Mature Software Development - Terminology

We have several mature software products. The terminology around maintenance and development has been ambiguous. This doc attempts to define some terms so that we all know what we're talking about.

Describing Activity

Maintenance - the work required to maintain the current level of functionality, while continuing to protect against changing threats. Usually this is the work of applying updates to upstream libraries, upgrading away from end-of-life versions of software, making decisions about what to do when functionality in software changes.

Bugfixes - correcting the software when a problem has been found that means that the software does not work as designed, or to correct a design decision that doesn't achieve the desired functionality. For example, this might be correcting a typo, or fixing a function that doesn't do what it says.

Hosting - for services that deliver functionality over the Web (as opposed to software that's downloaded and run on someone's own computer), hosting is the provision of server space, and the work to assure the regular availability of that server space - such as monitoring, performing routine software upgrades, and responding to incidents.

Incremental Improvements - changes to the software that enhance its functionality that don't require significant changes to the general approach of delivering the functionality of the software. For example, this might be adding a new data check or type of export, adding a new type of functionality to an existing tool, or changing the presentation of some results.

Iterative Improvements - larger pieces of work to make a step-change to a product, that may change the general approach to delivering the functionality of the software. For example - a redesign of an interface, a behind-the-scenes reworking of functionality to be faster or handle more data, the splitting up or combining of existing products to better serve users' needs.

Describing expectations of software

When changes to software are requested, there is a trade-off to be made between how far to follow the good practices that keep mature software stable and maintainable over the long term and how quickly new features can be tried out.

Any particular piece of software might have elements that are at different stages of development. For example, a new feature in a mature software product might be a prototype, while the main software product is production-ready.

This terminology provides a shared language around requesting changes to software, so that clients and developers alike can be clear on what is expected when a change is requested.

Production - This is the default. Code written to Production standard is ready to be put into live use, and requires no further work before we can deploy it and leave it in place indefinitely

Production software will have automated tests, and we'll have ensured that it can handle edge cases and bad input. There will be documentation of the features, and the code will be structured in a way that future developers can maintain it.

Note: Not all software that is in production use is actually production standard due to technical debt, historic practices & project limitations.

Prototype - A Prototype is something that that's developed with the minimal effort required to try something out; maybe to see if something is possible, to better understand what it would look like or how users might respond to it, or to try out a new technique. It could be a new product, or a new feature within a mature product. It probably won't have tests, or be necessarily implemented in a good way. It provides a quick way to see if something works, and shortens the path to Production code considerably, but there's often considerable work to be done to take a feature from Prototype to Production.