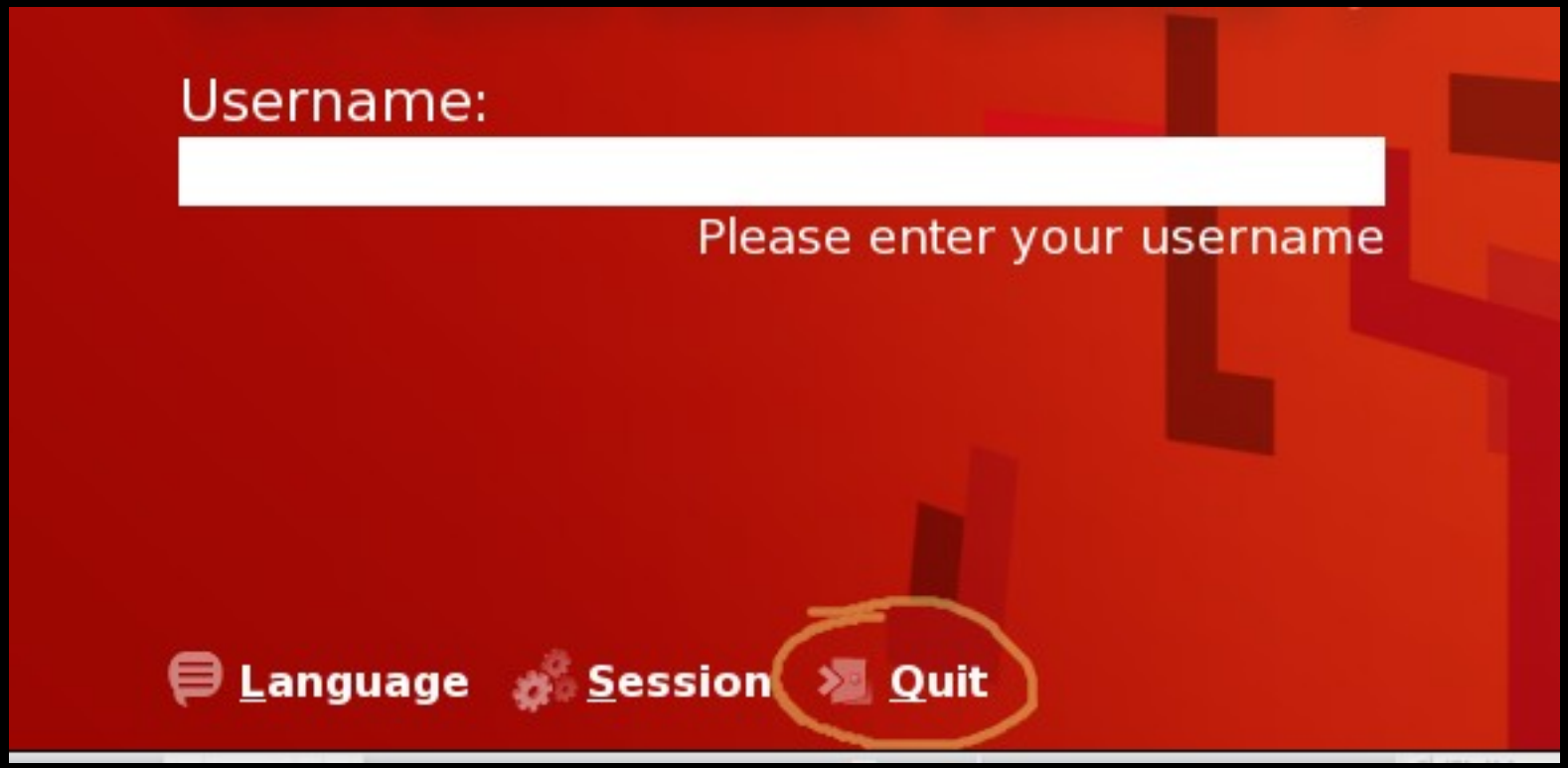
GNOME 2.16 Fast User Switching

GNOME 2.16 Fast User Switching

GNOME 2.16 Fast User Switching



GNOME 2.16 Fast User Switching



GNOME 2.16 Fast User Switching

What did we get wrong?

What did we get wrong?

- Selected a user in Switcher applet but Login Screen didn't use it.
- Screen goes all flashy-flashy between Switcher applet and Login Screen.
- Login Screen has a Quit button.
- Login Screen dies after a while.
- Login Screen either doesn't have a “face-browser” or has a not-so-good one.
- After authenticating we switch directly to... a locked screen.

What did we get wrong?

- After logging out we can end up on another locked screen or even an empty VT.
- Applications are still consuming resources in inactive session. (Videos, screensavers etc)
- Applications in inactive session may not relinquish devices. (Music players, webcams)
- Sessions may race to acquire new devices.

What did we get wrong?

- No power-management while at Login Screen.
- Potentially no wireless networking at Login Screen. (software updates, backups, remote login)
- No user status information at Login Screen.
- User switcher applet isn't seat aware.
- We continue to poll optical drives when no one is logged in.

What did we get wrong?

- And so on...

What did we get wrong?

- Survey: root passwords

What do we need?

Text Login Manager

- Open a new Session.
- Set properties for the Session.
- Maintain some state for life-cycle tracking.
- Close the Session at logout.

Graphical Login Manager

- Get a list of attached Seats.
- Know if the current seat supports session switching.
- List all sessions on the current Seat.
- Know which session is active for the current Seat.
- Notification of session active state changes.
- Notification when sessions are added or removed.
- Access to the metadata for any open Session.

System daemon

- Know if any user sessions are open.
- Know if the system is currently being used.

Hardware Abstraction Layer

- Determine what hardware is associated with a Seat.
- Determine if a Session is active or inactive on a particular Seat.
- Know when the session active state changes.
- Determine what Session a process belongs to.

Fast User Switching Agent

- Determine which session it is running in.
- Determine which Seat it is running on.
- Know if the current seat supports session switching.
- Get a list of all sessions on the current Seat.
- Find which session is active for the current Seat.
- Know when the session active state changes.
- Access to the metadata for any open Session.
- Know when sessions are added or removed.

What is a Session?

What is a Session?

- The collection of processes that are direct descendants of a single, authenticated, interactive login process for a real user?

Desperately Seeking a Session

- POSIX process session
- OS security context
- PAM session
- Entry in the accounting database
- D-Bus session
- Desktop session management
- All connections to a Xserver
- XDMCP Session

POSIX process session

- The problem here is that there are various ways to create new sessions (process groups) for a single user login. For example, when a program (gnome-power-manager) is daemonized it detaches from the controlling terminal and is no longer part of the same session.
- Not the behavior that we want

OS security context

- Didn't seem like it was possible to uniquely identify groups of processes.

PAM sessions

- Excellent support for defining the "who" of the session but little support for the "where" and "what".
- Shouldn't trust passed in PAM_TTY value.
- PAM_TTY value isn't always a tty but sometimes a $DISPLAY.
- Sometimes PAM_TTY is just bogus (eg. sshd uses "ssh")
- Can't reliably get a pid for session leader since PAM module isn't always run by that process but by a helper/proxy or parent.
- Prior art – pam_console/foreground

Entry in the accounting database (utmp)

- Information is not authoritative.
- Poor standardization of structure.
- Few rules for content / usage.
- Designed for TTY (no X11 support)
- Must monitor file, re-read, and compare with last known state in order to get change notifications (and still aren't atomic)
- Field length limitations.

D-Bus session

- Historically and currently, non-graphical "sessions" have not started a D-Bus session – but could.
- Not all processes are necessarily connected to a session bus - not a problem.
- The grouping characteristic isn't so much the connection to the bus as the knowledge of the bus address.

Desktop session management

- This is too high level and oriented toward stateful graphical applications. We need something that works as well for daemon processes and non-graphical sessions.

All connections to a Xserver

- Obviously precludes non-graphical sessions.
- Maintaining the X connection may not be desirable.
- Sessions probably shouldn't be defined server-side.

XDMCP Session

- Too specific obviously.
- But a case that must be handled.

So... what is a session?

- A session is a collection of all processes that share knowledge of a secret. In the typical case, these processes all originate from a single common ancestor.

Session Implementation Details

- For now, this secret should be stored in the process environment by the session leader under the name XDG_SESSION_COOKIE.
- When we are able to take advantage of a mechanism in the underlying system to store session registration information - we will.

Side Effects

- A possible advantage - it is quite easy for a process to opt-out of a Session by simply unsetting the XDG_SESSION_COOKIE variable. Well... probably have to fork/exec too in practice.
- A possible disadvantage - not being able to strictly limit visibility of the secret to a particular process ancestry. So, it is not possible to enforce session boundaries other than on a per-user basis. For example, we don't yet have a way to prevent a process from moving between sessions owned by the same user.

What is a Seat?

- A seat is a collection of sessions and a set of hardware (usually at least an output and input device).

What is multi-seat?

But...

- Aren't new, small and mobile form devices eliminating the need for multi-user support? Or... what part of One Laptop Per Child don't you understand?

But...

- Aren't new, small and mobile form devices eliminating the need for multi-user support? Or... what part of One Laptop Per Child don't you understand?
- If personalization of a potentially shared resource is helpful then it is worth considering multi-user support / profiles. (handwriting/voice recognition etc)

But...

- Aren't new, small and mobile form devices eliminating the need for multi-user support? Or... what part of One Laptop Per Child don't you understand?
- If personalization of a potentially shared resource is helpful then it is worth considering multi-user support / profiles. (handwriting/voice recognition etc)
- Ubiquity of mobile devices may push PC into more server-like and appliance roles – more integrated with a location. (Media/Entertainment center, Home server, Smart home)

What do we need to make it work better?

What do we need to make it work better?

