UTC inventory for Astronomical software

(plus time handling in software)

Rob Seaman

NOAO Science Data Management

What some said about Y2K

"We don't have any problems yet… We'll deal with the problem in the year 2000."

- Vladislav Petrov
  Russian Atomic Energy Ministry

Both like and unlike Y2K

- Urgency is artificial for UTC
- Cry wolf effect:
  - Y2K was real, but seen after as non-event
  - UTC issue is real, *currently* unfamiliar
- Leap seconds are a means to an end

Meanwhile in astronomy

- Entire astronomical software community had Y2K remediation responsibilities, *e.g.*:
  - NOAO had a telescope that tracked backwards
  - IRAF had to be patched
  - FITS standard had to be modified
  - Observatory admin software had to be updated

…

- Cost estimates are costly

Planning for absent planning

- If UTC is redefined, the leap second issue will no longer be a hypothetical exercise
- The astronomical community would be forced, at a much larger scale than Y2K, to conduct a comprehensive *(expensive)* inventory of its software, systems, processes, metadata and documentation

Planning for absent planning

- Afterward the functionality would be no better than before, likely worse
- Confusion injected throughout community
  - and this is only the Y2K-like aspects
- Algorithms would have to change to reflect the different time scale
- Archives would have to deal with both old and new time scales indefinitely
- New clocks and network infrastructure needed

What do civilians say about UTC and civil timekeeping?

Internet commentator, Tom Scott

Time zones versus leap seconds

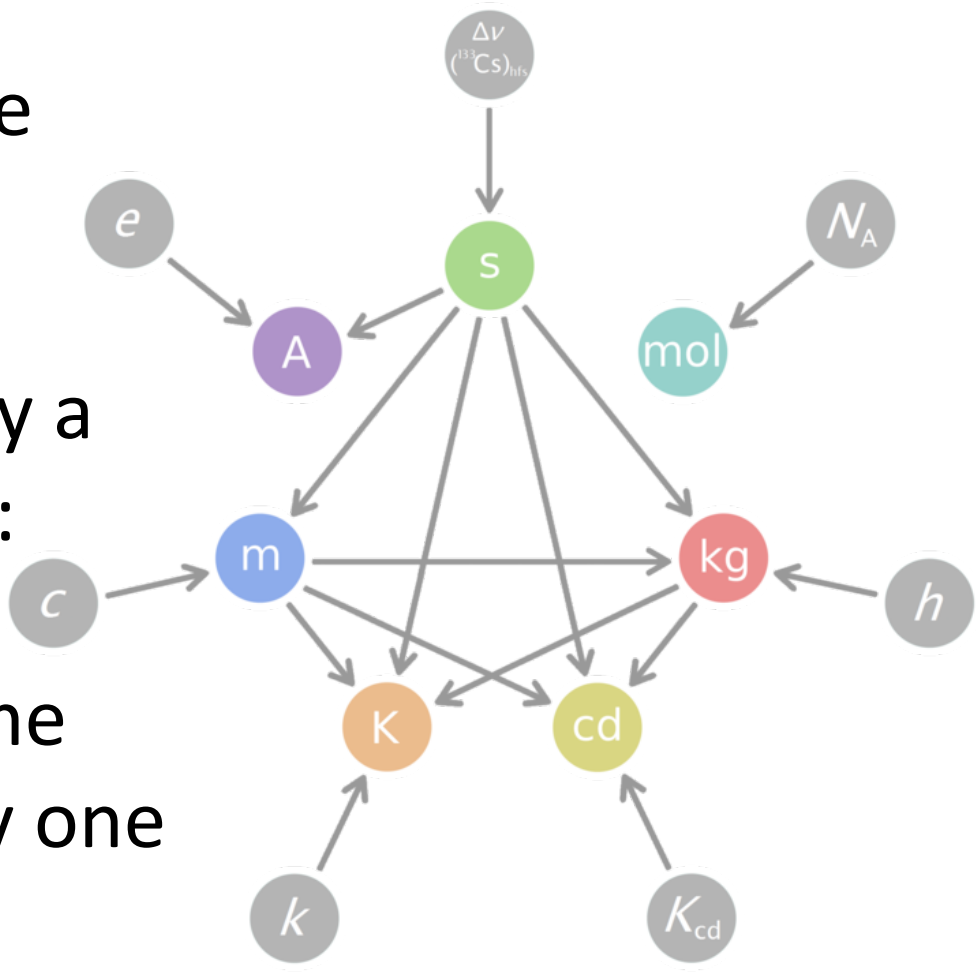
- Extent of planning for the immediate impacts of redefining UTC (that is, for astronomers)
  - "Give them five years and they'll figure it out"
- Long term planning:
  - "Can't predict what future timekeepers will do"
  - But *maybe* the embargoed leap seconds can be absorbed into the timezones and DST *(few countries observe DST)*
- Google (and Amazon and others) have implemented solutions

Time zone references

- Arthur David Olson (tz) database
  - *http://www.iana.org/time-zones*
- Steve Allen's tz-based leap second proposal
  - *http://futureofutc.org/2011/preprints/45_AAS_11-681_Allen.pdf*
- "One thing happens before another"
  - *http://research.microsoft.com/en-us/um/people/lamport/pubs/pubs.html#time-clocks*

What are the use cases?

