Thermo Fisher Scientific: Lesson #8 – Quality Assurance Specialists Do Not Test Most Story Deliverables

One of the predictions from my Agile Hardware research was that the traditional software Quality Assurance model would not transfer well to hardware development.

The Thermo Fisher Scientific engagement validated that prediction.

In software development, there is a long-standing practice of maintaining dedicated Quality Assurance personnel whose primary responsibility is testing the product.

The reasoning is straightforward.

Software developers focus primarily on designing and implementing functionality. Quality Assurance specialists focus on verifying that the functionality behaves correctly under a wide range of conditions. Although developers certainly perform testing, Quality Assurance is a specialized discipline with its own methods, tools, and expertise.

This arrangement works particularly well because software Stories typically produce observable product behavior. A Quality Assurance specialist can interact with the application through its user interface and verify that the implemented functionality behaves as expected.

The situation at Thermo Fisher was very different.

As discussed in previous articles, the team's work was dominated by Technical Stories rather than User Stories. The deliverables produced by these Stories were often components, subsystems, prototypes, schematics, interfaces, test fixtures, and engineering designs.

Most of these deliverables could not be evaluated through a user interface because there was no user interface.

More importantly, testing frequently required deep technical knowledge of the specific component being developed.

A circuit design might require electrical expertise.

An optical subsystem might require optical expertise.

A mechanical component might require mechanical expertise.

There was no single Quality Assurance discipline capable of independently testing all of these deliverables.

In many cases, the person best qualified to test a Story's deliverable was the engineer who created it.

This does not mean that hardware teams ignore quality or verification. Quite the opposite.

Hardware organizations typically perform extensive verification, validation, reviews, inspections, integration testing, and system-level testing. However, these activities occur differently than they do in software organizations.

What was missing at Thermo Fisher was the familiar software pattern in which a dedicated Quality Assurance specialist independently tests the output of most Stories.

There were some exceptions. Documentation, schematics, and other engineering artifacts could often be reviewed by peers. Nevertheless, the dominant pattern remained the same: testing responsibility was closely coupled to the engineering expertise required to produce the deliverable.

This led to another important lesson.

Lesson #8: Hardware organizations generally do not have a direct equivalent to the software Quality Assurance specialist who independently tests most Story deliverables. In many cases, the engineer who creates the deliverable is also the person best qualified to test it.

This observation follows naturally from several of the previous lessons. When Stories primarily describe technical deliverables rather than user-facing functionality, testing becomes a specialized engineering activity rather than a separate Quality Assurance function.

In the next article, I will discuss another software practice that proved difficult to apply directly to hardware development: estimating work using Story Points.

For those interested in the full case study, the complete Thermo Fisher Scientific story is available on my website at https://kevinthompsonphd.com/storage/papers/LessonsLearned-AgileHardware-v6.pdf.