Ricky Eckhardt

Filip Krstevski

GAP 225 OL1

09 Feb 2020

Rising to a Professional Standard: Get to Know Modern C++

The WG21 has adopted a strict three-year release schedule since going through the pains of two long release cycles with C++ 98 and C++ 11. Both of which were nine-year cycles where the releases where held until all feature implementations could be complete. After the pain of the cycles and their intended release dates slipping due to feature creep, the WG21 adopted a strict release date of every three years vice releasing when all features were completed for the specification of that release. Thus far WG21 has released C++14, C++ 17, and C++ 20 on time while maintaining a higher quality across the release.

If a feature is not fully completed and ready for implementation into the next release cycle it is simply pulled from the release. The pulled feature can be corrected and released in the next iteration of the standard at three-year intervals.

Release size on a strict three-year cycle is going to vary due to some major features taking longer to complete. The standard will continuously ship every three years regardless if “feature x” is complete, but when it is be it six to nine years later, it’ll release into that iteration causing a more feature rich “major” release that cycle.

The benefits of the strict three-year release cycle have been numerous. It’s lad to an overall higher standard quality. By metrics it has shown a reduction in defect reports, more work shipped, and a reduction in comments on the review drafts.

The strict release cycle has quickly become the standard going forward. It allows for compilers to ship feature rich to that standard on time which breaks the customers less overall. There is no reason to delay the release of C++ Next going forward for any reason, because feature X will always be around the corner. If WG21 slips for one feature, they will eventually slip for more. No feature is worth not shipping on time, to keep compilers relevant and customers up to date with what works.

The WG21 has clearly made the right choice as shown by the metrics alone. They should absolutely keep to shipping at a steady release cycle versus shipping features. Shipping features leads to an open date of release. That is bad for the specification and language, it slows compiler updates, and keeps working features out of the hands of developers.

To the point of this class, releasing by feature across the industry is simply bad practice, and leads to vaporware. Games with constant feature creep that never release are too numerous to count. Star Citizen is a great example of this. At the same time, the working board, company, or developer must curtail the expectations of the customer. You simply cannot show a video, scene, or whatever claiming feature X and not deliver that result. Blizzards release of Warcraft III Reforged is a great example. Be open and honest in your feature release and the customer will support you for it. It’s better to ship a game missing a feature, but be open about it, and patch it on the back end. Slipping release dates, slip to a undetermined time, and because that other features WILL creep in causing further delay.

Destiny was not fully baked at release. It did disappoint fans to some degree, but it got the game out. Once out, that game was patched to death with features and morphed into the promised game. No Man’s Sky is another game that released not feature rich, but they rode out they wave, and now it’s amazing.

Works Cited

Sutter, Herb. “Draft FAQ: Why Does the C Standard Ship Every Three Years?” *Sutter's Mill*, 14 July 2019, herbsutter.com/2019/07/13/draft-faq-why-does-the-c-standard-ship-every-three-years/.