Navigating Mobile Robotics Development & Delivery using Agile at a Global Mobile Robotics & Automation Machinery Manufacturer

KANIKA SHARMA, SCRUM MASTER

This report is based on my experience as a scrum master serving three teams at a Global Mobile Robotics and Automation machinery manufacturer. As someone new to this domain, I faced some new challenges which made me experiment with different flavors of agile and scrum. Before working in this domain, I did not understand the magnitude of the challenges while making changes in a product like an Automated Guided Vehicle (AGV). After this experience, I have more confidence in working with complex systems involving software running on hardware. I share my learnings in this report with anyone who is trying to adopt agile and find a good way of working for niche products like mobile robotics and AGV systems.

1. INTRODUCTION

Automation and robotics are no longer a dream but a reality. This experience report is set in the context of mobile robotics used to automate processes in logistics in various industries. While working for this Global Mobile Robotics and Automation machinery manufacturer as a Scrum master, I got first-hand experience with what the future can look like! Mobile robotics provide precision and control, are free from human errors, and are safe, especially when discussing dangerous working environments. These mobile robots ranged in size depending on the function and were designed to accurately maneuver or carry heavy pallets with minimum human interaction while delivering high performance. Before the events described in this experience, the company was already delivering AGVs, so what changed when we applied agile methodologies to our development? In this experience report, I would like to focus on the challenges I faced with a unique product, the dynamics that revolved around the teams and departments, and the lessons we learned along the way. I will also share recommendations based on my experience.

2. BACKGROUND

While I believe in Agile Principles and have been a follower of this way of working for more than a decade now, I have also learned that in practice, every industry, company, and organization adopts and scales differently. Experiencing product development in mobile robotics has been challenging and enriching at the same time. While the product has electrical and mechanical parts with their own procurement, assembly, and testing phase, software plays a major role in controlling these automated guided vehicles' functions. On top comes the non-negotiable safety requirements which cannot be ignored at any cost. After all, these automated guided vehicles will be handling warehouses with minimum human intervention and should hence be at the highest safety level for the humans they interact with. Before working in mobile robotics, my experience was predominantly in software and app development. It was thus a new environment for me to apply my learnings of agile and adapt based on the need of my organization, product, and most importantly, my teams. While developing software and apps, you can always engage customers from a very early stage of development. It is easier to demonstrate what your product might look like and collect early feedback. In products like mobile robotics, we had to rely on prototypes and simulation environments, but the real feedback was received only when the customer acquired a tangible product. Only after experiencing the complexities of AGVs (automated guided vehicles) and mobile robotics and how any small change led to a series of finetuning between the hardware and software components and the immense testing that followed, I realized and could appreciate the ease of making changes to the software.

Author's address: Kanika Sharma, Germany; email: kanikasharma17@gmail.com

3. CHALLENGES

Our products are Automated Guided Vehicles (AGVs) which are specialized for material handling catering to automotive, healthcare, manufacturing, and intra-logistics industries. Based on where these AGVs are deployed, the size and configurations vary. For example, the AGV developed for light industrial and healthcare applications where space is limited, offers best-in-class maneuverability. Whereas the AGV developed for automotive industries was designed to lift heavy loads of automotive parts and pallets. The supervisory software that controlled the AGV and guided it around the layouts was the common element for all vehicles. The intricate interaction between the vehicle software with the supervisory software makes these vehicles autonomous.

With the fast-developing markets, our product was facing heavy competition. There were companies out there that were able to produce more compact mobile robotics and gave us a tough fight especially because of our long delivery timelines. It was hence decided to adopt agile, in a zest to deliver faster to market and cater to our customer feedback promptly.

The challenge of adopting agile was related directly to the complexity of our product and aligning the different groups that make the final product deliverable. We have procurement teams, electrical engineers, mechanical engineers, system engineers, software engineers, product test engineers, and customer support engineers, all of whom had expertise and knowledge of the product based on their area of work. So, before the product could even be ready for software interaction, there were plenty of checkpoints in place in the hardware procurement and assembly life cycle which followed a more or less waterfall model of development.

