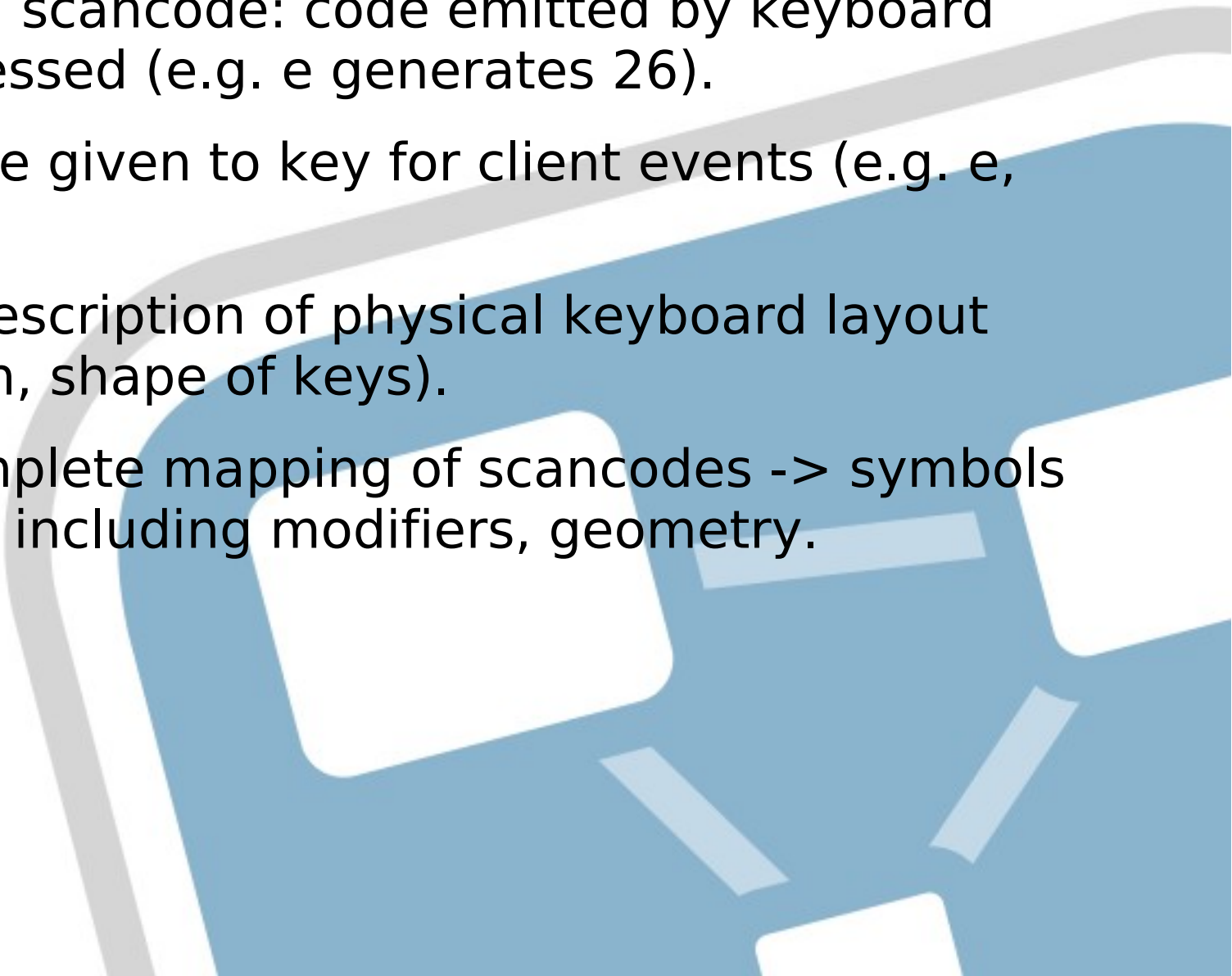


# XKB: The Way to Be

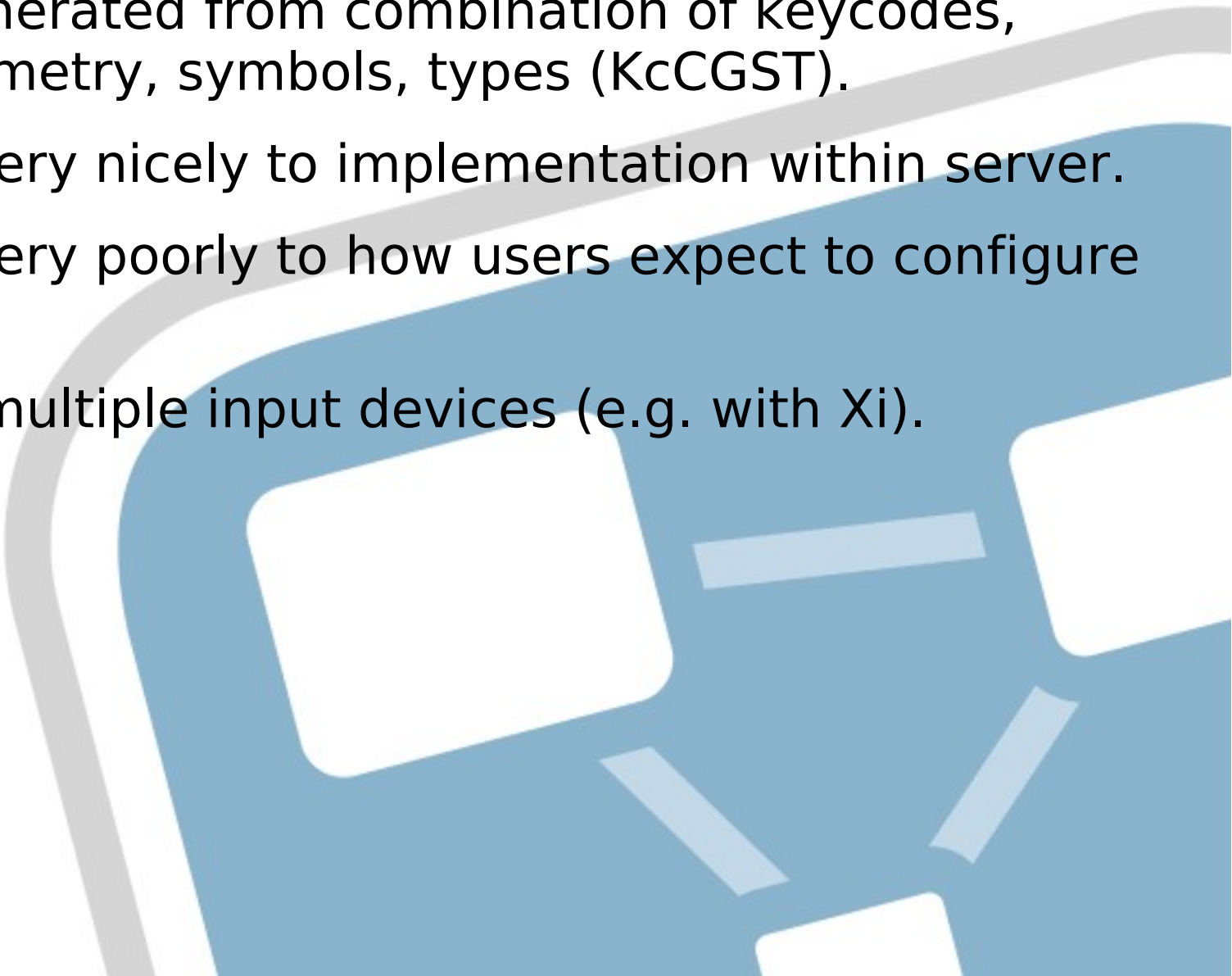
Daniel Stone  
'hacker, iterant idiot'  
daniel@fooishbar.org



