Pursuing KDE Neon Core

Akademy 2025, Berlin

Kevin Ottens

HAUTE COUTURE

enioka

ENIOKA - MANAGING IT COMPLEXITY

Who am I?

- Started to use KDE with 1.0-beta1 in 1997
- Procrastinated until 2003 to finally contribute code
- Fell in love with the community back then
- Kept doing things here and there… most notably helped with:
  - kdelibs
  - KDE Frameworks architecture
  - the KDE Manifesto
  - Community Data Analytics
- Part of the enioka Haute Couture family
- Living in Toulouse

Previously…

Ubuntu Core

> *Ubuntu Core is a minimal, secure and strictly confined operating system*

Ubuntu Core Desktop

> *A fully containerized desktop, where each component is immutable and isolated*

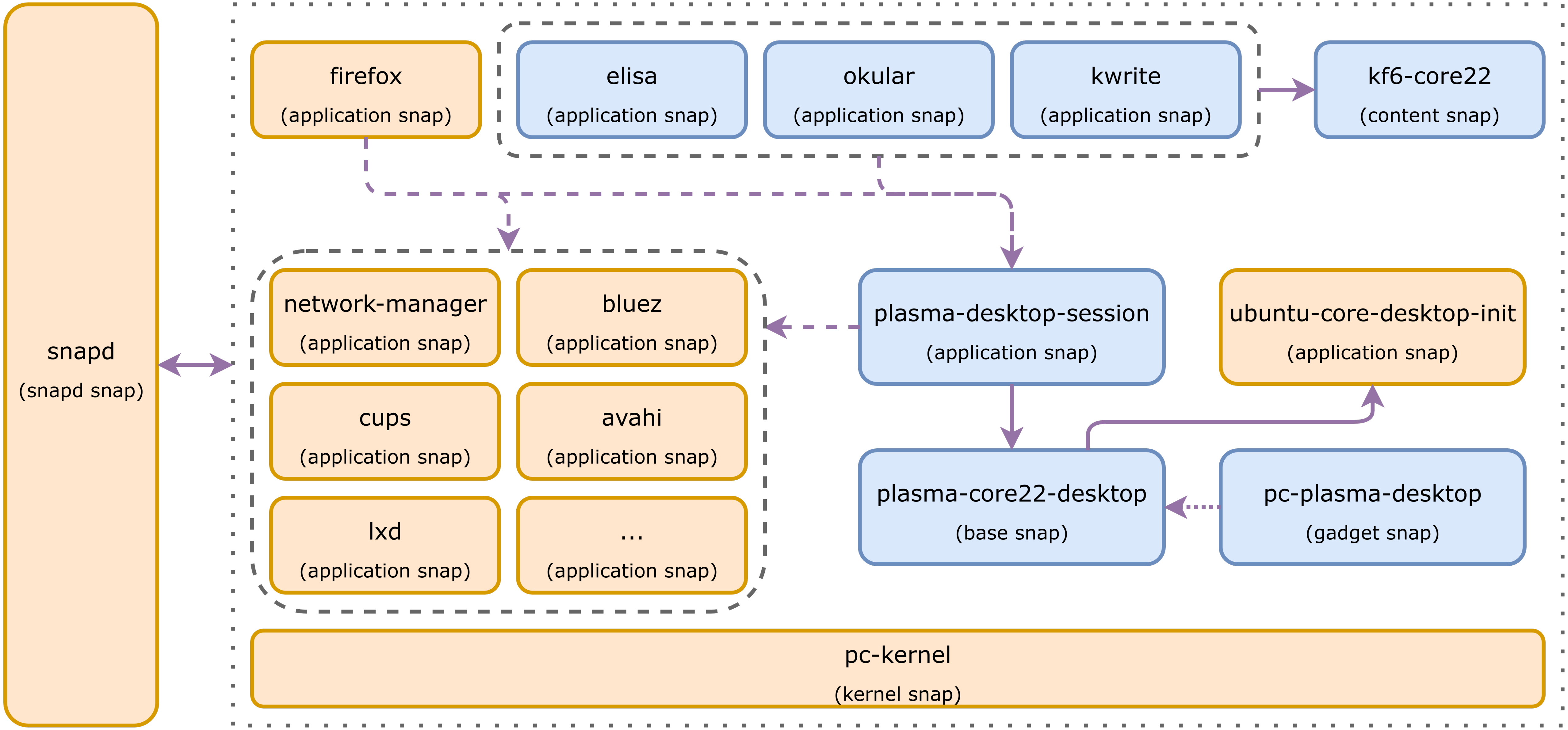
KDE Neon

> *The latest and greatest of KDE community software*

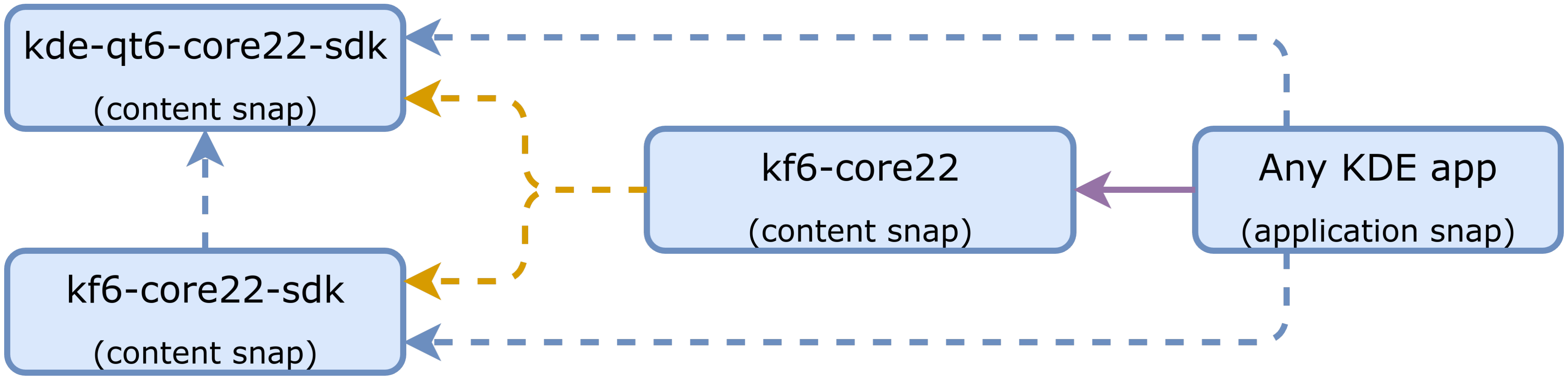
KDE Neon Core

> All the KDE Neon benefits on top of Ubuntu Core

KDE Neon Core Architecture 2024

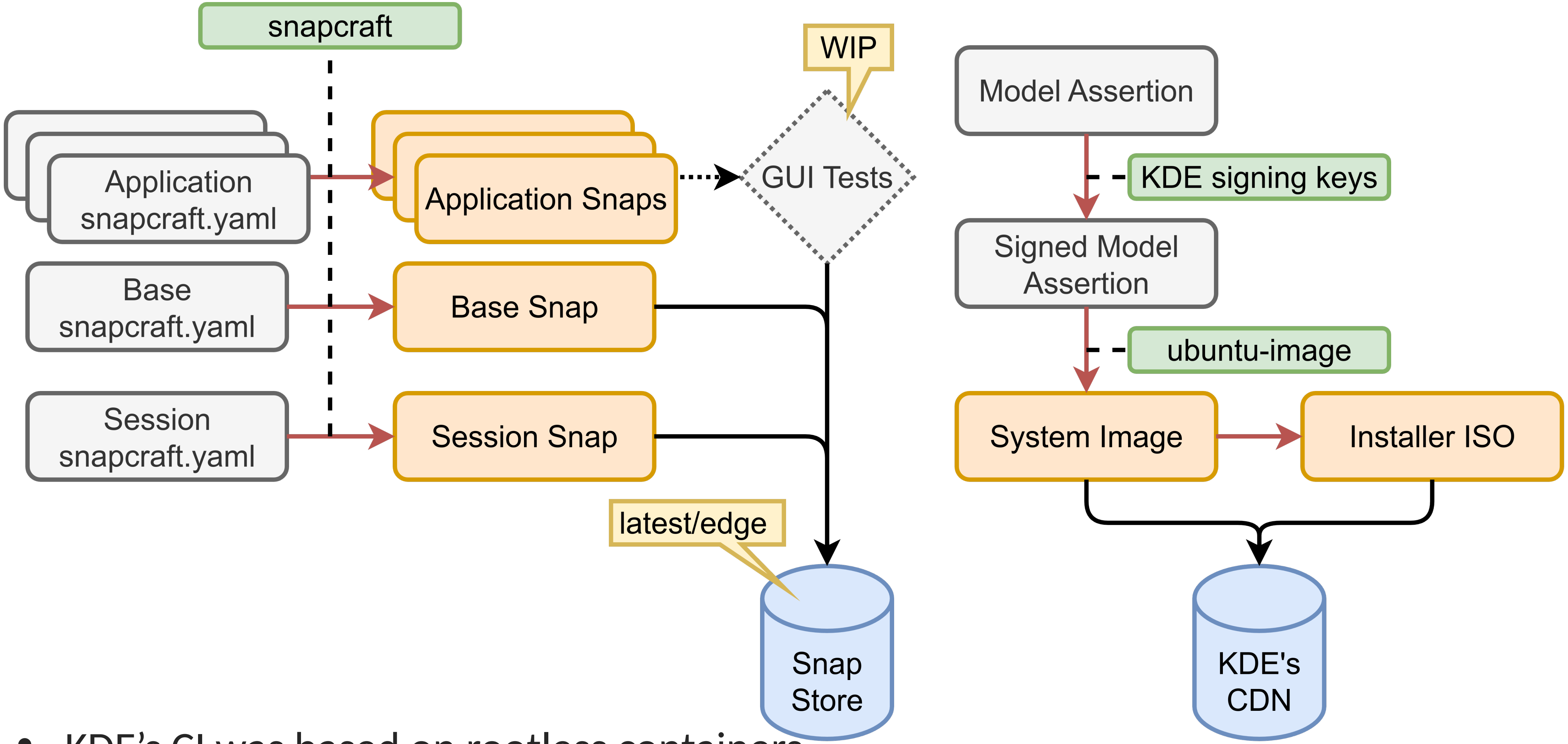


Building Blocks for Apps



- Application and content snaps are populated by building from the code

Build and Publish



- KDE’s CI was based on rootless containers
- We had to use a VM instead

And also, we talked about...

- How Snap confinement works
  - Including how interfaces are implemented (with ties to Seccomp and AppArmor)
  - And how systemd units integration is achieved

- Problems created by some of our services talking to systemd at startup
- Stories of how putting KWin in confinement wasn’t that easy
- Patches we proposed to snapd
- How we approached the problems we encountered and how to debug them