To enable agile delivery of our product, it was key to identify how to apply agile development to most of these facets of Product Development and by including other departments that directly impacted our delivery timelines. Here are a few of the many challenges I faced:

3.1 Agile or Not to Agile

Our Automated Guided Vehicles had various departments involved before they could be finally delivered to our customers. Between the procurement of components to the final deployment of the AGV at the customer, various expert groups ensured that the product was built right. The software department took the lead in adopting agile as their way of working. We had a department head with a vision to transform software delivery to agile and hired Scrum masters to ensure that the teams received coaching and guidance through this transformation. On the other hand, we also had departments that were questioning this way of working. System Engineering and Project Management departments believed in planning to the minute details and then executing these plans. As a scrum master, my role was not only restricted to helping my software department teams but also engaging actively with System Engineers, Project Managers, and Engineers working on the AGVs to enable the delivery of the product. For the products my teams were developing, I initiated chats with System Engineers who were mainly interacting with customers and visiting customer sites before and during the development. By sharing the challenges my software teams were facing, I was able to convince a few System Engineers to interact more with the developers. The biggest challenge was to convince all these people to try the agile way of working and of course, not everyone was on board even after workshops and coaching sessions that were provided to them.

3.2 Difficult Stakeholders

Project Management was the heart of the organization and was the recognized working model in the past. Although the shift towards an agile way of working brought ownership and control to the software developers, the project managers felt insecure about being unable to have direct control over teams. This caused friction sometimes, as they believed being agile meant that the teams should be open to changes all the time and that all new work should somehow fit in the sprint backlog. Prioritization was not done, and everything was priority number one. This took away a lot of focus from development teams and left them in constant troubleshooting mode.

3.3 Changes in Layout of the Factory/warehouse

The final playfields for our automated guided vehicles (AGVs) are huge warehouses or factory floors that are automating their manufacturing floors. Based on the layout, the Supervisory software was designed for these mobile robots to perform the required functions and drive to charging stations when the battery reached the lower threshold. This software was designed and optimized to the originally agreed layout that was approved by our customers. Many times, these layouts were later modified to adjust the floorplan and were required to be re-optimized both on hardware and software components but were either not communicated or informed at a very late stage to our development teams. This led to unnecessary delays as every little change added re-optimization and series of testing (from testing in simulation to test floors to System tests both in-house and at the customer site) to match the new floor plan which in return frustrated our customers (due to delays) and the developers due to finetuning and testing overheads.

3.4 Supporting Customer Issues and Customer Support Engineers

Customer Support engineers did their best to resolve customer issues, but we had a lot of these issues that landed on the plate of software development teams, especially if it was related to the Supervisory software. This meant, that every sprint we had to dedicate some focus towards these customer issues and depending on the criticality, even push them to priority 1. This was a constant distraction to the software teams but crucial to stay in business. It was hence vital for me to create a workable process to bring these issues to the teams promptly and help them get into our planning.