# Definitions

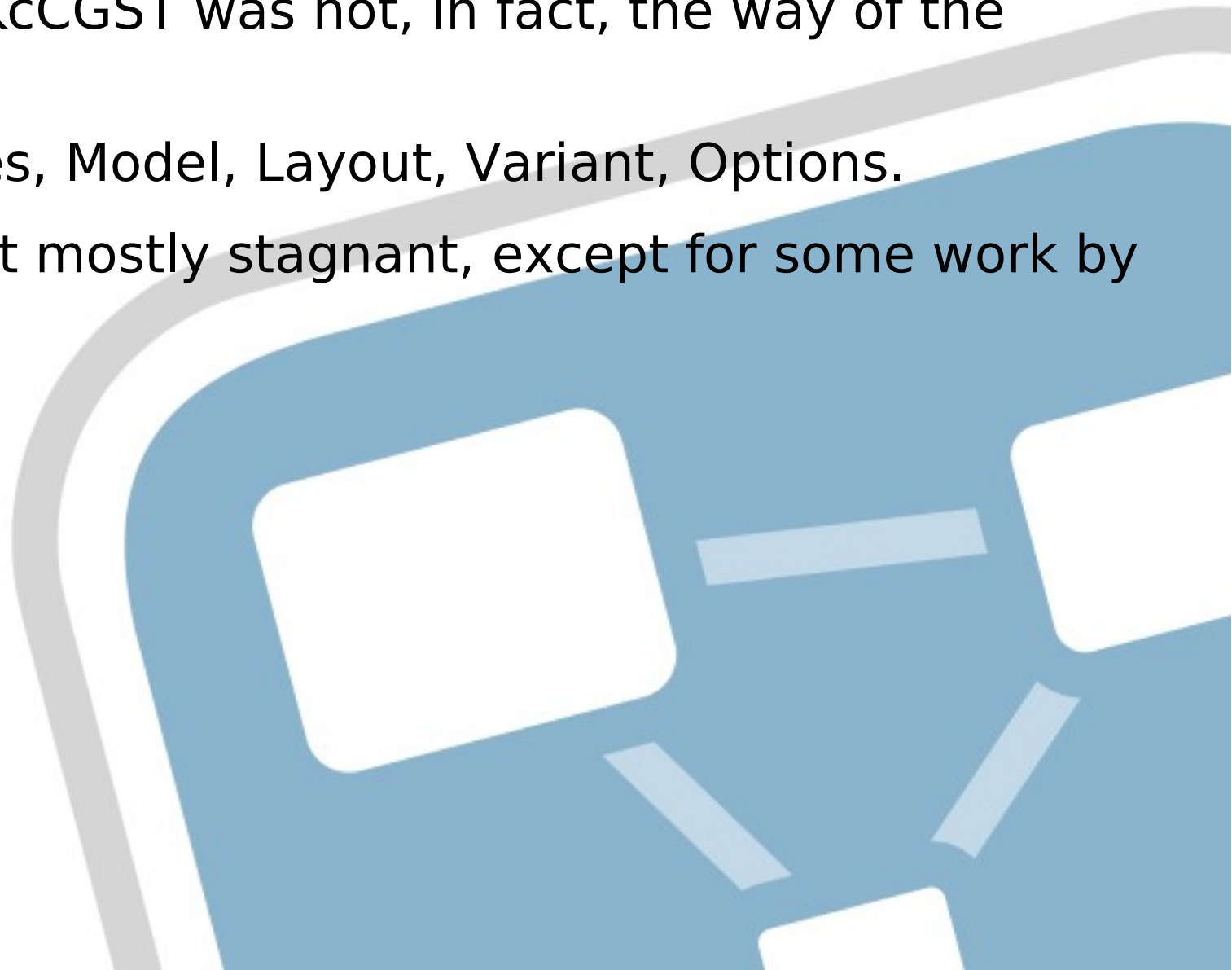
- Keycode, aka scancode: code emitted by keyboard when key pressed (e.g. e generates 26).
  - Symbol: name given to key for client events (e.g. e, Xfree86Play).
  - Geometry: Description of physical keyboard layout (size, location, shape of keys).
  - Keymap: complete mapping of scancodes -> symbols on keyboard, including modifiers, geometry.
- 
- A stylized, semi-transparent blue graphic of a keyboard is overlaid on the right side of the slide. It shows the outlines of several keys, including a large white key in the center, and the surrounding frame in shades of blue and grey.

# XKB in the SGI Age

- Keymaps generated from combination of keycodes, compat, geometry, symbols, types (KcCGST).
  - Maps very, very nicely to implementation within server.
  - Maps very, very poorly to how users expect to configure keyboards.
  - Unaware of multiple input devices (e.g. with Xi).
- 
- A stylized, abstract graphic of a keyboard in shades of blue and white, positioned in the lower right quadrant of the slide. The keys are represented by simple geometric shapes, and the overall design is clean and modern.

