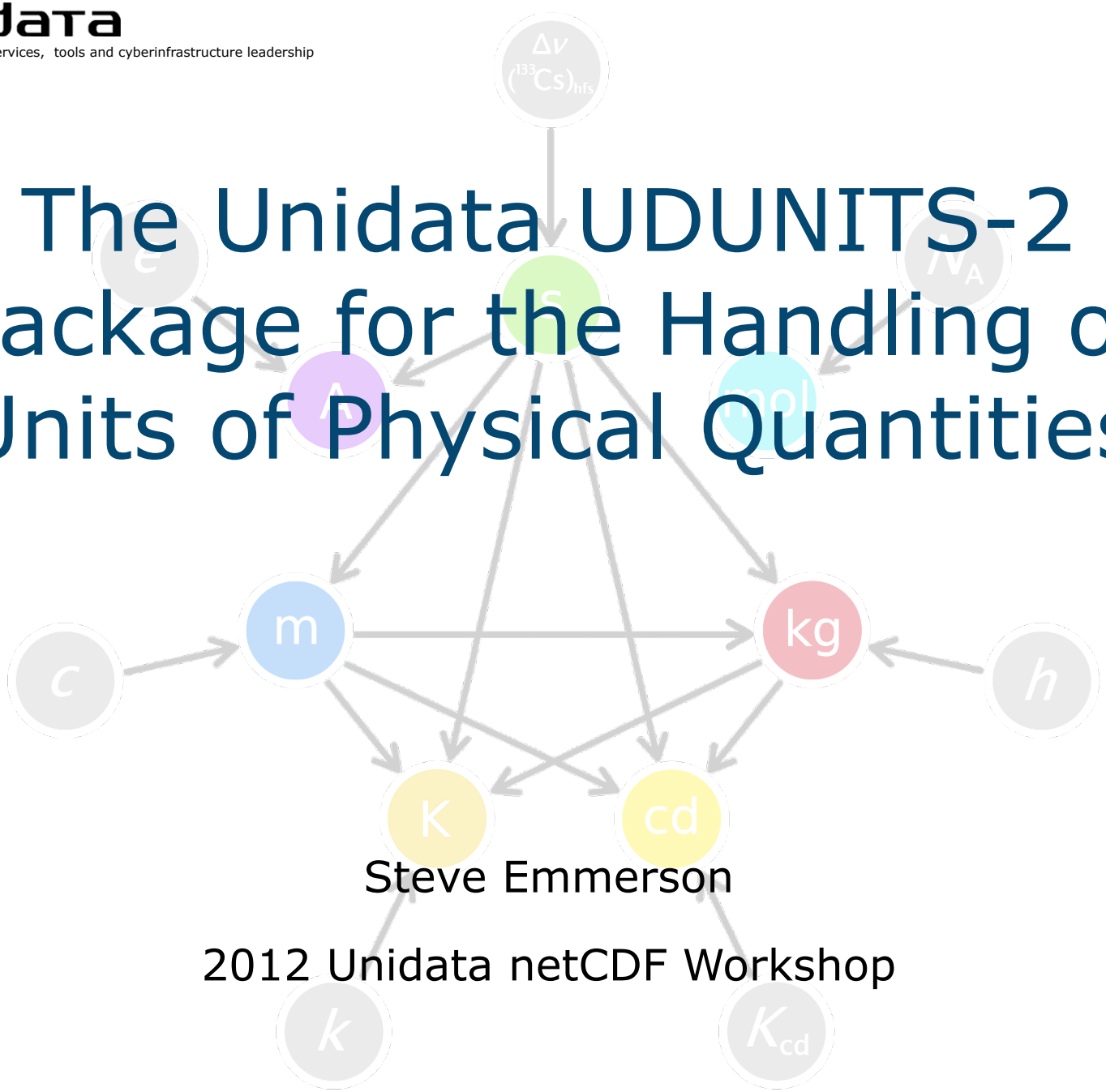
unidata
providing data services, tools and cyberinfrastructure leadership

The Unidata UDUNITS-2 Package for the Handling of Units of Physical Quantities



Steve Emmerson

2012 Unidata netCDF Workshop

What About the Package?