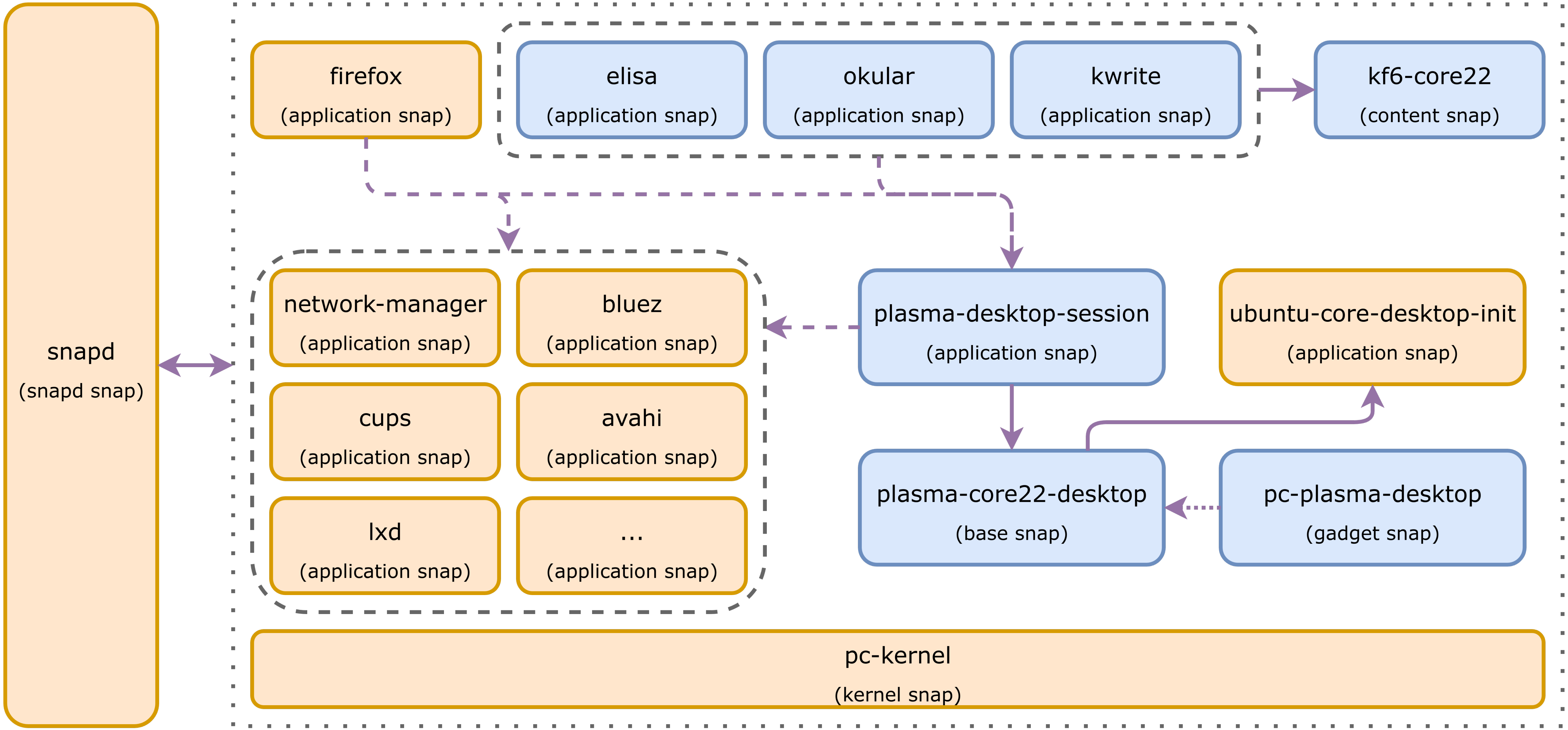
- And black screens... lots of them