# XKB in the XFree86 Age

- Realisation KcCGST was not, in fact, the way of the future.
- RMLVO: Rules, Model, Layout, Variant, Options.
- Development mostly stagnant, except for some work by Ivan Pascal.



# RMLVO: A New Dawn

- Rules file: mapping of all of the above to legacy definitions (e.g. requesting layout us, variant intl, adds '+us(intl)' to the symbols list).
- Model: Physical model of keyboard – usually pc104. Other models include sun, and abnt2 for Brazil.
- Layout: Which symbols to attach to the certain keys (e.g. US-based QWERTY, German QWERTZ, French AZERTY).
- Variant: Minor variations on layouts (e.g. removing dead keys).
- Options: Other miscellaneous changes (e.g. use Caps Lock as an additional Control key).

# RMLVO in Theory

Very, very nice.



# RMLVO in Practice

Suck.



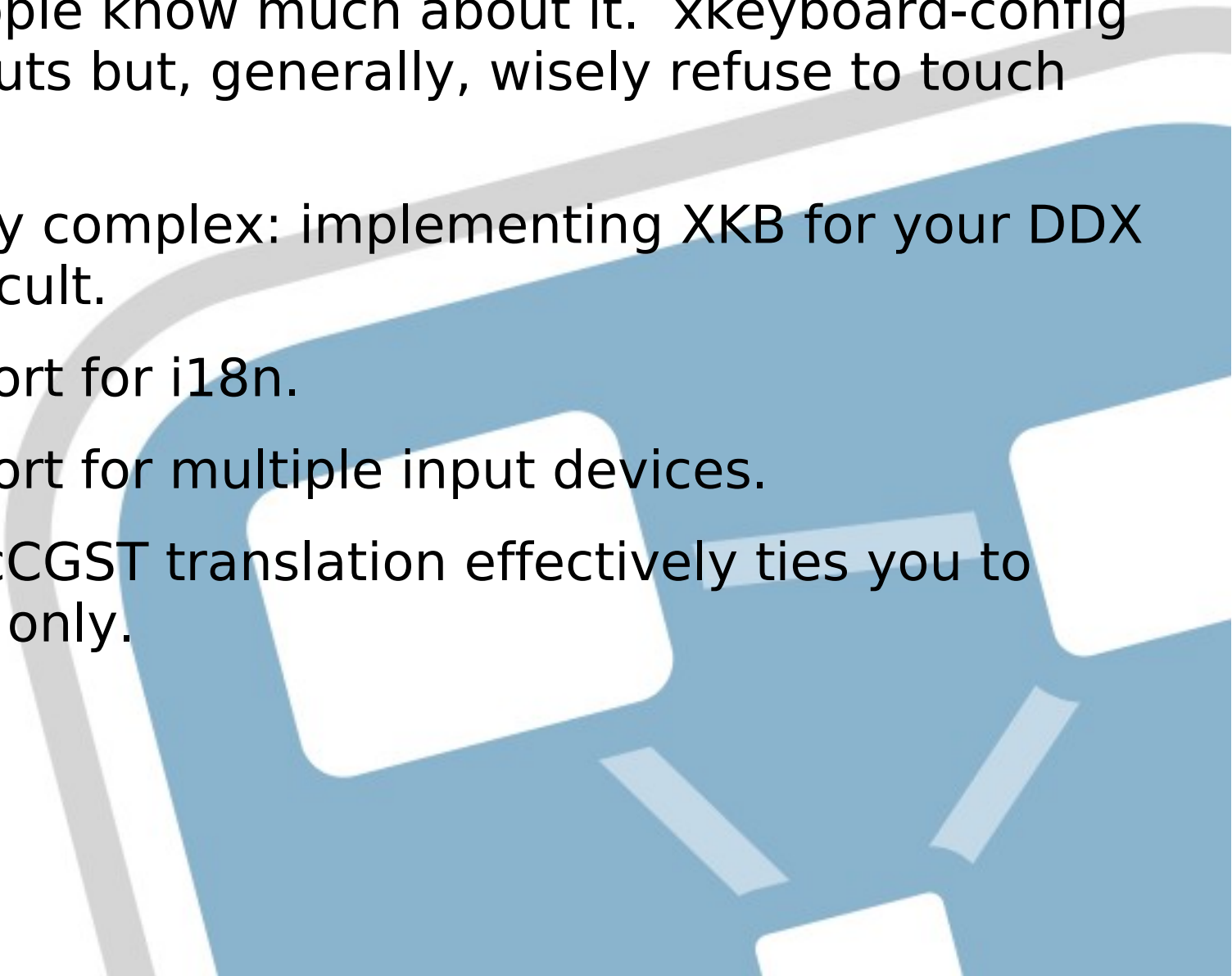
# Problems with RMLVO

- Bolted on to the current protocol. Clients perform RMLVO -> KcCGST mapping, then make KcCGST request over the wire.
- Bolted on to the server implementation: hacky support for setting initial RMLVO map in XFree86 DDX but otherwise difficult.
- Difficult from a client point of view: no uniform way to get the list of available components, no uniform way to set them.

