SWCP Portal

August 2026

Time Trials

By Mark Costlow

Some technologies are so enmeshed in the modern world we don't even think of them as technology. One of these is Time. We take for granted that we always know what time it is, or could find out instantly. Watches, tablets, computers, phones, cars, ovens, TVs - everything knows the time, and they all agree to an amazing degree of accuracy.

But the history of computing is littered with **time-related mishaps**. Ironically we have repeatedly built infrastructure to keep and report time without giving due consideration to how long it might last. Thus many of those systems contain literal time bombs that detonate after a certain date.

First, let's look back to a pre-computing time problem, which is really a place problem. This is when we first woke to the fact that knowing WHEN you are can tell you WHERE you are. As we'll see later, that is still true, and more critical than ever.

The Longitude Problem

17th century ocean navigation was risky business. The maps were improving, but of no help if you don't know where you are. To pinpoint yourself you need two things: **longitude and latitude**.

Latitude measures your distance North/South from the equator. One can figure their latitude at sea by measuring an angle with a sextant, consulting charts, and doing some math.

Longitude, your East/West position, is more difficult. There are no landmarks in the sky to hint at your location. But people knew a clever trick:

- Bring an accurate clock, set to the port's time.
- At sea, observe solar noon, and check how far off the time is on the clock.
- Convert the time difference between the ship and home into the East/West distance. Every 4 minutes of time is one degree of longitude.

The problem: accurate clocks used pendulums, which require a solid surface. They do not keep time when being tossed about on the open sea.

This was a well-known unsolved problem for centuries, like cold fusion is today. Countries offered "longitude prizes", starting with Philip II of Spain in 1567. England's is perhaps best known because it produced a result: **John Harrison's** marine chronometers, **culminating in "H4"**, kept accurate time regardless of changes in temperature, pressure, humidity, and the constant motion of the sea.

Y2K and Y2K38

Let's return to the 21st century, where computers tell time with astonishing accuracy, but still occasionally fall on their clock faces.

In the early days of computing, memory was precious. RAM was minuscule. External storage was small and slow, often magnetic tape or punched cards. Programmers strove to keep their programs and data tiny. When storing dates, it was natural to use only 2 digits for the year. Why use 4 bytes to store "1968" when "68" would do?. By the time it becomes a problem, these systems will have been replaced. Right?

Those systems lasted *much* longer than the pioneers guessed. Worse, follow-on systems kept backward compatibility, so for decades the 2-digit date format was propagated into newer systems.

The Y2K Problem is that when the calendar tipped over to the year 2000, computers could misinterpret the data. Workarounds such as "if the year is under 20, it must mean *after 2000*" are fragile and incomplete. For example newborn in 2000 recorded as "00" could appear to be 100 years old.

A monumental effort in the late 1990s fixed software that might make these mistakes, generally by storing 4-digit years. This was wildly successful. The world braced for absolute chaos on New Years Eve 1999, but it was mostly a non-event. A few systems had non-critical problems and were soon patched.

A similar problem looms in 2038. In 1973, the Unix system elected to store time as a "32-bit signed integer", which could count up to **2 Billion seconds starting from Jan 1, 1970**. At the time, this seemed like an eternity, over 68 years! Computers could not even process 32-bit numbers natively, so it was a bold choice for the day.

The 68 year mark is approaching, and that early Unix system has since become the most significant computer system on the planet. The Unix family includes Linux, iOS, and Android, which together power a huge portion of our society. It's also embedded in network routers and switches, and countless Wi-Fi devices such as light bulbs, TVs, thermostats and garage door openers. If they suddenly stop being able to tell time correctly, it would bring the Internet to a halt.

We are starting to see **bug reports regarding "2K38"**. The deadline is 12 years away, but computers often deal with future dates. Some already have trouble expressing futures more than a dozen years on. The consequences of incorrect time are worse now than in 2000. Secure SSL connections (those "HTTPS" URLs) rely on certificates valid only for narrow date ranges. If your computer's time is off by a year, you can't connect to most web sites.