A year later…

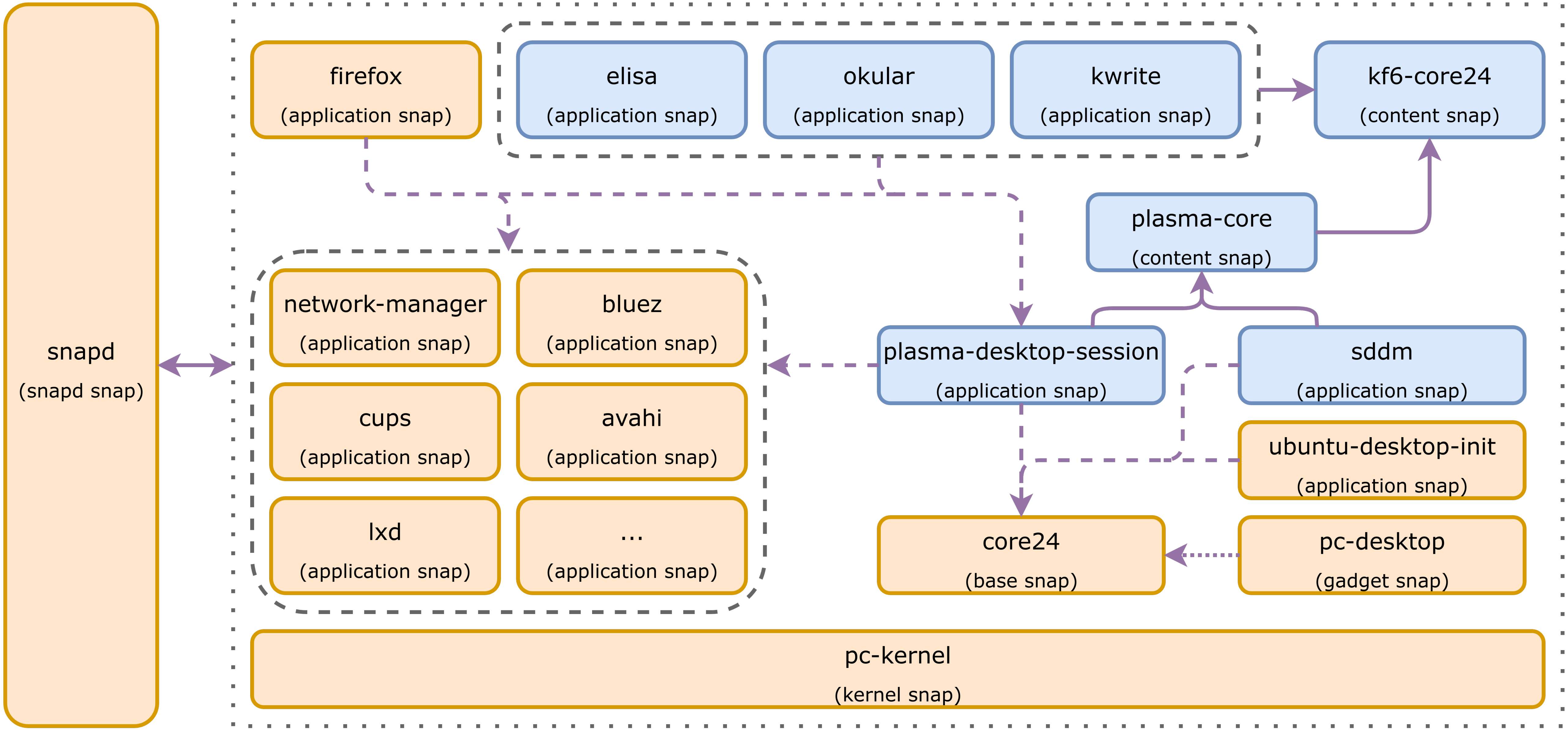
What’s new?