- ConsoleKit
- HAL
- PolicyKit
- DisplayManager
- Kernel mode drivers
- Session Agents
- Well-behaved applications

ConsoleKit

- http://gitweb.freedesktop.org/?p=ConsoleKit.git
- http://people.freedesktop.org/~mccann/doc/ConsoleKit/ConsoleKit.html
- http://lists.freedesktop.org/archives/hal/2007-January/006996.html
- First prototype: 26 Sept 2006
- Initial check-in to git: 25 Oct 2006
- Announced: 11 Jan 2007

ConsoleKit

- ConsoleKit is a framework for defining and tracking user login sessions and seats.
- It can be used as a replacement for utmp.
- Who is logged in, from where, for how long, are they active, inactive, idle etc.

Session Object

- Properties
  - ID
  - Seat ID
  - Session type
  - Unix user
  - X11 display
  - X11 display device
  - Remote hostname
  - Active state
  - Local state
  - Creation Time
  - Idle hint state
  - Idle hint since time
- Signals
  - Active changed
  - Idle hint changed
  - Lock
  - Unlock

Seat Object

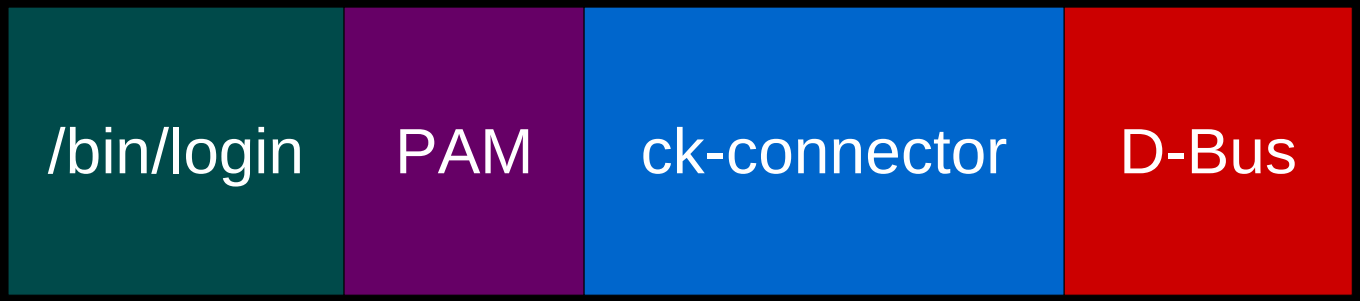
- Methods
  - GetId
  - GetSessions
  - GetActiveSessions
  - CanActivateSessions
  - ActivateSession
- Signals
  - ActiveSessionChanged
  - SessionAdded
  - SessionRemoved

Manager Object

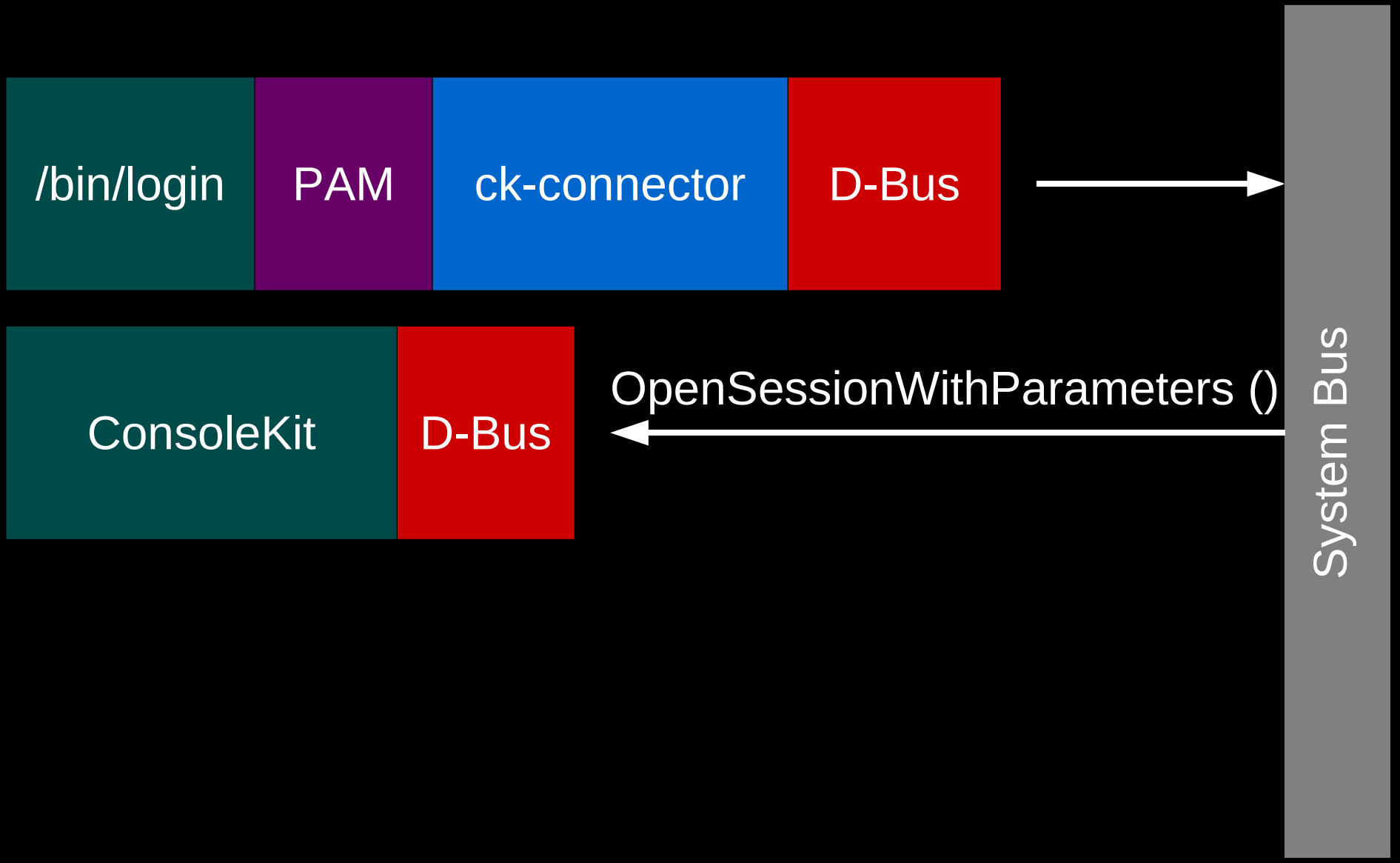
- Methods
  - OpenSession
  - OpenSessionWithParameters
  - CloseSession
  - GetSeats
  - GetSessionForCookie
  - GetSessionForUnixProcess
  - GetCurrentSession
  - GetSessionsForUnixUser
  - GetSystemIdleHint
  - GetSystemIdleHintSince
- Signals
  - SeatAdded
  - SeatRemoved
  - SystemIdleHintChanged

How does it work?