# Problems with XKB

- Massive codebase -- 880KB, 28141 lines of code in server; 424KB, 12749 lines of code in libX11; 348KB, 10670 lines of code in libxkbfile; 52KB, 754 lines of code in libxkbui, 624KB, 18925 lines of code in xkbcomp. Total 1.9MB, 71239 lines of code.
- Massive code duplication. Some functions retain the name but change purpose entirely based on `XKB_IN_SERVER` define.

# Problems with XKB (cont.)

- Very few people know much about it. xkeyboard-config guys do layouts but, generally, wisely refuse to touch code.
  - Unnecessarily complex: implementing XKB for your DDX is rather difficult.
  - Lack of support for i18n.
  - Lack of support for multiple input devices.
  - RMLVO -> KcCGST translation effectively ties you to local servers only.
- 
- A stylized, semi-transparent blue graphic of a keyboard is overlaid on the right side of the slide. It shows the outlines of several keys, including a large white key in the center, and is partially obscured by the text of the list.

# Problems with XKB (cont.)

- ddxLoad.c:
  - Receives KcCGST request from client, which has compiled from RMLVO.
  - Runs system() on xkbcomp, requesting KcCGST layout be compiled.
  - xkbcomp writes to XkbDir/compiled.
  - ddxLoad reads in custom weirdo binary format.
  - Et voilà!
  - Numerous stack smashes and shell injections only fixed recently. Shell injections may well still remain today.

