Carlo Gavazzi Controls Black Belt Case Study

1 Introduction

This document illustrates the EDT black belt improvement project by the **UWP product team at Carlo Gavazzi Controls**. Their black belt certification is based on their continuous improvement journey, focused on achieving a more efficient and faster way to deliver better quality value to their final user.

The journey here is guided by the rigorous systematic engineering approach to continuous improvements as described in **Engineering the Digital Transformation** by Gary Gruver. This process guided us in our transformation from a standard waterfall (v-model) to a **continuous delivery** approach. The results represent the contributions from everyone on the team with special recognition to Alessandro Fardin, who led the effort. We would also like to thank the leadership team of Carlo Gavazzi Controls for letting us share this journey with the broader community in the hopes it will help inspire others to start their continuous improvement journey.

This story is somewhat unique in that it is focused on embedded systems. Too many times embedded SW teams think they are unique and that good SW engineering approaches could not be applied. This example shows good engineering approaches like **continuous delivery** can and should be applied to every SW/FW context, embedded included, because it is the most efficient way to build in quality.

2 Application Description

1 UWP System

UWP is a system that integrates **Energy Management/Efficiency** and **Building Automation** applications. UWP business targets are Industries and Big buildings.

2 UWP components:

UWP is composed by 3 main components:

- UWP controller: (embedded device)
- UWP WebApp: WEB Application embedded in the controller
- UWP IDE configuration SW (Windows application)

3 The Journey

1 The need for change

In 2018, we were dealing with many difficulties: We had serious quality problems that made our final users unhappy. We were often late with our deliveries. We spent time fixing defects caused in previous releases instead of developing new features. The teams and I needed a way to be happy and proud of our work and our product. We needed a change in our approach.

We heard about DevOps and continuous delivery, and discovered Gary Gruver's approach, prompting us to start on our continuous improvement journey.

Carlo Gavazzi Controls Black Belt Case Study

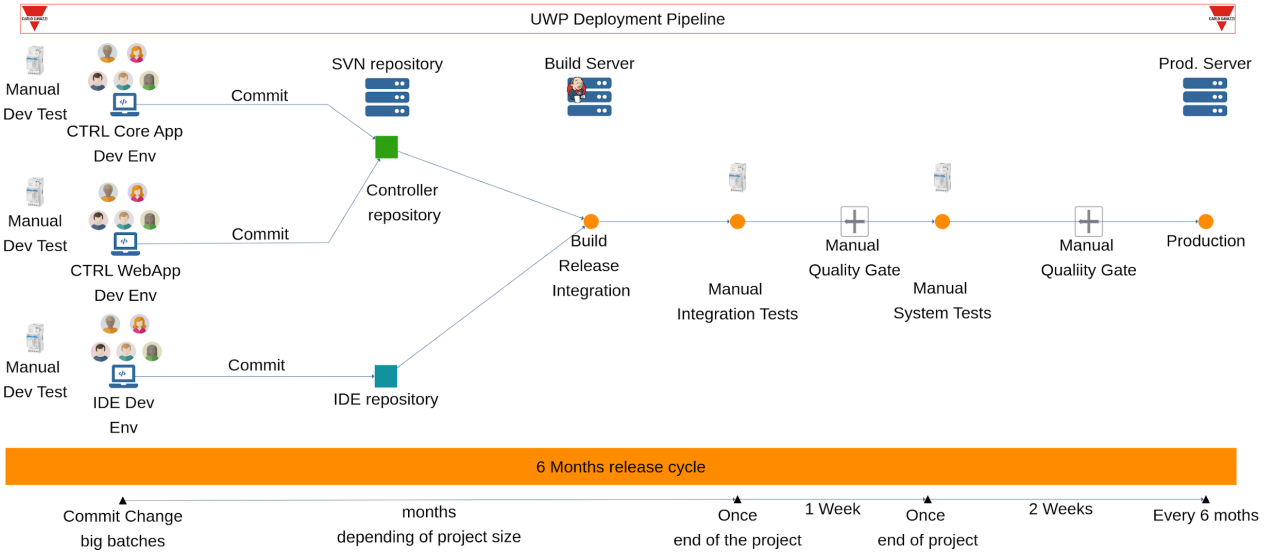
2018: The Starting Point

Back in 2018, we had a v-model waterfall developmental approach, with **6 month release cycles:**

- UWP quality strategy was based totally on manual tests.
- Large separate phases for development and inspecting quality
- Separate teams for development and testing

Our process as shown in the UWP Deployment pipeline graphic below had been based on developing large features called projects, so we used to follow the standard SDLC approach, but with slow and poor feedback loops.