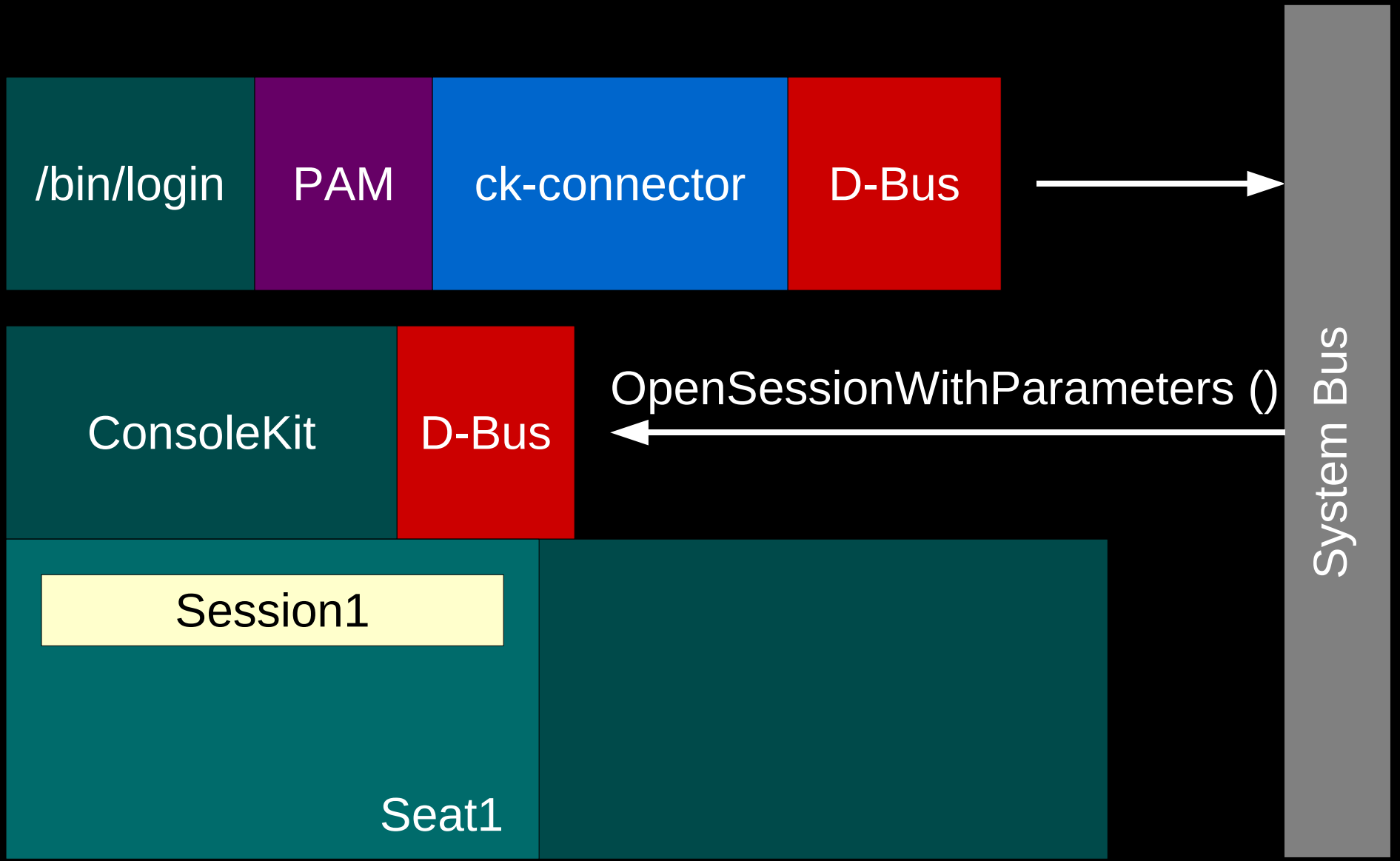
Text Login



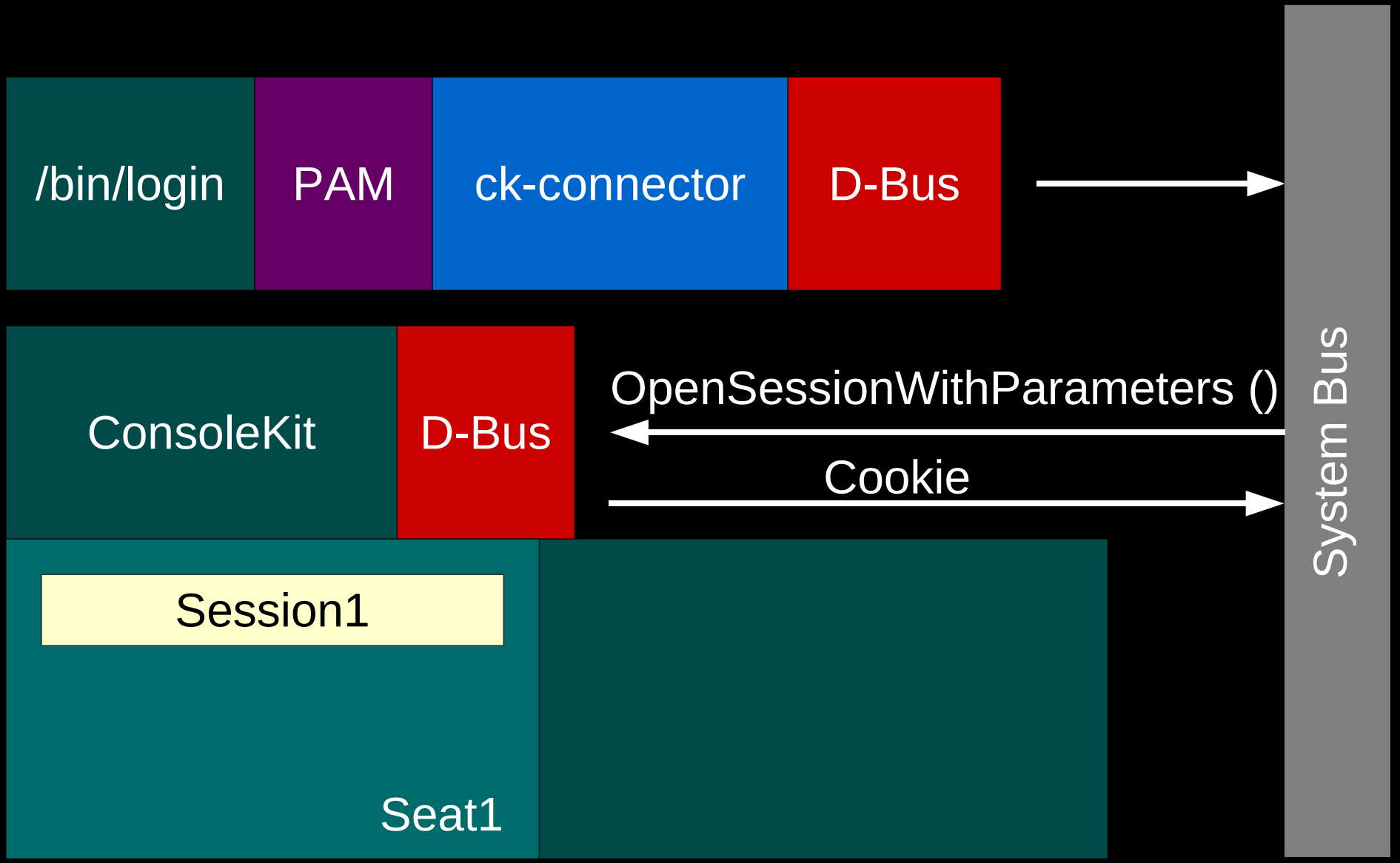
Text Login


/bin/login
PAM
ck-connector
D-Bus
System Bus
ConsoleKit
D-Bus
OpenSessionWithParameters ()

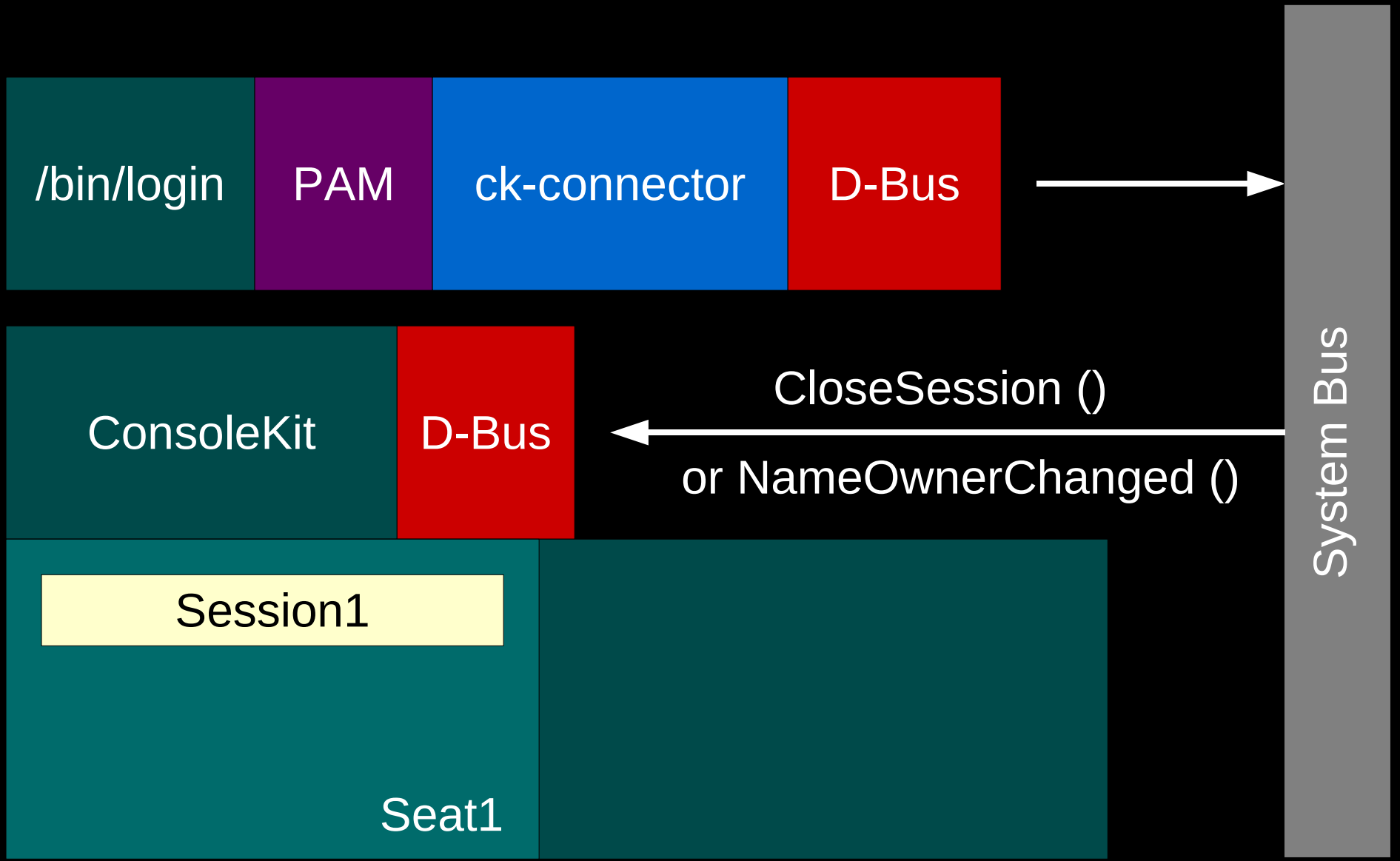
Text Login


/bin/login
PAM
ck-connector
D-Bus
System Bus
ConsoleKit
D-Bus
OpenSessionWithParameters ()
Session1
Seat1