Thankfully people are aware of the problem and fixes are in the works, changing the time value to a "signed 64-bit integer" that will not roll over for 292 billion years. It will take a long time to fix the billions of systems but we are starting with more lead time and understanding than we did with Y2K.

GPS: Where AND When

The Longitude Problem showed that knowing when you are can also tell where you are. That principle is central to **GPS, the Global Positioning System**. There are now several positioning systems, from different countries. We are talking specifically about the US GPS system, the first of its kind.

GPS uses a constellation of satellites, each carrying an extremely precise **atomic clock**. They broadcast their location and the exact time the signal was sent. A GPS receiver compares that time with the moment it receives the signal. Since radio waves travel at the speed of light, that tiny difference becomes a distance measurement. With signals from at least four satellites, the receiver can figure out its latitude, longitude, and altitude.

Time is therefore at the heart of GPS: rather than rulers or radar, it measures the time it takes a signal to travel from a satellite to your device. It only works because the satellite clocks are synchronized to within nanoseconds. This has a side effect that nobody saw coming.

Computer networks need accurate time sources to correlate logs, verify authentication certificates, and schedule tasks. Telecom networks use precise timing for cell-tower hand-offs, call routing, and billing. Engineers looking for sources of "time truth" realized some of the most accurate clocks ever made were broadcasting nanosecond-accuracy time pulses, for free, to every spot on earth. GPS became a common source of extremely accurate time for large networks.

But **GPS has its own "Y2K" style problem**. GPS time starts from January 6, 1980. Its "week number" variable, the number of weeks since 1/6/1980, was stored as a 10-bit number. A 10-bit number has a maximum of 1024, so the "week number" rolls over to 0 every 19.6 years. (Newer GPS devices have a patch to keep them working for 157 years).

GPS receivers have an internal offset to calculate the right week, based on when that product was created. The "**Symmetricom SyncServer S300**" GPS-based network time server was made in 2006 and could give accurate time for 20 years. It was discontinued in 2016, only half-way through that stretch. Such equipment is often used long after the "end-of-life" date, which is fine if it is properly maintained. That meant applying software updates, which Symmetricom provided.

The **Australian national telecom, Telstra**, used the SyncServer 300 to provide authoritative time/date information. However, they did not maintain it, and **the 20-year rollover bug hit them on July 8, 2026.** When the counter reset to 0, the device began to broadcast 2006 as the current year throughout Telstra's network, the largest in Australia. Mobile phones and computer systems were off-line for up to 12 hours. The fix is "simple" when there is a functioning network, but once it has collapsed, you can't even log in to a server to get the patch files needed to fix it, so it took several hours.

An Australian Senate inquiry discovered that Telstra had received notices from Symmetricom about updates for the device in 2022, 2025, and 2026, but failed to act. At the peak of the outage, 45% of all mobile voice and data services in Australia were down. Luckily the 0-0-0 (like our 9-1-1) system used a different clock source and remained up in most places. Large fines have been levied, and one assumes many lawsuits are pending. Companies worldwide are taking note of this wake-up call. **Accurate time is not a convenience. It has become an essential prerequisite** for a functioning network. In our hyper-connected society, it is essential to modern life itself.

Looking to the Future

In modern computing, nobody should create a new system with a limited time storage format. Hopefully the Y2K, Y2K38, and GPS rollover bugs are just growing pains of the information age.

We shouldn't judge the designers of GPS and Unix too harshly for using formats that would "only" work for 20 or 70 years. There were real-world costs to doing something different. For GPS, memory that must survive the harsh environment of space was VERY expensive, and had to be economized. They made engineering vs. cost trade-offs, with no idea their systems would be in such wide use decades after they were made. That's a testament to the lasting quality of those systems, as well as a reminder that we need to make room for adequate time in the future.

This Month in Ideas & Coffee

- **August 12 6pm-7:30pm –** Phish Proof Founders Cybersecurity Event
- **August 18 6pm-7:30pm –** *WordPress Work Along* – Answers to your pressing WordPress questions. Bring your laptop!
- **September 2 6pm-7:30pm** – Karen Arnold from Automattic talks about AI in WordPress.

Watch the **Ideas & Coffee** event calendar at **https://swcp.com/calendar** for future info.