# Future directions

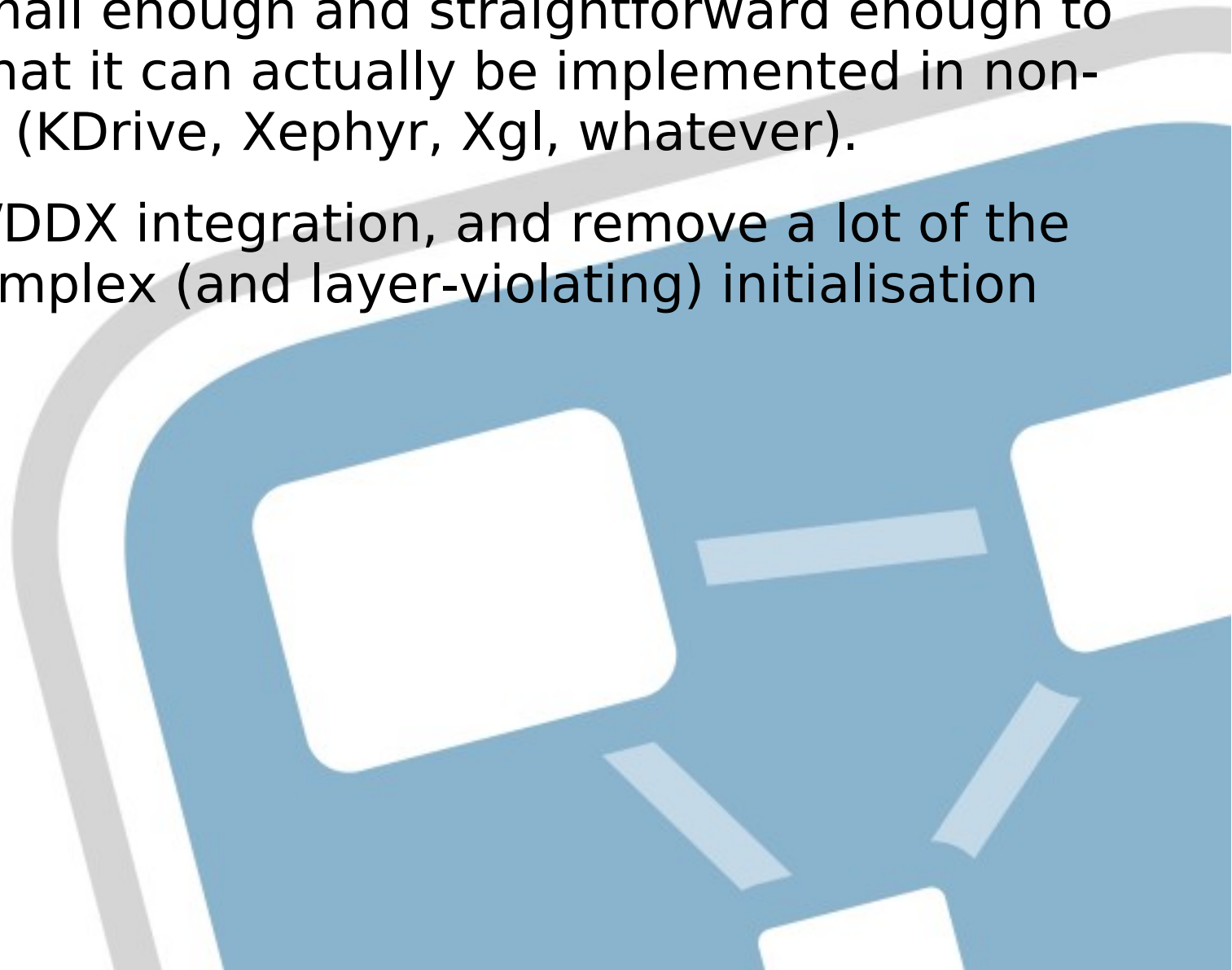
- Add RMLVO requests to the wire protocol; client discovers which keymaps are available directly from the server, asks the server to set an RMLVO keymap.
- Add i18n support to this: client requests descriptions from a specific locale. Read out of XML rules file instead of plain-text.
- Requests become Xi-aware; keymaps are per-device.
- Remove ddxLoad.c abomination.

# Future directions (cont.)

- Turn libxkbfile into a proper API for dealing with XKB on-disk: fold xkbcomp functionality into that, so you can request libxkbfile to compile a given RMLVO set and hand you back the XkbDescPtr.
- Remove XKM (custom weirdo binary format) support: pre-compiling your keymaps on the client is not particularly useful. Certainly not worth the code weight this imposes.
- Eventually deprecate KcCGST over the wire?

# Future directions (cont.)

- Should be small enough and straightforward enough to implement that it can actually be implemented in non-Xorg servers (KDrive, Xephyr, Xgl, whatever).
- Simplify DIX/DDX integration, and remove a lot of the unusually complex (and layer-violating) initialisation code.



# Current status

- Server accepting RMLVO requests over the wire, gets an XkbDescPtr back through libxkbfile.
- Initialisation sequence significantly cleaned up.
- ~7000 lines of code removed from the server in the process; some thousands from the client side also.
- Should be in for the server released with 7.2 (7.1 is close now, remember!).
- xkeyboard-config guys seem happy with it (based off svu's proposal on [xorg@lists.freedesktop.org](mailto:xorg@lists.freedesktop.org)).