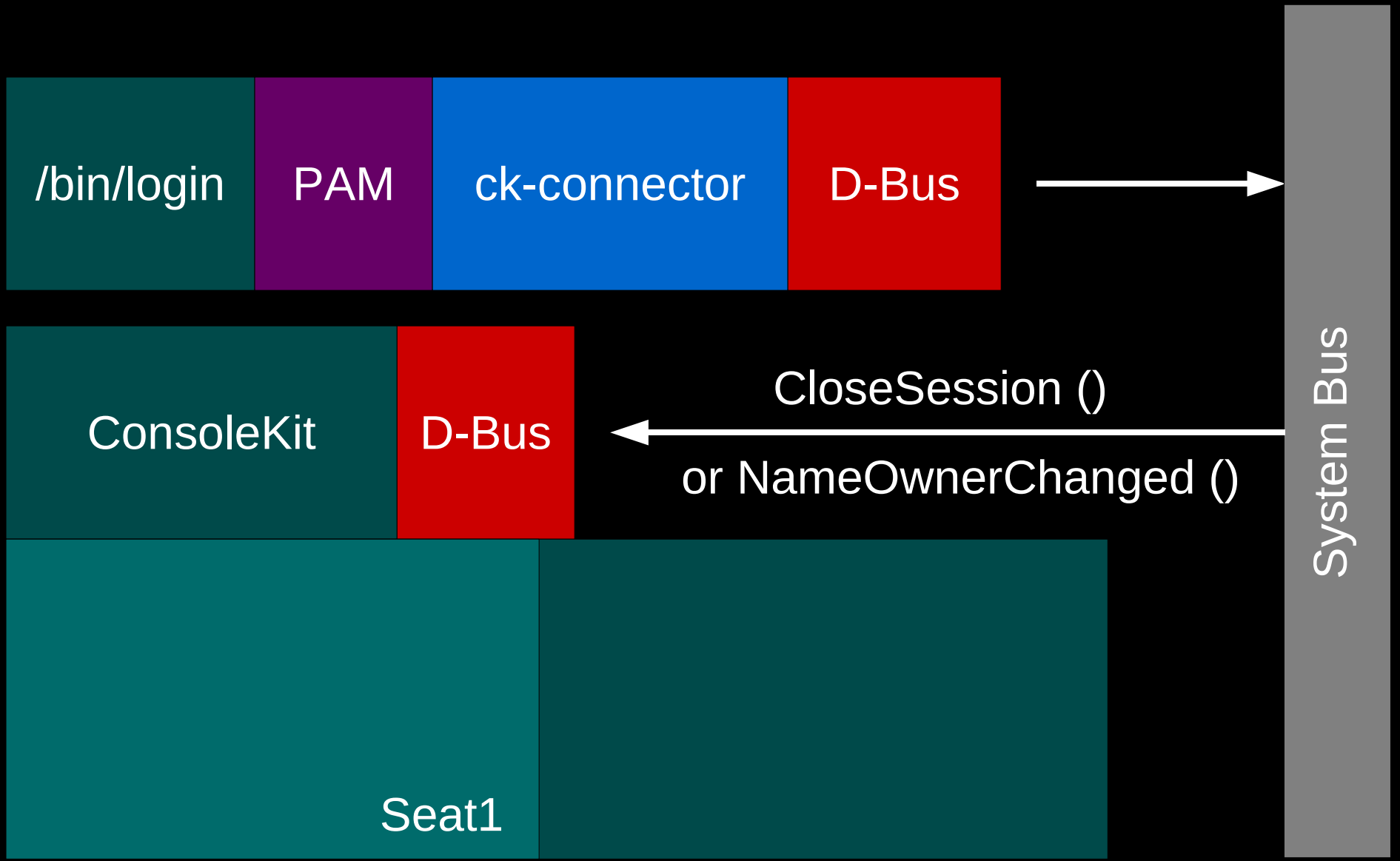
Text Login


/bin/login
PAM
ck-connector
D-Bus
System Bus
ConsoleKit
D-Bus
OpenSessionWithParameters ()
Cookie
Session1
Seat1

Text Login

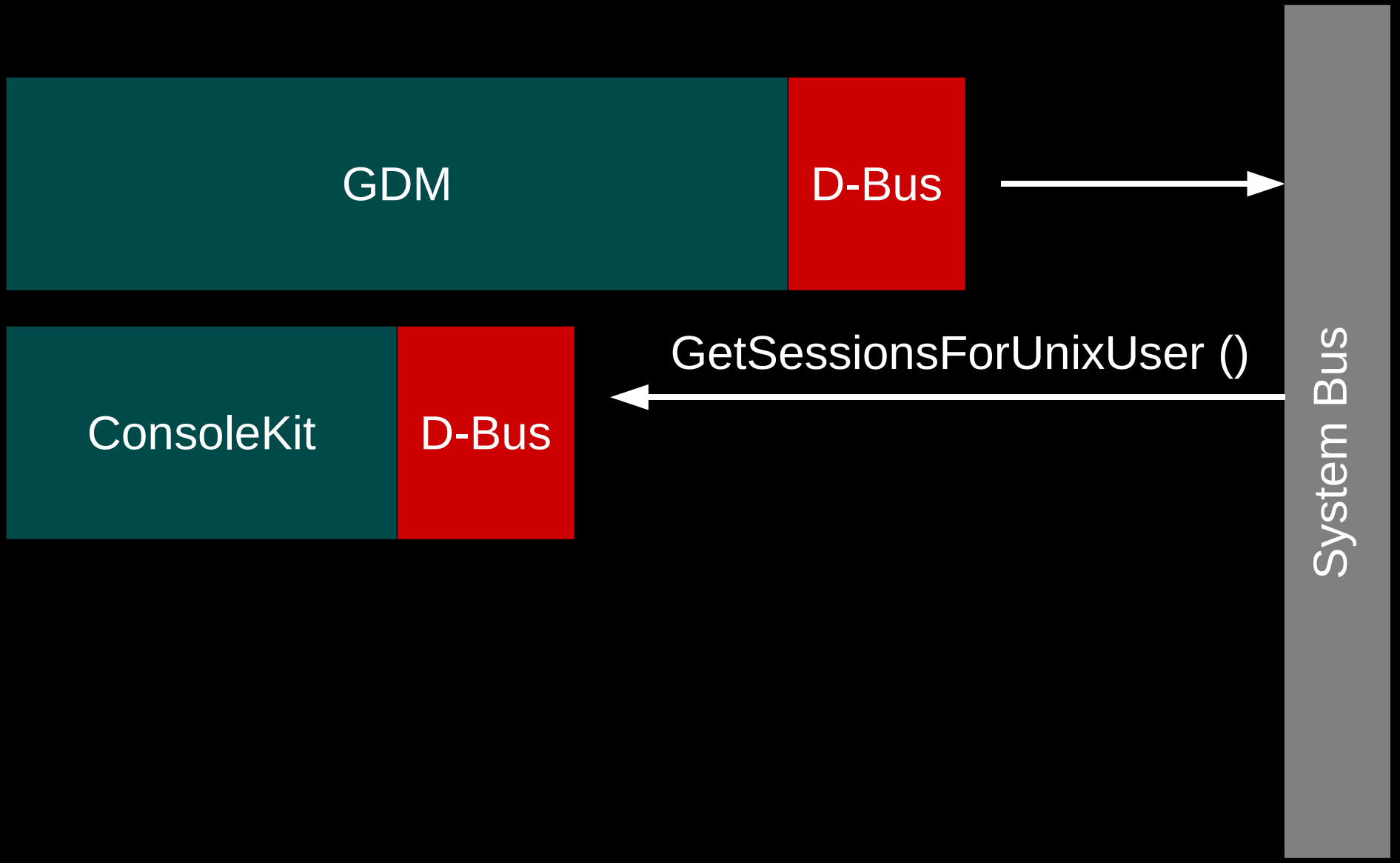

/bin/login
PAM
ck-connector
D-Bus
System Bus
ConsoleKit
D-Bus
CloseSession ()
or NameOwnerChanged ()
Session1
Seat1