4. FROM CHALLENGES TO COLLECTIVE LEARNING

4.1 Developing a common goal

Although departments were working differently than the software department, what helped us work through our differences was to identify and develop a common goal: Building and deploying a high-quality AGV that is efficient and secure. We hold great pride in our AGVs. All departments were informed by the Software department Head about the agile way of working that the software department was adopting, and he encouraged others to experiment with it. Scrum Masters were requested to collect topics to plan workshops and anyone who needed coaching was provided support. In a Lean Coffee session that was initiated, many System Engineers had questions about how their role could work in the scrum structure. Support Engineers were curious about how to handle customer issues that popped up during the 2-week sprint. After collecting all questions, the topics were grouped, and we started coaching other departments as well. Some departments like procurement continued to follow what worked for them. Procurement had a process of ordering electrical and mechanical parts like sensors etc. which were then tested before they were used to build the AGV. For them, this method meant fewer surprises that the parts were faulty. Hence, they decided to opt out of trying any new ways. I identified the System Engineers and Project Managers that were directly working on AGVs that my teams were developing as our immediate stakeholders and started inviting them to our Reviews and Demos. These were the people talking to our customers to bring requirements and changes in requirements and hence it was crucial to engage them with the teams. Some Project Managers were reluctant to join reviews and simply did not show up and wanted updates every few days. I prepared Sprint Reports (refer to Figure 1 in section 7.1) that were sent out to key stakeholders and team members to reduce one on one updates and to create transparency. Sprint Report started with the Sprint Goal, steered to the Burndown chart for that Sprint, and included stories that were Completed and those which were not Completed. It also contained the list of customer issues that were resolved in that Sprint.

4.2 Managing Stakeholders

The differences between software teams and Project Managers caused occasional friction sometimes leading to developers being frustrated especially when changes were brought in right after Planning the sprint. These tensions were crucial to resolve and to strike a good balance between the two groups. I organized workshops for Project management to help them understand the way of working and help them bring customer wishes by prioritizing. Teaching them the value of prioritization to place these ''wishes'' in the Product backlog in agreement with the Product Owner proved to be key to improving relationships between the developers and Project Managers. For Project managers, it was important to be able to forecast product delivery timelines to customers. With the undergoing transformation, not all teams had high predictability. We tried to find similar scoped projects and for the simplicity of understanding use T-shirt sizing to help with forecasting. For example, a small project was considered as one vehicle driving on 4-5 routes. A medium complex project would be with 2-3 vehicles with routes to be defined based on the layout of the factory. It proved to work for small and small-medium projects but was not useful for large projects with more complexity. For example, one of the automotive giants in Europe was our client but they had different types of mobile robots deployed in their floor plan. The factory also had many humans working who were not trained yet on how to operate and work around our AGVs. All these factors tied together and increased the complexity of the calculations the different AGVs had to do at run-time while running on the designed routes.

4.3 Bringing People Closer

The vehicle software was developed in a separate building that had layout designs to test our AGVs. The Supervisory software teams were located close by but in a different building. As the first step, I made sure to collect information and create a holistic overview for myself. Parallelly, I started making 'lunch walks' to spend time with different groups to obtain diverse viewpoints. If there was a possibility, I tried to take the discussions to the coffee machine to create a more informal environment to exchange information. What helped by bringing these people closer was this environment of constant sharing of news. If a System Engineer visited a customer, they tried to take questions from the software and hardware teams to the customer and vice versa. If the System Engineer had questions for software teams, they were encouraged to always reach out, inviting them to sit with the teams and work closely together. Due to this high engagement between departments, a lot of layout changes were communicated earlier. We still had situations where these layouts changed later, but it was reduced by 75-80%.

4.4 Resolving Customer Issues

In collaboration with Customer Support Manager, we came up with a process where all 'customer issues' were prioritized by the support department. Software teams blocked 20% of their Sprint capacity to resolve these issues. The issues were detailed, and a support engineer was available if software teams had further questions. This practice helped software teams a lot as customer issues were now more structured and they had enough focus to work on other 'planned' items in the sprint. It also strengthened our customer relations as our SLAs were now better defined and all medium-priority customer issues were tackled within days instead of weeks which was 20% faster than before. High-priority customer issues were agreed to take precedence over ''all other work'' for which we followed the simple process- Any customer issue entering the backlog cost a low-priority work item to be pushed out to the next Sprint.

4.5 Problems at the Customer Site

There were issues reported at customer sites after deploying our mobile robots. 7 out of 10 times, it was due to a safety feature or a sensor getting triggered due to an object on the way. Some of our customers who are transitioning towards automating their site, also had people working on the same floor as our AGVs. If these people were not trained on how to behave around the AGVs, they would unknowingly trigger the safety stop feature which caused performance to drop, sometimes up to 75-80% for that loop. Additionally, some layouts needed finetuning during the System Acceptance Tests which needed knowledgeable engineers at the site during and after the deployment of AGVs.