Reworked Architecture

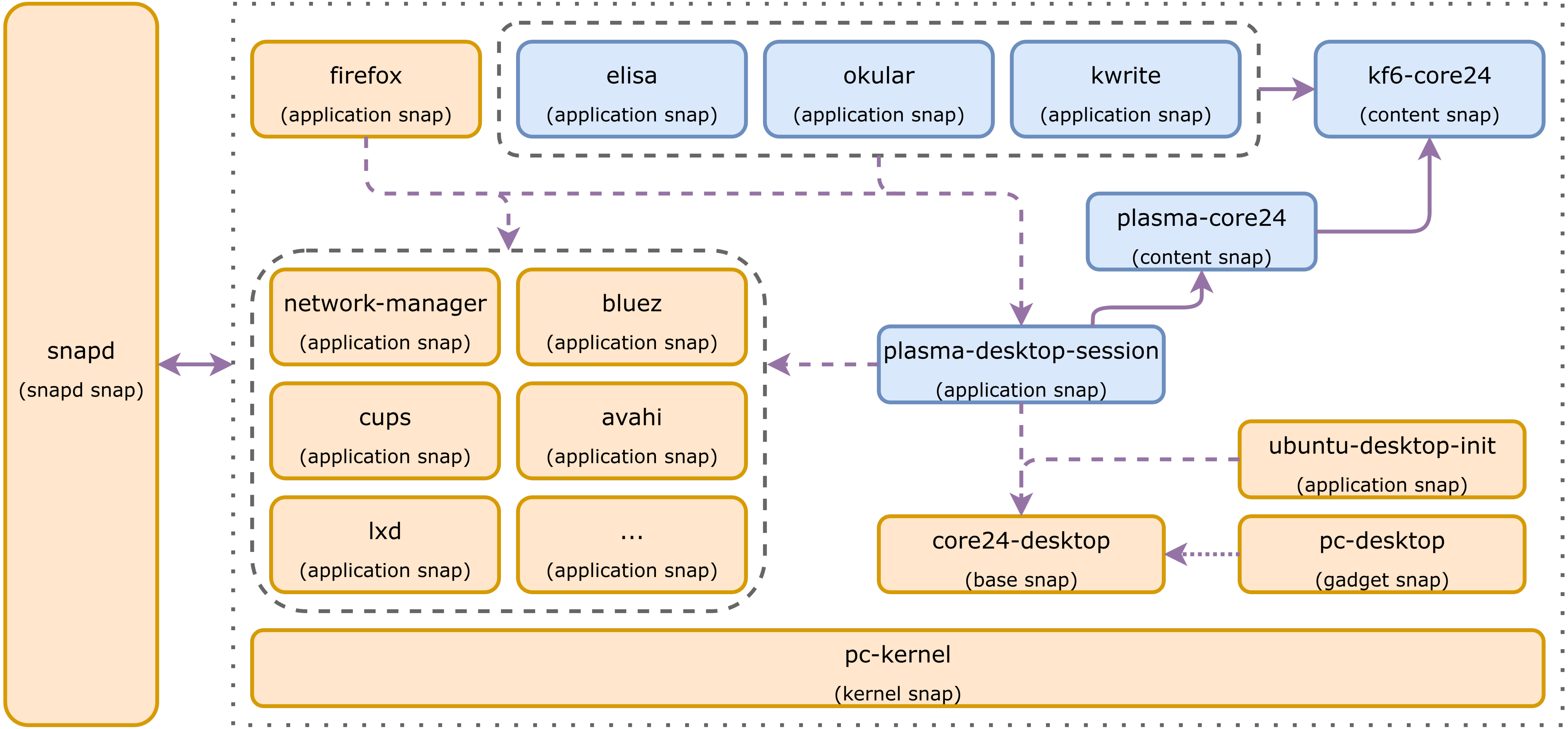
2024 Architecture



Target Architecture



2025 Architecture



Impacts

Relationship to KDE Neon

- Moved from using KDE Neon packages to compiling everything KDE
  - More convenient with the change of prefix
  - Also we're depending on the previously built kf6-core24 runtime anyway

- Still pulling a couple of packages from KDE Neon
- Could probably cut the ties if needed, but…

Relationship to KDE Neon cont’d

- Not depending on KDE Neon archives raises question about our dependencies
- Barely a year past the Noble release
  - KDE Neon already contains some important backports
  - For instance: libdisplay-info2 is needed for KWin