Text Login

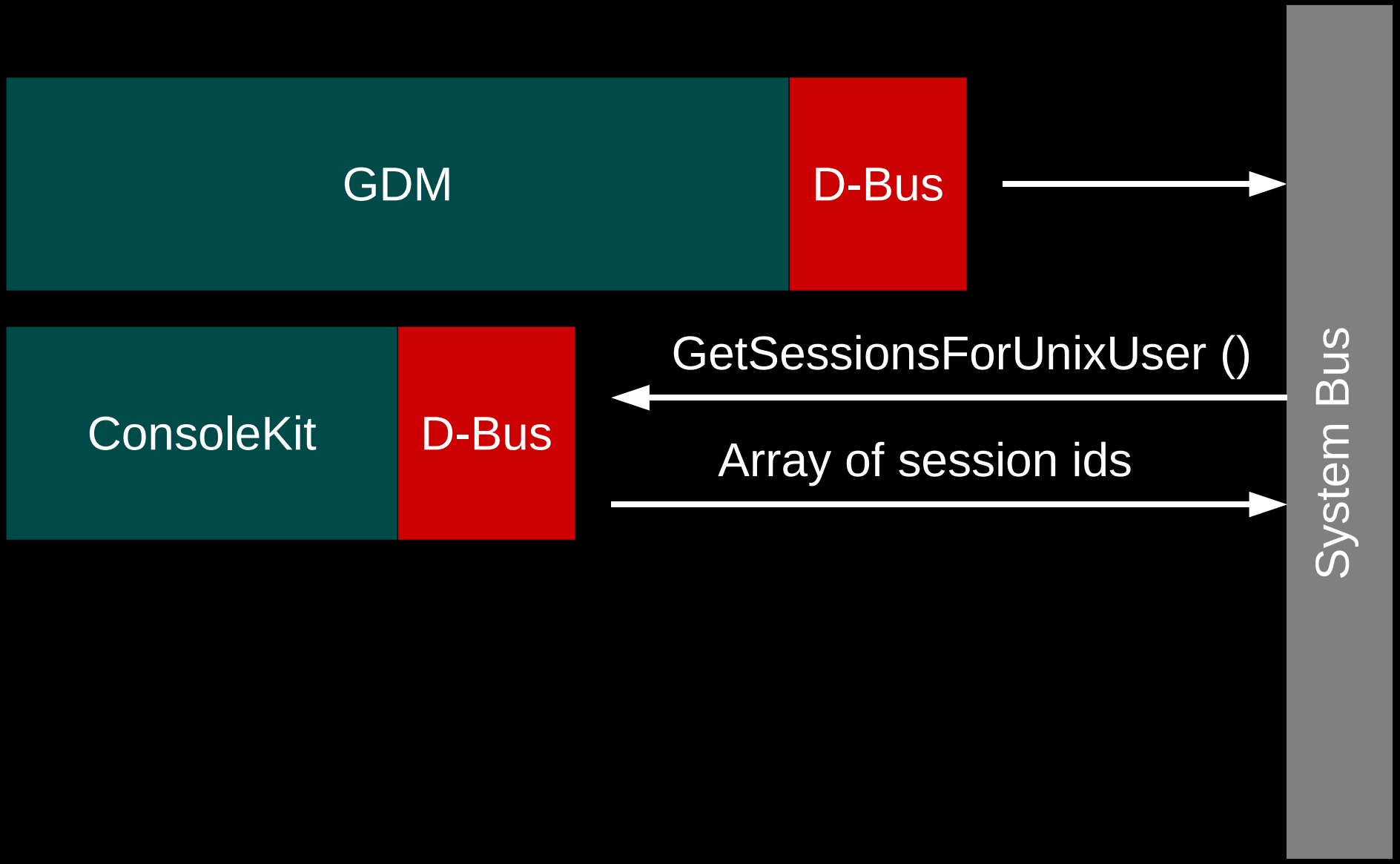

/bin/login
PAM
ck-connector
D-Bus
System Bus
ConsoleKit
D-Bus
CloseSession ()
or NameOwnerChanged ()
Seat1