5. RESULTS

The attempt to convince all departments to adopt agile practices was not achieved. The Scrum Masters were hired only for the Software Department and were shorthanded to coach all departments. Nevertheless, I am proud to say we did make significant improvements, especially with the departments that closely interact with the Software department which helped us improve in various ways. Examples include the following:

- Achieved improved transparency both within the teams and with stakeholders by almost 95% as indicated by an internal survey which was conducted per project. The Project Managers were satisfied receiving updates and information via the Sprint Reports and some of them also started attending the demos regularly.
- Accomplished doing more planned work and less firefighting by reserving 20% Sprint capacity for Customer issues.
- Due to improved collaboration between System Engineers and Software developers, the number of layout changes made to floors of the factories where our AGV's routes were calculated was reduced by almost

80% compared to the number of layout changes made before within a timeframe of 6-8 months. Including vehicle and supervisory software engineers for the layout, designs proved to be phenomenal in our case.

6. LESSONS LEARNT

Here are some valuable learnings from my experience and some of them can apply to teams undergoing agile transformation, especially in complex systems like mobile robotics.

6.1 Stakeholder Mapping

It was my first experience working in the mobile robotics domain, so before jumping in with solutions, it was important to gather a holistic overview of existing processes. I also saw it crucial to collect diverse viewpoints to address what was needed first. My very first focus was to identify my stakeholders. I was working with some very smart developers who were trying to solve complex customer problems. Out of the three teams I worked closely with, two were always in fire-fighting mode, trying to solve issues that were found at the customer or trying to help another site engineer who was at the customer, deploying an AGV. I made it my mission to understand the business and started making 'lunch walks' with colleagues where we could informally discuss the challenges. I made sure to 'hang out' or start a conversation at the coffee machine with different people. This helped me not only understand the challenges of my teams better but also helped me recognize who or what could help make the situation better. My teams and the people impacting my teams like the System Engineers, Customer-support engineers, and Project Managers were on my Stakeholder map.

In the beginning, the developers saw me as 'another layer in management' but I could slowly earn the trust of my teams when they realized I was trying to make their life easier.

On multiple occasions, I helped the Customer Support manager to prioritize customer issues by asking questions like impact, what will happen if we do this in a week, is the customer blocked or is it just nice to have functionality? This helped remove some of the distractions my teams were constantly facing.

6.2 Improve Collaboration

As mentioned earlier, the vehicle software teams were mainly working in the layout and testing area in a different building. The supervisor software teams were in a different building nearby. We had stand-ups separately for both teams and found the software team blocked by the vehicle team or vice versa. To solve this situation, we agreed with the teams to have a stand-up (like a scrum of scrums) together twice a week to let developers interact with each other directly. On other days, the scrum masters of both teams interacted regularly to improve collaboration. This helped both teams proceed at a good pace while maintaining high interactions. We also managed to establish good relationships with System Engineers, Project Managers and teamwork with Customer Support engineers.

6.3 Establish Information Flow

Whether it is the customer, product managers, stakeholders, or the teams, make sure that important information flows top-down and bottom-up always. At our company, we had Roadmap sessions, where Product Managers shared the short-term and long-term vision and set expectations for the teams. The developers know the capabilities and inhibitions of our supervisory software. The earlier they get information about new projects and potential AGVs that are required to be deployed, the faster they can access the feasibility of the system. For example, one of the routes defined by the customer had a single charging station and our developers pointed out that a second charging station was inevitable for the efficiency of the system, without which the vehicle would not be able to complete loops to the charging station and would have to be manually brought to charging station. Such an early realization was made possible by having a good flow of information between different departments.