- Several months in the develop phase (new features and bug fixing)
- 1 week of integration tests phase (dev inspecting for quality) 100% dev team
- 2 weeks of system test phase (test team inspecting for quality) 100% test team



Because quality was inspected in the last phase, when the software was handed over from developers to testers, we found that there were a lot of defects and deviations from what the business actually expected. We worked hard to fix them, but there were too many to fix in order to finish on time, so a lot of the defects were then discovered in production. This led to a huge bug fixing activity that had to be done at the same time as the next project.

As stability and quality decreased upon every new release, we found that we spent 60% of our development time in bug fixing activities. As our branch strategy was feature branches, we needed at some point to merge features and bug fixing onto the release branch. This merge activity cost us 10% of time and resources.

State of Development Before the Transformation:

- 60% of capacity on bug fixing
- 10% of capacity to merge from feature branches
- 30% of capacity for new features
- the number of bugs increases at every release
- about 200 bugs had accumulated
- 75% defects were discovered in production
- 6 months between deployments

Carlo Gavazzi Controls Black Belt Case Study

2019: Start of the Journey

In mid-2018, we merged Energy Management and Building Automation applications into a single platform. But because of the large amounts of accumulated bugs, we had no choice but to apply a feature freeze for some months to allow time for fixing the majority of the instability problems.

In 2019, we started adopting DevOps, and, as suggested by Gary, we found ways to reduce waste in our pipeline. The first step was to reduce batch size and utilize user-stories to make small features testable. So we removed the integration test phase, and integrated a continuous test approach during the development phase. A one-week system test phase stood in place to filter late integration defects. Tests were still manual and slow but because the defects were found earlier we had time to fix them at a sustainable pace. This way we were able to introduce no or very few defects in new releases. The number of bugs also constantly decreased. We then took the step to reduce the release cycle to 3 months. This resulted in reduced cycle time, which permitted us to focus less on bugs and more on building in quality.

2019: Our Effort to Reduce Waste Delivers Results

- 50% of capacity fixing bugs
- 10% of capacity to merge from feature branches
- 40% o capacity for new features
- the number of bugs decrease at every release
- 100 known bugs still to fix
- 30% defects were discovered in production
- 3 months between deployments

2020: The Right Direction

In 2020, we disbanded the system test team and merged the test people into the Dev team, to form a cross-functional team where people develop features and write automation tests. We moved from SVN to GIT and more importantly to **Trunk Base Development** using feature flags. This enabled us to develop and deploy releases from the same branch (the trunk), removing the need to merge, which saved us another 10% of resources that were now free for developing features. We started using some of our automation tests. We started using Kanban board to better visualize our value stream. We took the step to reduce the release cycle to 1 month.

2020: Our Effort to Reduce Waste Continues to Deliver Results

- 45% of capacity bug fixing
- 0% of capacity to merge from feature branches
- 55% of capacity for new features
- Number of bugs decrease at every release
- 0 known old bugs to fix
- 10% of defects discovered in production
- 1 month between deployments

Carlo Gavazzi Controls Black Belt Case Study

2021: Wow, It Works!

In 2020, we reached the minimum cycle time that our business could accept. In 2021, we worked to amplify our feedback loops, in order to remove more waste and reduce defects. We started using executable specification by building a 4-layer automatic acceptance test suite, built on top of python/behave/gherkin based on David Farley's work. Tests were small and binded to behavior (not to implementation) in order to make it maintainable and sustainable. Devs started to write automated tests as soon as small stories were concluded. We used the suite as a quality gate. We used an early version of an automated pipeline with static code analysis, some unit tests, and acceptance tests.

Our first acceptance test suite was slow and brittle, and as the number of tests rose, they became flaky and took nearly 14 hours to run. There was still waste that needed identifying and removing:

2021: Our Effort to Reduce Waste Delivers More Results

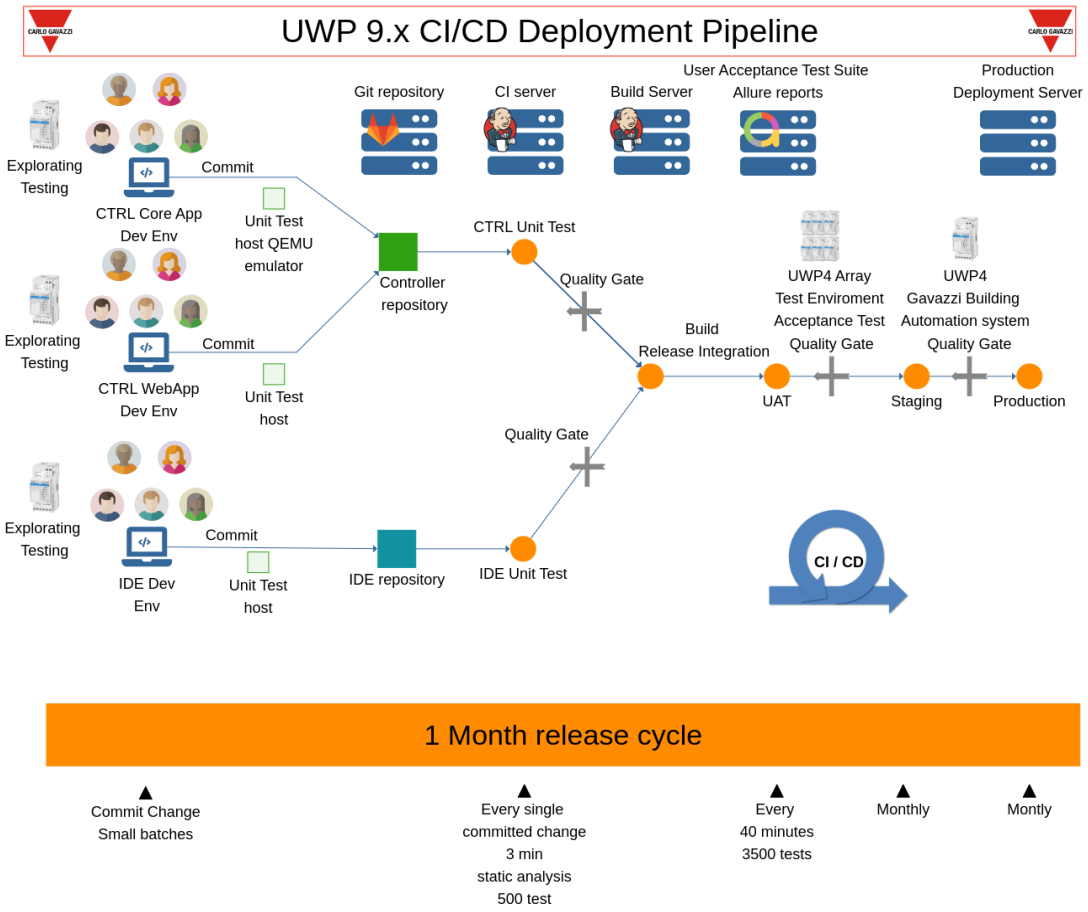
- 40% of capacity bug fixing
- 0% of capacity merging from feature branches
- 60% of capacity for new features
- number of bugs decrease at every release
- 0 known old bugs to fix
- 10% of defects discovered in production
- 1 month between deployments

2022: The Stable Quality Signal

Having gotten my EDT white belt certification, I've understood the importance of having a stable quality signal. As our suite became flaky and slow. We worked to make it stable, running tests at the same time to make it even faster, and we went from 14 hours down to only 40 minutes to run 3500 tests. With such fast feedback, we started improving test stability passing from a 70% success rate to a 99% success rate. We worked on daily bases with priority to fix flaky and failed tests. We introduced a fully automated pipeline that runs at every commit, and in 40 minutes can give us the results. We've introduced a monitoring system that continuously shows us the status of our embedded system. It enabled us to find and fix problems like memory leaks, and high memory usage in our environment.