Graphical Login

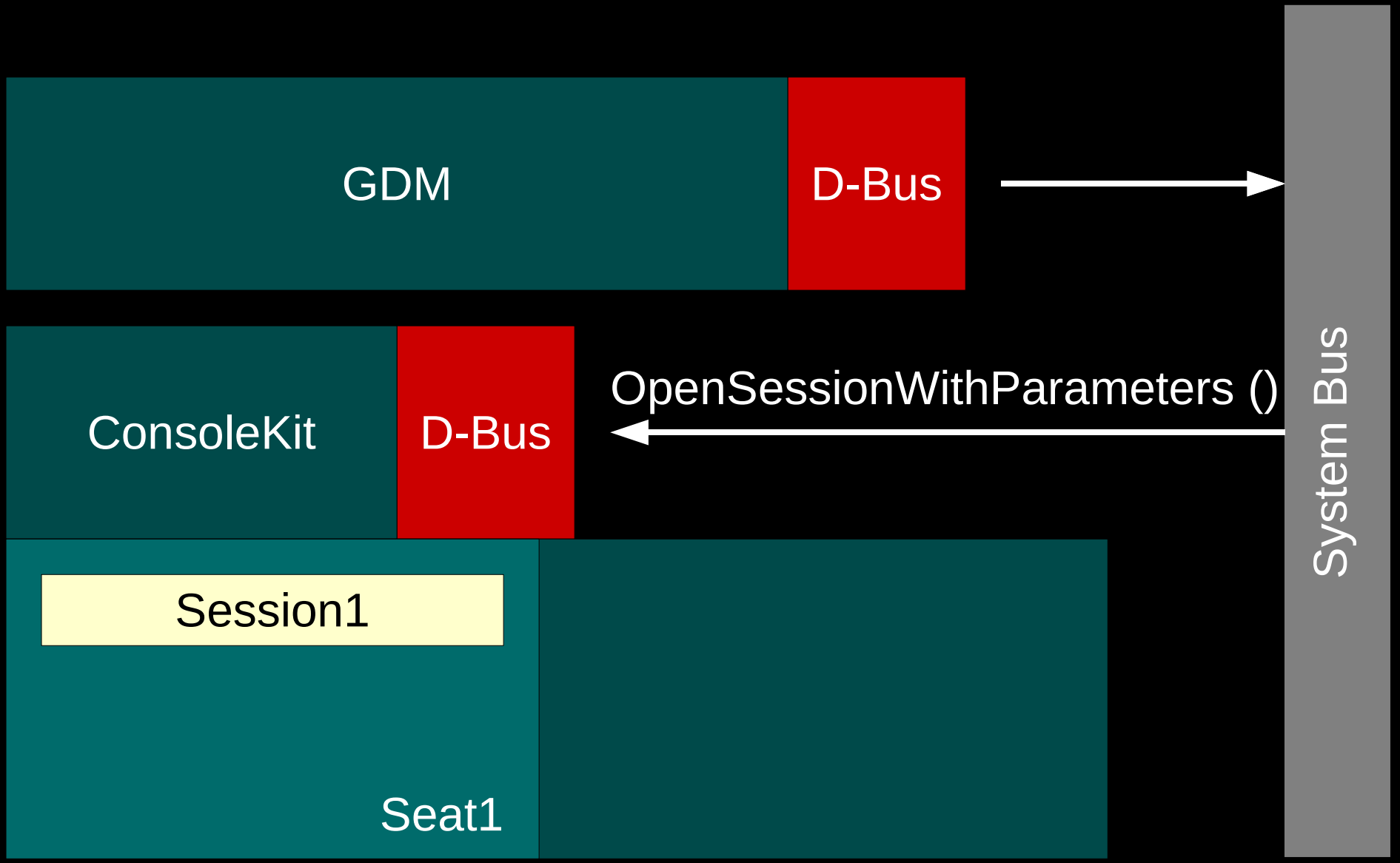
Graphical Login



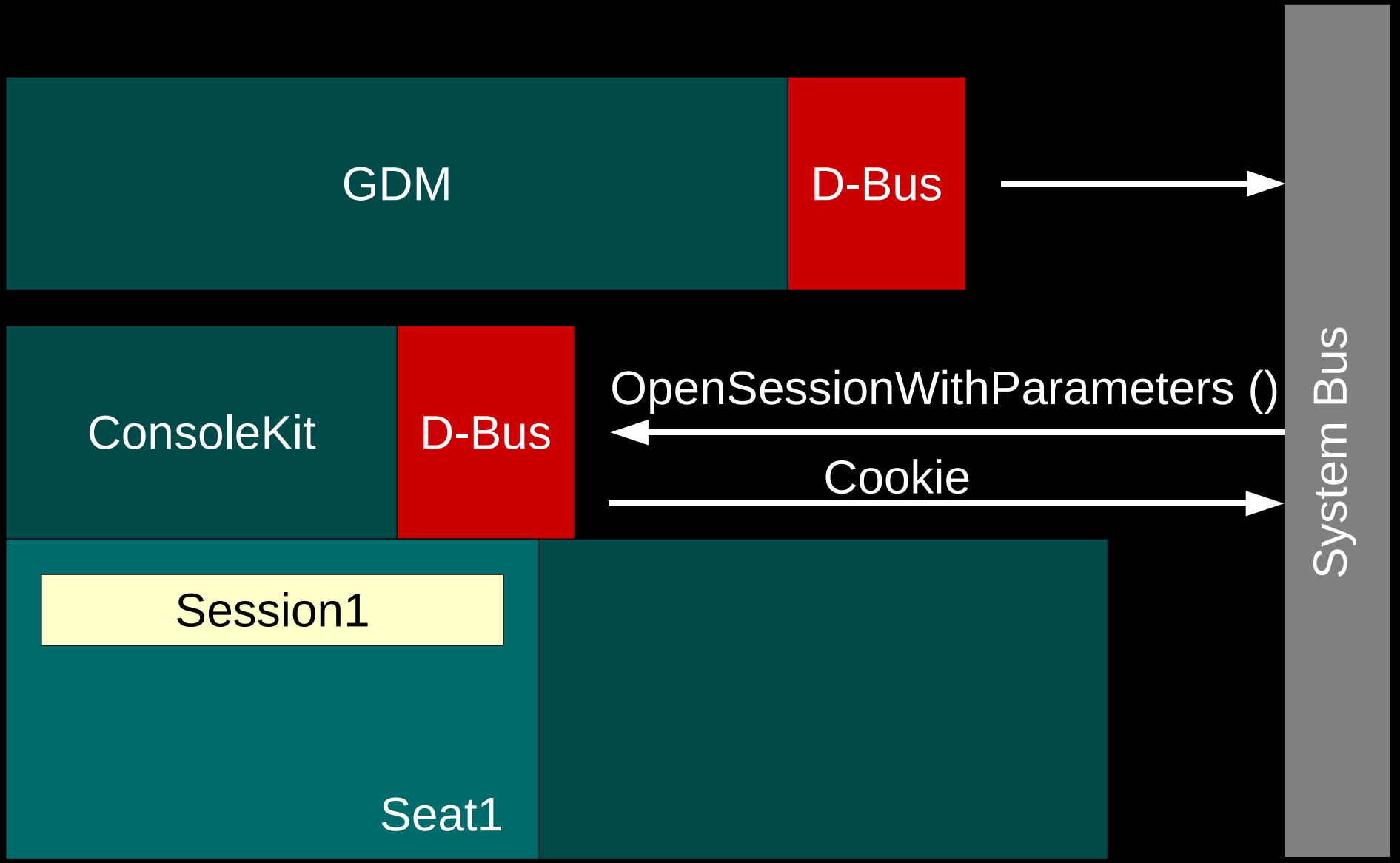
Graphical Login



Graphical Login

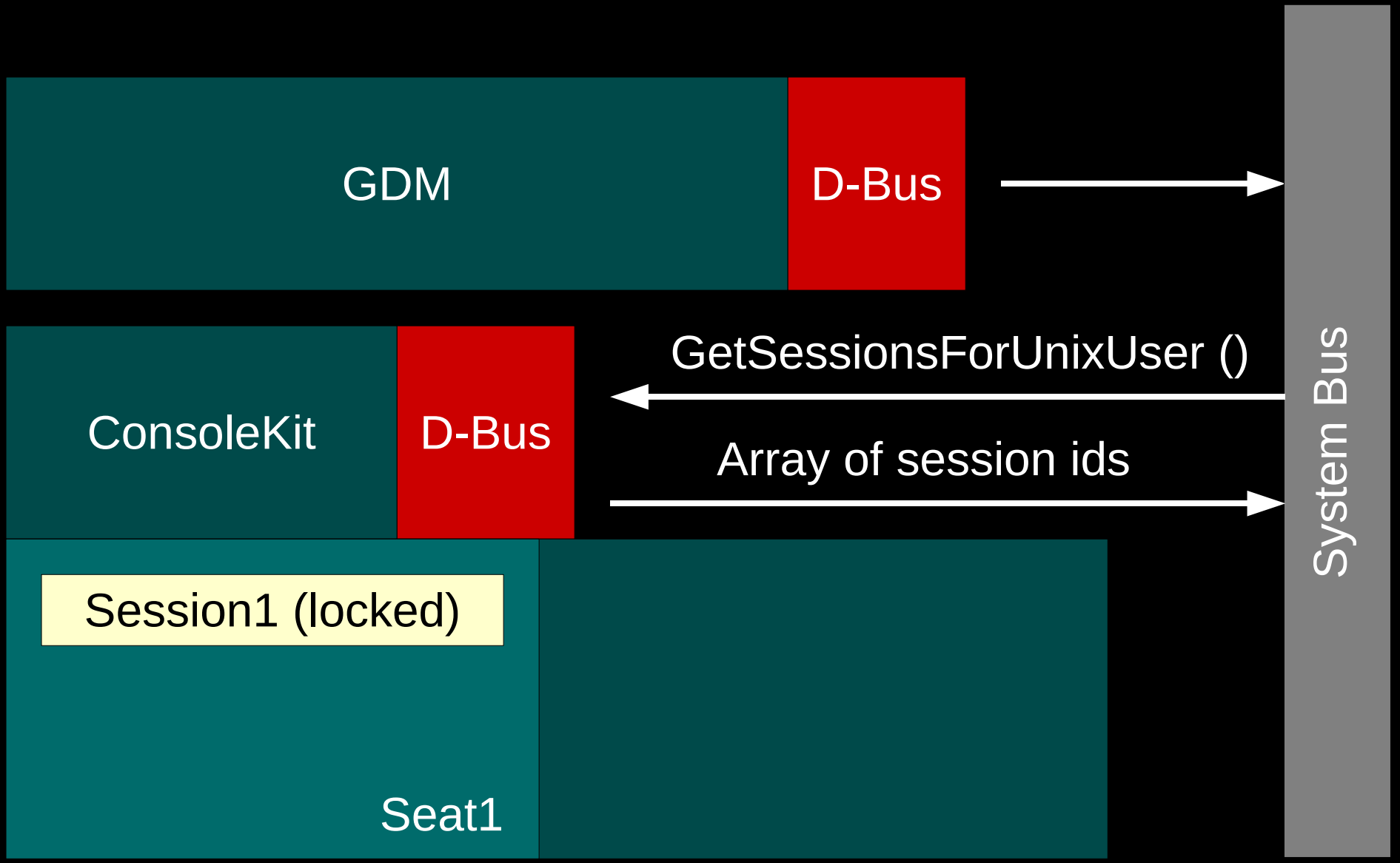


Graphical Login

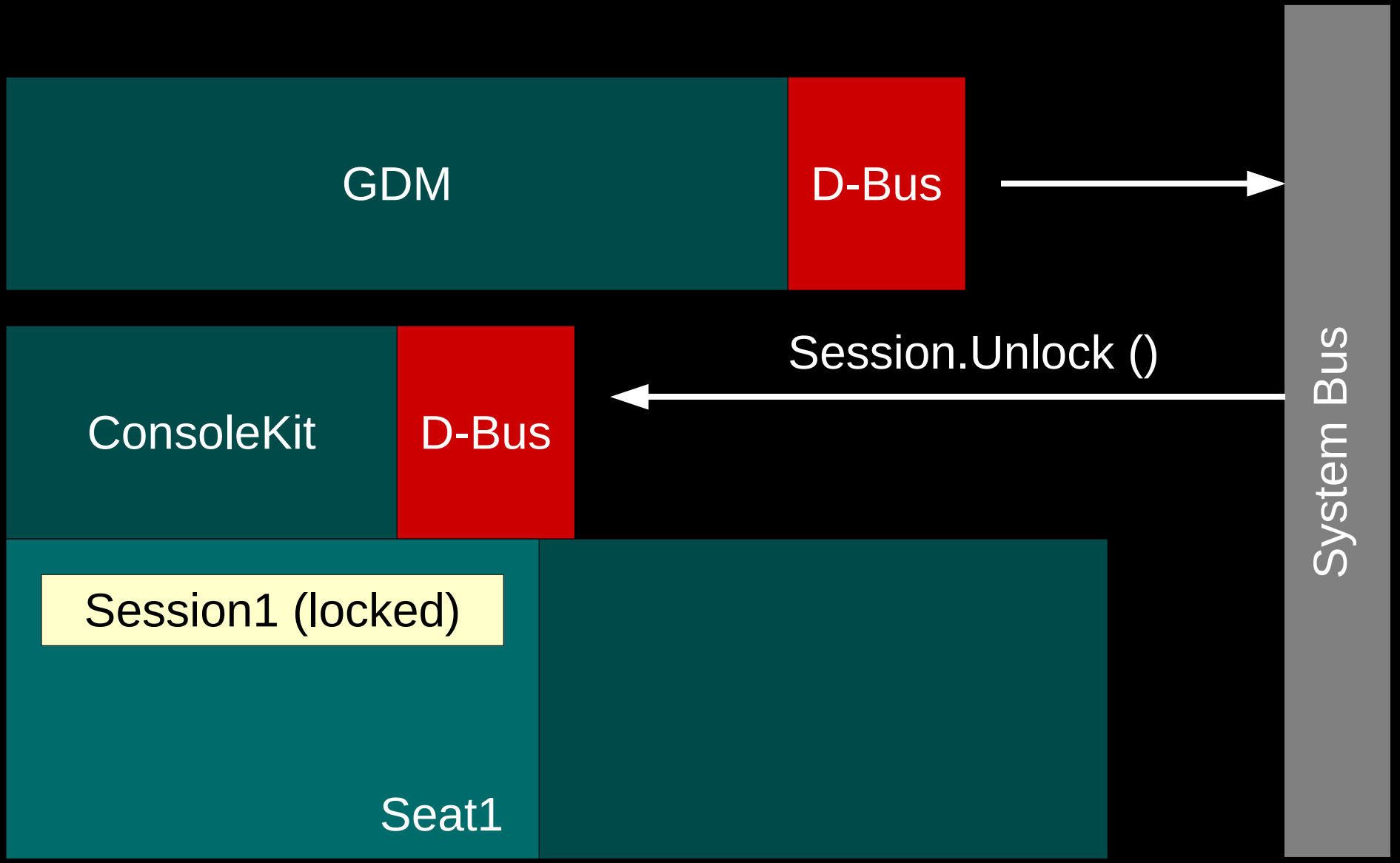


User Switching

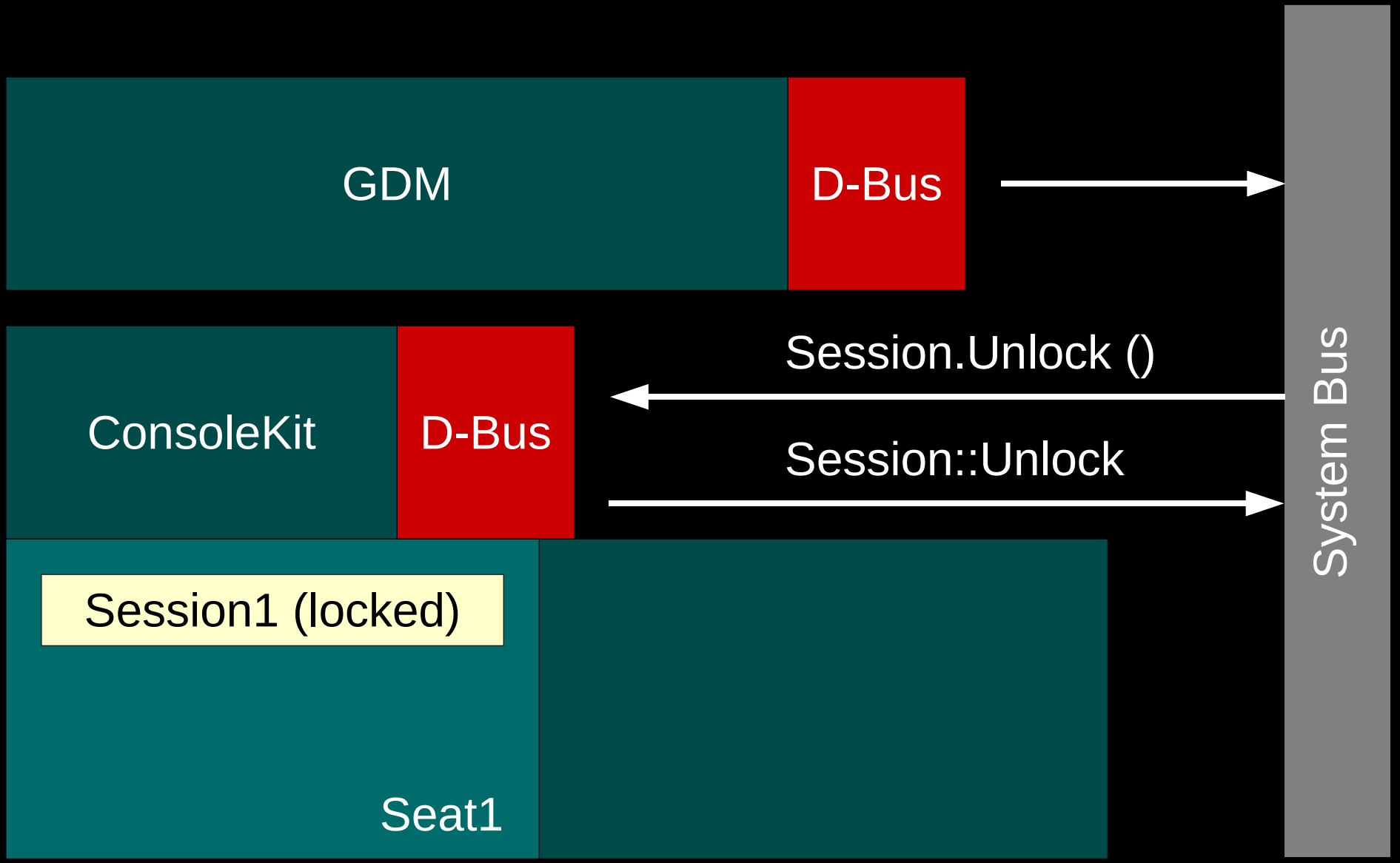
User Switching



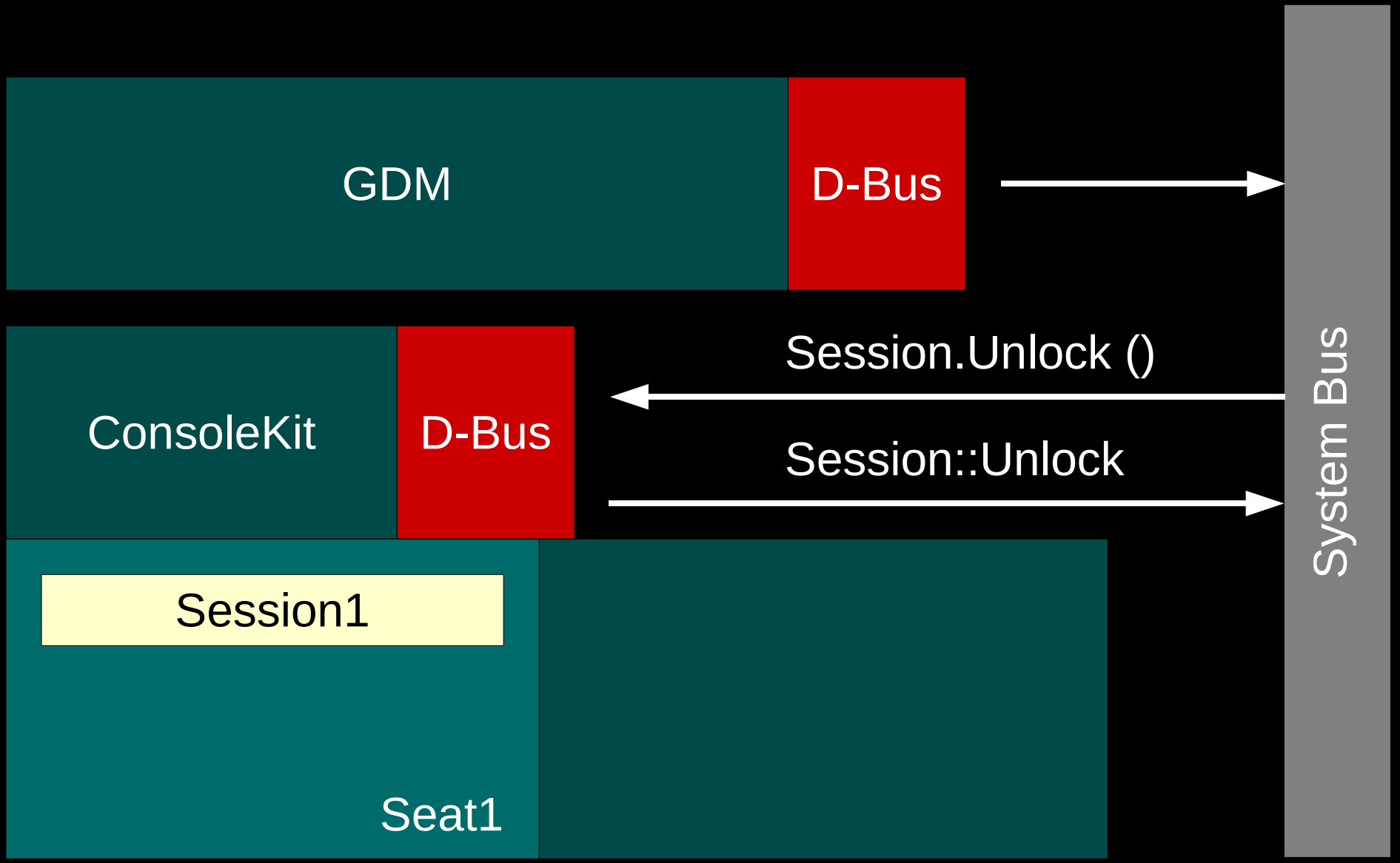
User Switching



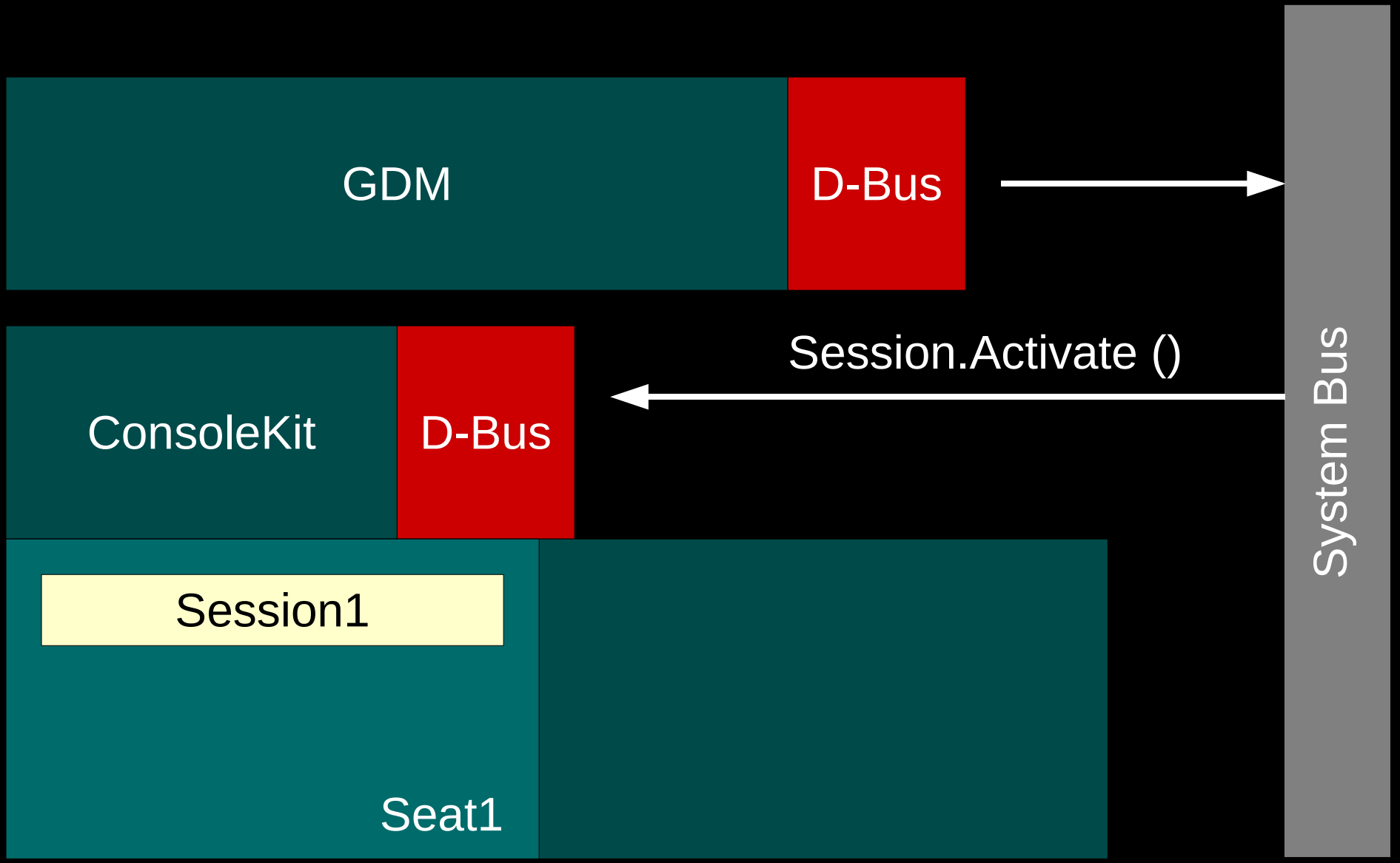
User Switching



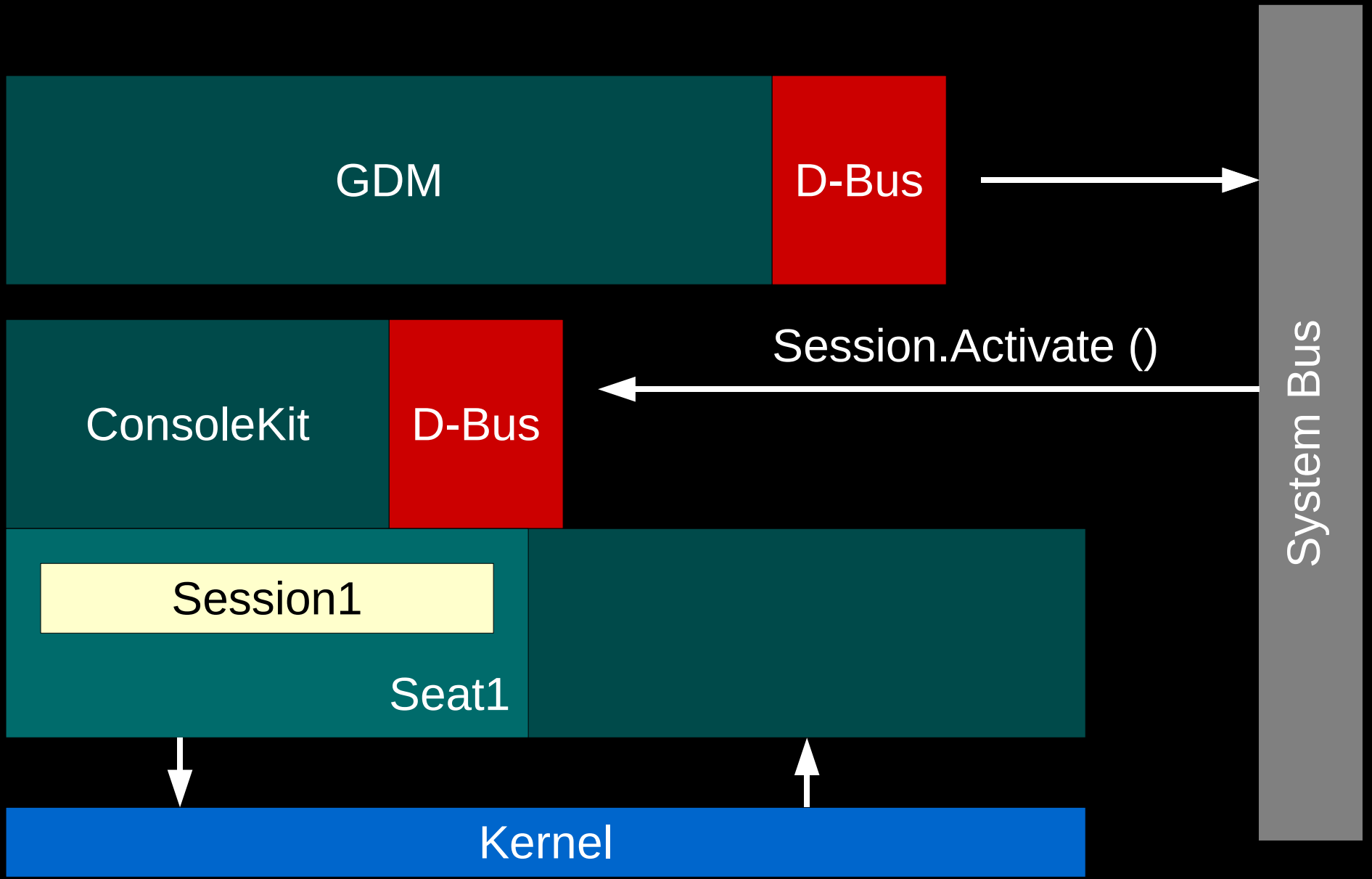
User Switching



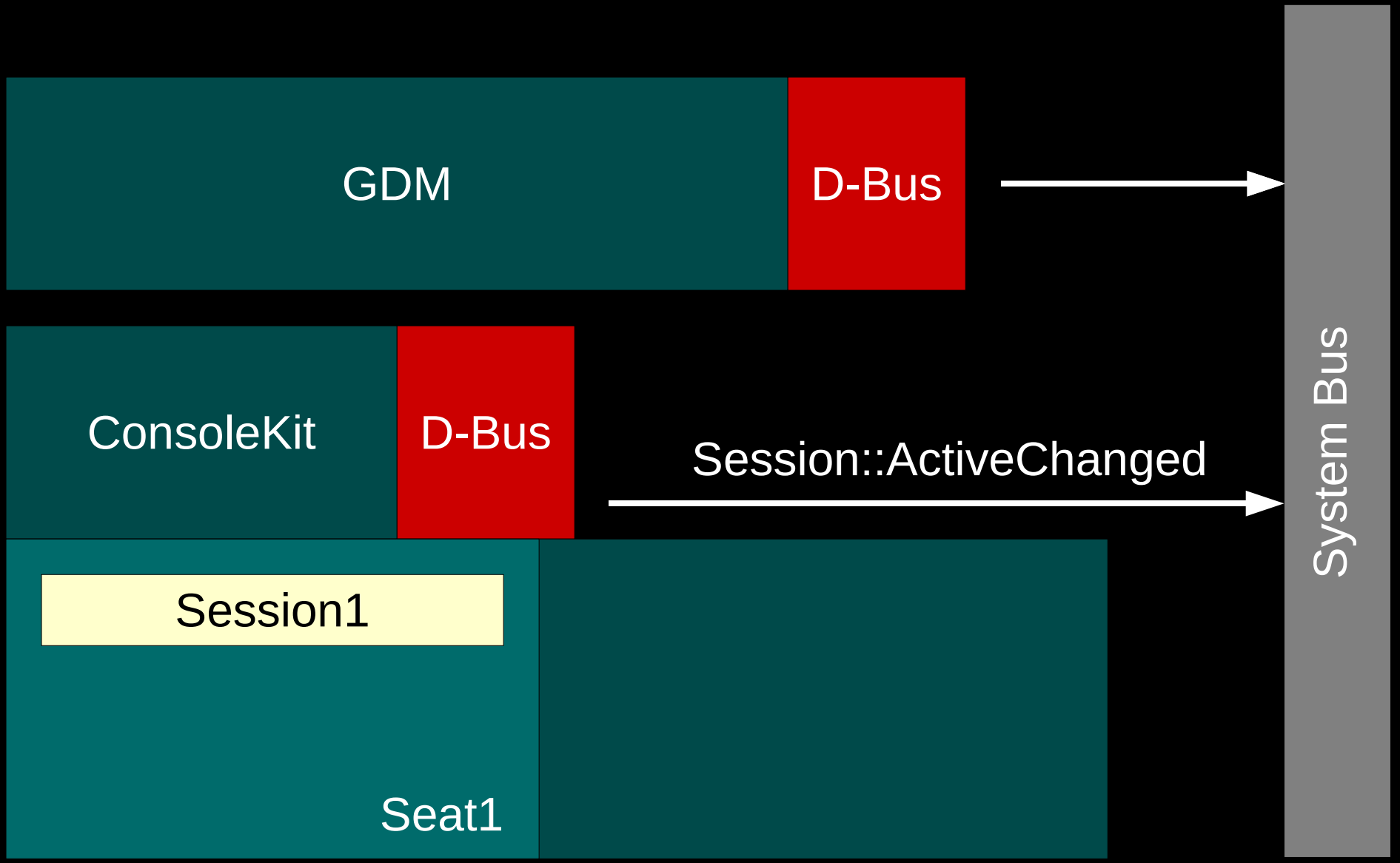
User Switching



User Switching



User Switching



HAL

HAL

- Now that we have well-defined sessions, and session change notifications, HAL can refuse service to inactive sessions and stop polling when the system is idle.
- However, whether a session is active is only one possible criterion for policy decisions...

PolicyKit

- Decide whether something can be done by a given user to a given resource.
- Decide whether a given resource is accessible from a given session.
- Decide what resources belong to which seats.

DisplayManager

- Needs to detect changes in seat configuration.
- Start a greeter per seat.