- Programmatically converts between string specifications of units of physical quantities and internal binary representations
- Operates on the binary unit representation
  - Unary Operations: raise to a power, logarithm, etc.
  - Binary Operations: multiply, divide, compare, etc.
- Converts numeric values between binary unit representations
- Has a library and a utility (program)
- Has a user-editable database
- Written in C

Converting from String to Binary

```
include <udunits2.h>
...
ut_system* system     = ut_read_xml(NULL);
ut_unit*   watt       = ut_parse(system, "joule/second", UT_ASCII);
ut_unit*   fahrenheit = ut_parse(system, "K/1.8 @ 459.67", UT_ASCII);
ut_unit*   wattPerOhm = ut_parse(system, "(kg·m²)/(s³·Ω)", UT_LATIN1);
```

The supported characters encodings are US ASCII, ISO 8859-1 (Latin-1), and UTF-8 (Unicode)

Unary Operations

```
include <udunits2.h>
…
ut_unit* yard    = ut_scale(3, foot);
ut_unit* celsius = ut_offset(kelvin, 273.15);
ut_unit* meter2  = ut_raise(meter, 2);
ut_unit* meter   = ut_root(meter2, 2);
ut_unit* bel1mW  = ut_log(10, milliwatt);
…
ut_free(yard);
```

Binary Operations

```
include <udunits2.h>
…
ut_unit* newtonMeter    = ut_multiply(newton, meter);
ut_unit* meterPerSecond = ut_divide(meter, second);
…
if (ut_compare(unit1, unit2) == 0) {
    /* Same units */
    …
}
```

Converting Values

```
include <udunits2.h>
…
if (ut_are_convertible(foot, meter)) {
   cv_converter* ft2m   = ut_get_converter(foot, meter);
   float         meter  = cv_convert_float(ft2m, 3);
   float*        feet   = …; /* input array of n floats */
   float*        meters = …; /* output array of n floats */

   cv_convert_floats(ft2m, feet, n, meters); // _doubles(…)
   cv_free(ft2m);
}
```

UDUNITS-2 Utility

```
$ udunits2 -A
udunits2: using default XML database
You have: watt
You want:
    m2.kg.s-3
You have: furlongs/fortnight
You want: cm/minute
    1 furlongs/fortnight = 0.997859 cm/minute
    x/(cm/minute) = 0.997859 (x/(furlongs/fortnight))
You have: ^D
$
```

XML Database

```
<?xml version="1.0" encoding="US-ASCII"?>
<unit-system>
   <import>udunits2-prefixes.xml</import>
   <import>udunits2-base.xml</import>
   <import>udunits2-derived.xml</import>
   <import>udunits2-accepted.xml</import>
   <import>udunits2-common.xml</import>
</unit-system>
```

XML Database of Common Units

```
<?xml version="1.0" encoding="US-ASCII"?>
<unit-system>
    <unit>
        <def>4.5359237e-1 kg</def>
        <aliases>
            <name><singular>avoirdupois_pound</singular></name>
            <name><singular>pound</singular></name>
            <symbol>lb</symbol>
        </aliases>
    </unit>
    …
</unit-system>
```

Challenges

- No Fortran support in the package yet, but Michel Valin has a Fortran-2003 interface module on GitHub:

  https://github.com/mfvalin/wrapper-code

- Timestamp "units"
  - Example: "**`seconds since 2012-10-10 17:17:00 -6:00`**"
  - Gregorian calendar only (ergo, no 360-day calendars)
  - No leap seconds
  - Appropriate for simple situations only

Resources

- Web Site Homepage

  http://www.unidata.ucar.edu/software/udunits

- FOSS Repository

  https://github.com/Unidata/UDUNITS-2

- Email Support

  support-udunits@unidata.ucar.edu