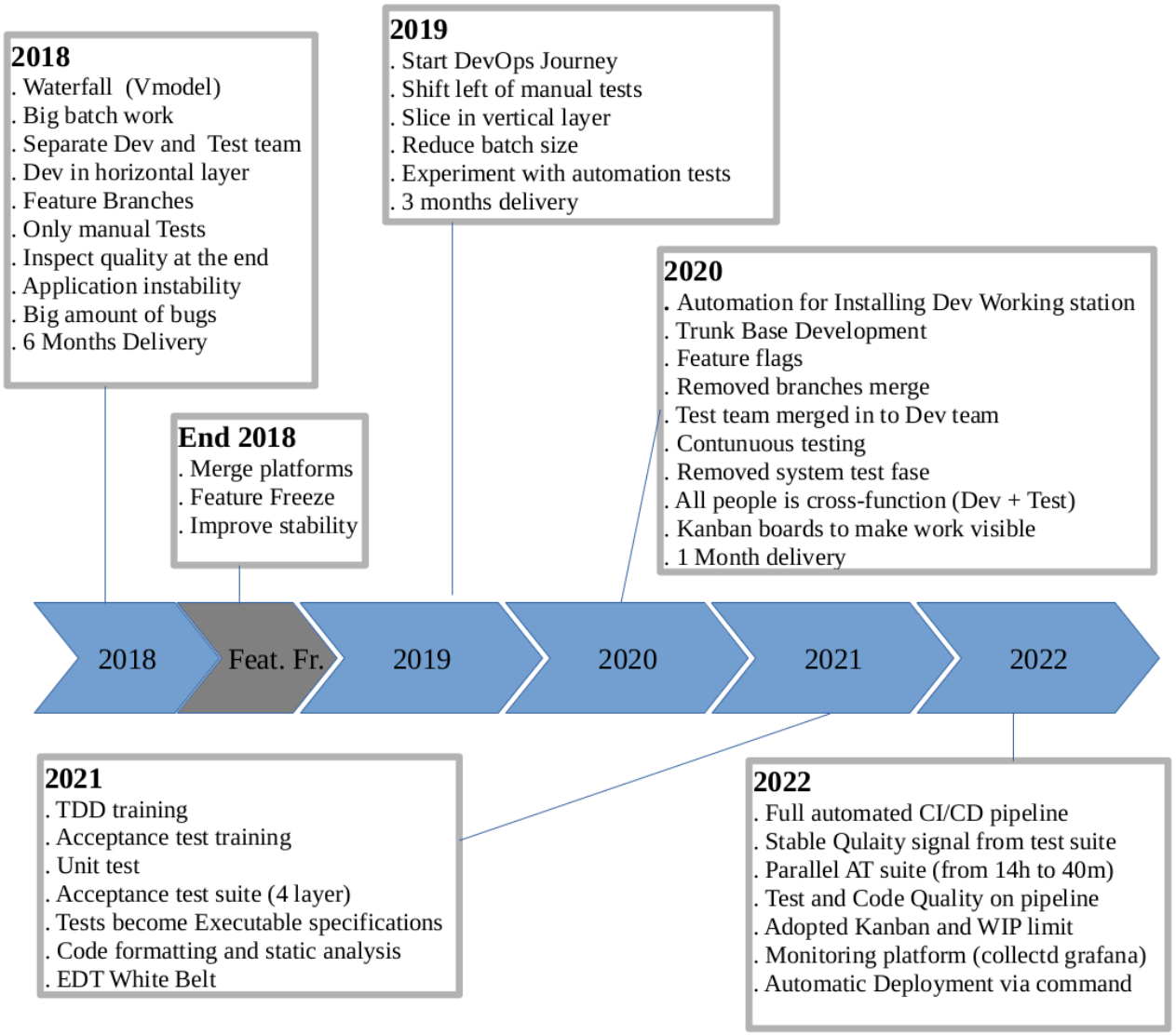
Carlo Gavazzi Controls Black Belt Case Study

The end result was a completely different Deployment Pipeline as shown in the graphic below:



Carlo Gavazzi Controls Black Belt Case Study

Our transformation journey took perseverance and several years as you can see summarized in the graphic below:



By the end of 2022, we had completely transformed how we developed software, resulting in the improvements listed below:

- 30% of capacity bug fixing
- 0% of capacity merging from feature branches
- 70% of capacity for new features
- All found bugs are fixed up to the next release
- 0 known old bugs to fix
- 2% defects discovered in production
- 1 month between deployments

Carlo Gavazzi Controls Black Belt Case Study

4 Conclusion

Applying the EDT continuous improvement approach, we have successfully removed tons of waste that had been slowing down our process. Applying a continuous delivery approach with focus on improving stability and building in quality, allowed our current releases to be in a releasable state nearly every single day. **This transformation has had a tremendous and positive impact on the business, enabling us to more than double the capacity for delivering new capabilities to the business (30% to 70%) and dramatically improve the quality for our customers (75% to 2%).** Also significant, applying the EDT continuous improvement approach created a better work environment for the team, where everyone felt like they could be successful.

Removed wastes:

	2022	2018	Removed waste
Time spent on new features	70%	30%	40%
Time spent on bug fixing	30%	60%	30%
Time spent on merging branches	0%	10%	10%
Known bugs from old releases	0	200	200
Bugs trend	Stable 0	Constantly increasing	No stability degradation
Defects discovered in production	2%	75%	73%
Feedback cycle time	40 minutes	Some months	Some months
Deploy frequency	1 month	6 months	5 months