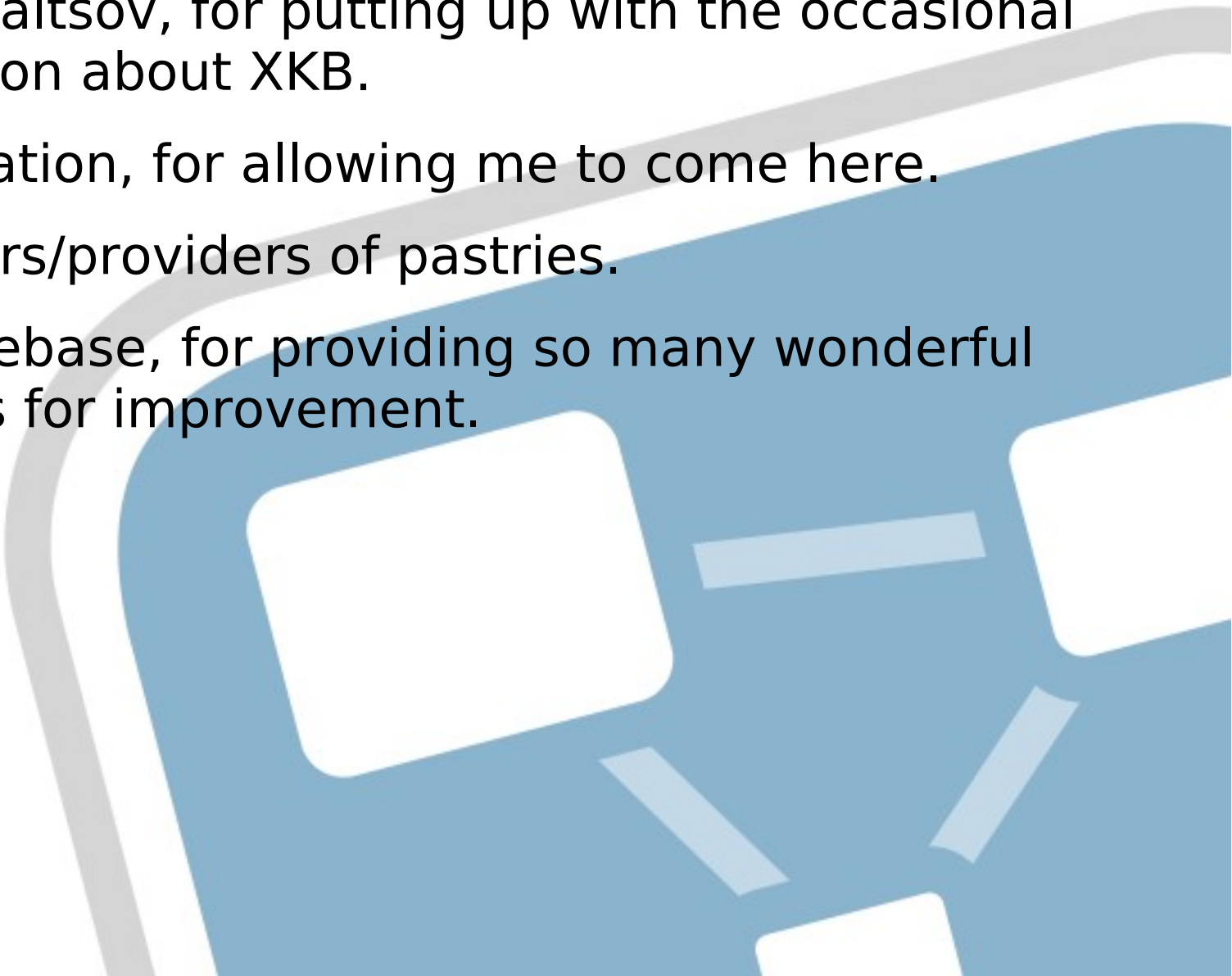
# Outstanding questions/problems

- Should the old wire requests eventually be removed completely? Thinking of KcCGST requests in particular, to make it RMLVO-only. Would allow server codepaths to be simplified.
- The code is still incredibly hairy, and needs some serious cleanup action. Good janitor-style task if anyone's really bored.
- How exactly to interact with Xi? Notion of one keymap per screen disappears with multiple keyboards, possibly with different layouts.

# Questions?



# Shoutouts

- Sergey V. Udaltsov, for putting up with the occasional stupid question about XKB.
  - X.Org Foundation, for allowing me to come here.
  - The organisers/providers of pastries.
  - The XKB codebase, for providing so many wonderful opportunities for improvement.
- 
- A large, stylized graphic of a keyboard key, rendered in shades of blue and white, is positioned in the lower right quadrant of the slide. The key is shown from a slightly elevated perspective, with its rectangular shape and a central rectangular area. The background behind the key is a light blue gradient.