- Is duration really the "killer" time app?
- SI second is arguably a frequency standard:
- Assumption that time means one and only one thing



Civil timekeeping requirements

- Southern time zones are important to astronomers
  - Logistics, data flow
- DST and time zones have very short notice updates compared to leap seconds
  - Both prior and retroactive use cases
  - Intercalary changes will always be frequent
- Overlaid multiple time zones *(scales)* exist
  - West Bank and Western China

Overly simple conceptual model

- Ignorance of roles of IAU, ITU, IERS, *etc.*
  - "Universal Coordinated Time" *vs* "Astronomical Time"
- Everything *doesn't* break with leap seconds
  - only minor issues after two dozen leap seconds
  - unlike Y2K *(without remediation)*
  - but all clocks in Libya *did* break due to DST
- Unix timestamp is asserted to be a solution
  - but advice is to accept spaghetti code
  - while tz DB could be a pragmatic solution for leapsecs

Can a non-solution be a solution?

- "That way lies madness"?
  - Is time more difficult than other software issues?
- Exasperation is not justification for failing to attempt to solve a problem
- "Continuity is more important than accurate time"?
  - What does this even mean?

Even so…

- He does *not* argue to eliminate leap seconds, time zones or daylight saving time
- Lists several specific things wrong with time zone / DST system…but nothing specific about leap seconds *(multi-radix is descriptive)*
  - Is it because time zones / DST are artificial?
  - Whereas leap seconds are a means to an end modeling a physical aspect of the real world?

UTC inventory for Astronomical software

For more details

http://futureofutc.org/2011/preprints/AAS_11-677_Seaman.pdf

Y2K remediation

- Issue was that two digit years had been encoded into software
- Solutions included
  - Recoding with four digits
  - Introducing an explicit pivot
  - Retiring software
  - Replacing hardware
  - Documentation and procedures

UTC versus Y2K

- UTC is broader impact and less clear-cut
  - astronomy software cares deeply about UTC
- Systems *(HW + SW + processes)* assume

  UTC == Universal Time *(UT1)*, or

  UT1 = UTC +/– DUT1
- In the first case, we need to introduce a new distinction of time scales *(terminology)*
- In the second case, need to vet as Y2K *(0.9s)*

What needs to change?

- Algorithms have to accommodate changes
- New infrastructure to supply UT1 and/or DUT1
- Requirement for leap second table never vanishes, code gets "spaghettier"
- Documentation would become more complex and contradictory *(or worse yet, not change)*
- Revised data and metadata definitions

Systems assuming UTC ≈ UT

Would need to do at least one of:

1. be rewritten to distinguish between two separate meanings of “Universal Time”
2. be isolated to receive a vetted UT1 input *(or UT1-like input)*
3. be retired and/or replaced

Systems with *UT1 = UTC ± DUT1*

Software and systems making a DUT1 correction would need these steps:

4. be vetted for proper operation under values of DUT1 > ±0.9s, and

5. be isolated to receive a vouched DUT1 input, likely from a new source

Classes of astronomical software

- Extremely diverse
- A few examples:
  - Observing preparation tools
  - Astrometry & catalogs
  - Telescope control
  - Instrument control
  - Data handling
  - Data transport

Classes of astronomical software, #2

- More examples:
  - Archives
  - Pipeline processing
  - Virtual Observatory / Astro-informatics
  - Time domain astronomy / transient alerts
  - Robotic follow-up
  - Desktop data reduction & analysis
  - more…

Justifying a UTC inventory

Proponents of redefining UTC make assertions:

1. the affected codebase is small, and
2. there is also a cost for issuing leap seconds

Performing a UTC inventory

Each is an argument for a coherent inventory:

- Is the range of affected software systems small?
  Then it should be easy to complete an inventory.
  - *Or perhaps the inventory would be larger than imagined?*

- Is there a cost for leap seconds?
  Then this should form part of the inventory.
  - *Or would the cost for leap seconds be found to be negligible?*

UTC remediation would include

- Consistent local and community-wide planning
- An inventory of dependencies
  - *For Y2K was basically "19", "year" and "century"*
  - *For UTC, varies and is subtle and more involved*
- Resources
  - *For Y2k was fraction of several NOAO staff for three years, ~ 1.5 – 2.0 FTE-years*
  - *For UTC would be significantly more*
- New clocks and network infrastructure

UTC search terms

- The search terms will vary with software package
- For IRAF, an initial inventory was performed with these terms, roughly in descending order of efficiency in generating good hits:
  - *UT, UTC, GMT, JD, MJD, DUT, LST*
  - *Hour, minute, second*
  - *Year, month, day*
  - *Solar, sidereal*
  - *Clock, calendar*

UTC search terms, *cont.*

- Other terms are too general
  - *Date, time*
- And others simply do not appear
  - *Leap second*
  - *Intercalary*
- Will vary for other packages and SW types

UTC inventory for IRAF

Search Term	Number of files	Term	Files
UT	250	Day	156
UTC	23	Month	68
GMT	38	Year	100
JD	158	Sidereal	20
MJD	63	Solar	65
LST	67	Calendar	10
Second	857	Clock	73
Minute	66		
Hour	145	*Total*	*1312 (of 11,600)*

UTC comparison to Y2K

- 1,312 / 11,600 = 11%
  - UTC tally excludes documentation
  - Also excludes the external packages
  - Each file counted only once
- The IRAF Y2K tally was 124 files (including documentation), less than 1%
- IRAF UTC remediation would be a larger, longer term, more expensive project

Then what?

- A good hit is a file with a plausible connection to timekeeping *(most of these are plausible)*
- With Y2K the search terms resulted in a short list of hits with a high yield of needed changes
- The ultimate goal is to identify all files requiring mitigation, without fail
- For scientific code this requires human review to comprehend the intent of the methods / algorithms *(vs recognizing 2-digit structures)*
- data formats, data structures, interfaces
- documentation

Big impact on astronomers

- Our software is the canary in the coalmine for this issue
- Not a zero-sum trade-off – past leap seconds remain to be dealt with
- Large cost to the community simply to perform the necessary inventory
- No significant benefit to us from ceasing leap seconds