- Make a greeter more session-like.
- Integrate ConsoleKit more deeply.

GDM2

- Just not up to the task in its current form.
- Serious problems at just about every level.
- Had reached the threshold of maintainability.

A Fresh Start

- In May 2007 I decided to branch gdm and essentially start from scratch.
- http://svn.gnome.org/viewcvs/gdm2/branches/mccann-gobject/

What is interesting about it?

- New configuration system but still uses the old .desktop file backend. Designed to simplify migration to a hypothetical system-wide GConf.
- Replaced the internal and external socket protocols with D-Bus interfaces.
- Dramatically simplified the interface between daemon and greeter.
- Made greeter a real session with gnome-power-manager running.
- Includes a factory greeter display that spawns sessions on new VTs.

Homework: Write an awesome greeter

```
GDM_GREETER_DBUS_ADDRESS

<interface name="org.gnome.DisplayManager.GreeterServer">
  <method name="AnswerQuery">
    <arg name="text" direction="in" type="s"/>
  </method>
  <method name="SelectSession">
    <arg name="text" direction="in" type="s"/>
  </method>
  <method name="SelectLanguage">
    <arg name="text" direction="in" type="s"/>
  </method>
  <method name="SelectUser">
    <arg name="text" direction="in" type="s"/>
  </method>
  <method name="Cancel">
  </method>
  <signal name="Info">
    <arg name="text" type="s"/>
  </signal>
  <signal name="Problem">
    <arg name="text" type="s"/>
  </signal>
  <signal name="InfoQuery">
    <arg name="text" type="s"/>
  </signal>
  <signal name="SecretInfoQuery">
    <arg name="text" type="s"/>
  </signal>
  <signal name="Reset">
  </signal>
</interface>
```

How can I make an application multi-user aware?

How can I make an application multi-user aware?

- Detect multiple instances
- Don't expect exclusive access to devices
- Yield resources
- Know when to take a nap
- Be a good neighbor
- Do your grunt work when system is idle

What's next?

- Finish new DisplayManager
- Integrate with PolicyKit
- Port applications to new interfaces
- Work on multi-seat configuration
- Experiment with hot desking / session migration
- Session hibernation?

How can I help?

- Build / Test / Develop / Discuss
- ConsoleKit / PolicyKit / HAL: http://lists.freedesktop.org/mailman/listinfo/hal
- DisplayManager: http://mail.gnome.org/mailman/listinfo/gdm-list