- Could we manage to bump dependencies less aggressively?
- At least provide conditional compilation to deal with API changes in the newer versions of dependencies?
- If not, we’ll need to keep an archive of such backports to stay relevant in the Ubuntu Core ecosystem

Can’t… find my binaries

- Changing the prefix wasn’t a walk in the park… led to missing symbols and other niceties which were quickly fixed
- For Qt/KDE/xdg components, a few environment variables did the job
  - LD_LIBRARY_PATH, PATH
  - XDG_CONFIG_DIRS, XDG_DATA_DIRS, XDG_DESKTOP_PORTAL_DIR
  - QT_PLUGIN_PATH, QML2_IMPORT_PATH
- But some dependencies were less accomodating, had to rely on snap layouts

```
layout:
  /etc/alsa:
    bind: $SNAP/plasma/etc/alsa
  /usr/share/alsa:
    symlink: $SNAP/plasma/usr/share/alsa
```

Can’t… find my systemd services

- Added some exceptions in the session wrapper script from core24-desktop

```
if [ $session_type == "KDE" ]; then
    # Temporary workaround until we have a better way to expose our services and targets
    # 1. Expose our targets, services and overloads
    rm -rf $XDG_RUNTIME_DIR/systemd/user.control
    mkdir -p $XDG_RUNTIME_DIR/systemd

    ln -sf /snap/plasma-core24-desktop/current/usr/lib/systemd/user \
           $XDG_RUNTIME_DIR/systemd/user.control

    # 2. Reload the daemon so that it picks up our changes
    systemctl --user daemon-reload
    [...]
fi
```

Can’t… link to systemd targets

- Remember, we don’t control the name and place of the service files
- The content snap doesn’t expose the files coming from the build
- They are symlinks to the snapd generated service files!
- Which are not part of the graphical-session target…
- We need to tune all this for the session to go well

```
for service in "${snapped_services[@]}"; do
    rm "usr/lib/systemd/user/$service.service"
    ln -s "/etc/systemd/user/snap.plasma-desktop-session.$service.service" "usr/lib/systemd/user/$service.service"
    mkdir "usr/lib/systemd/user/$service.service.d"
    cat >"usr/lib/systemd/user/$service.service.d/override.conf"<<EOF
[Unit]
PartOf=graphical-session.target
EOF
done
```

Can’t… start the desktop portal

- xdg-desktop-portal isn’t confined (even though xdg-desktop-portal-kde is)
- But it is started after startplasma pushed its environment to systemd
- It will thus try to find libraries and config at the wrong place, failing horribly
- Another override to the rescue!

```
mkdir "usr/lib/systemd/user/xdg-desktop-portal.service.d"
cat >"usr/lib/systemd/user/xdg-desktop-portal.service.d/override.conf"<<EOF
[Service]
ExecStart=
ExecStart=/usr/libexec/xdg-desktop-portal -verbose

UnsetEnvironment=XDG_CONFIG_DIRS XDG_DATA_DIRS LD_LIBRARY_PATH PIPEWIRE_CONFIG_DIR

Environment="PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
games:/usr/local/games" "XDG_DATA_DIRS=/snap/plasma-core24-desktop/current/usr/share:"

EOF
```

Can’t… start applications

- Only affected the couple of utilities shipped with the session (systemsettings, kinfocenter…)
- The desktop files were pointing out of the snap!
- Needed to tune them so they point in the content snap

```
prefixes=(
    usr/share/applications
    usr/share/kglobalaccel
)

for prefix in "${prefixes[@]}"; do
    find "$prefix" \
        -type f,l \
        -name "*.desktop" \
        -exec bash -c "i=\$1; sed -i 's:^Exec=/:Exec[\$e]=\$SNAP/plasma/:g' \$i" shell {} \;
done
```

Snapd improvements

- Most of the patches to snapd from last year got merged in snapd upstream

- Pipewire has been moved out of the session and content snaps
  - Meant introducing a specific new interface to allow the session to talk to pipewire directly (needed for desktop sharing)

- There are plans to also remove udisks2 from the base snap
  - This would lead to a similar situation with an ad hoc interface for sessions

Snapd improvements cont’d

- systemd-user-control interface didn’t make it upstream
  - Waiting on a more ambitious approach with a service proxying systemd
  - This is unfortunately still on the drawing board
  - Would avoid some of the pitfalls we encountered

- Even more needed in snapd to fully complete the architecture transition
  - A way to describe services dependencies
  - A way for a snap to start another one (only for login manager use)
  - Proper pam handling (login manager and screen locker)

What else happened?

- Kept pushing packages through the snap store
  - Add quite a bit of delay for testing, can be troublesome
  - To be fair, they're trying to make sure nothing bad gets in

- Efforts on the applications GUI tests continued
  - Overall the Selenium Webdriver AT-SPI has led to very fragile tests
  - Some of those tests had to be dropped (e.g. kate)

- Presented the KDE Neon Core effort at Ubuntu Summit 2024
- Attended snapd architecture meetings, discussing the needed improvements

What else happened? cont’d

- Got to producing ISOs and bootable systems
- Helped test drive the VM based CI, we had an early feel for it
- Efforts have unfortunately been paused…
  - Canonical changed priorities internally
  - Obviously impacted the funding
  - Still, the needed snapd changes to finish the architecture transition won’t appear for the time being

And now what?

A path forward?

- Until the effort resumes on the Canonical side, we could produce a viable product on the current architecture
- Blueprint would be
  - Revive our own gadget and base snap
  - Make the base snap extremely lean, just the necessary to:
    - bootstrap the login manager
    - pre-register allowed session snaps
    - Have essential services like xdg-desktop-portal
  - Produce a snap for plasma-login-manager when it stabilizes
    - This would run unconfined

A path forward? cont’d

- One caveat though

- It relies on an Ubuntu Core Desktop specific snapd
  - Contains a few extra patches we need
  - Like the systemd-user-control interface

- It is dormant as well… but would need to be kept up to date with snapd upstream
- This would require an agreement with Canonical people to make it happen

- Good way to show goodwill from KDE

KDE Benefits

- This gives access to devices where KDE technology is not present yet
- Any Ubuntu Core based device with a screen could use part of our stack all the way to a full session
- This would be usable for
  - TVs
  - Kiosks
  - Point of sales
  - Digital signage
  - Other embedded devices

KDE Benefits cont’d

- Our work is being noticed outside of our usual circles
- Our SDKs have some deployments already, even the newly built ARM64 ones
- I’ve been in touch with people interested in those
- It has the potential to bring more sponsors and projects to the community

Lessons Learned

Lessons Learned

- Selenium Webdriver AT-SPI remains a headache
- Slicing things up in smaller confinement boxes keep bringing surprises…
  - but there's always a knob somewhere
- The bandwidth of the snapd and snap store teams are smaller than you'd expect
- Getting things through the snap store keep taking time and needs to be planned for
- Staying relevant in the Ubuntu space might require us to be more conservative with our dependencies… or accept maintaining backports
- There's more interest for the KDE Stack than you'd expect

Where to Contribute?

- Code
  - https://invent.kde.org/neon/ubuntu-core
  - https://invent.kde.org/neon/snap-packaging
- Documentation
  - https://community.kde.org/Guidelines_and_HOWTOs/Snap
  - https://community.kde.org/Neon/Core
- Come talk to us!

Acknowledgments

- Thanks to the enioka Haute Couture team
  - Benjamin Port
  - Antoine Gonzalez
  - Antoine Herlicq

- Thanks to the contractors we've been working with
  - Scarlett Moore
  - Carlos De Maine

Upcoming BoF

Immutable distros & Obsalat BoF

Monday 8, 14:00

Room 1

Thanks for your attention
Questions ?

Kevin Ottens
kevin.ottens@enioka.com