6.4 Focus on value

Unlike software applications, to demo a product like our AGVs needs a sign-in from many. In the zest of creating a feedback loop with our customers, we created simulation environments where different navigation loops could be set and tested multiple times. In the beginning, the demos were organized per team which meant that to get an overview of the project, the stakeholders had to attend 2-3 Demos which was not viable always. Based on the feedback, I reorganized our demos per product and all the teams working on that AGV were invited to showcase their progress. Just a small change, brought multifold value as we had all people, be it, Supervisory software engineers, vehicle software engineers, System Engineers, or Project Managers involved and engaged in what was happening. We improved our response time on open questions by almost 50%.

6.5 Deployment and training at the customer site

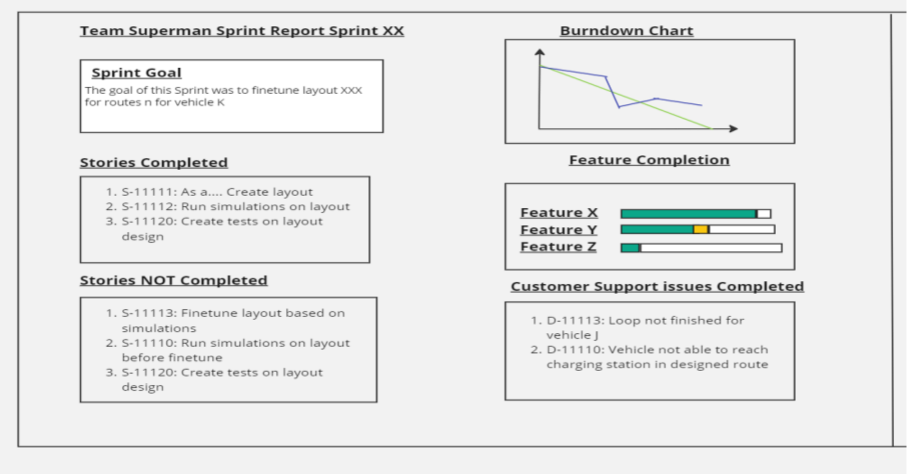
There were substantial issues reported immediately after the deployment of AGVs at the customer site. As mentioned before, 7 out of 10 times these issues were caused by interference from people working on the floor. But before reaching this conclusion, the development teams would spend effort and time trying to analyze the root cause. Once we found out this was a common situation, we also proposed training not only to the users of the AGVs but also to the people around the layout floor to reduce these human-induced errors.

7. RECOMMENDATIONS

Here is a list of recommendations for you to check out, based on my experience on my journey of agile transformation in mobile robotics.

7.1 Power of small changes

It became clear to me that the small changes were easily accepted and seen as low risk by groups. In my case, for example, Project Management was one difficult department to agree to anything. They were reluctant to join demos, and still always demanded updates. Sharing Sprint Reports every Sprint was a small change that brought 100% more transparency to all those involved with the product. Based on feedback received from stakeholders, I also adjusted the Sprint Report to make it more useful and informative. Changing the format of the demo to give Project Managers product-level updates instead of team-level updates at the demo eventually got me a buy-in. Similarly, with the teams, any minor change in the way of working was accepted easily.



Team Superman Sprint Report Sprint XX

Sprint Goal

The goal of this Sprint was to finetune layout XXX for routes n for vehicle K

Stories Completed

1. S-11111: As a.... Create layout
2. S-11112: Run simulations on layout
3. S-11120: Create tests on layout design

Stories NOT Completed

1. S-11113: Finetune layout based on simulations
2. S-11110: Run simulations on layout before finetune
3. S-11120: Create tests on layout design

Burndown Chart

Feature Completion

Feature X

Feature Y

Feature Z

Customer Support issues Completed

1. D-11113: Loop not finished for vehicle J
2. D-11110: Vehicle not able to reach charging station in designed route

Figure 1. **Example Sprint Report**

7.2 Adapt and constantly change based on what works

As a scrum master and coach, the biggest lesson for me is that I need to be flexible in providing recommendations and ways of working depending on the organization and teams I work with. There are no tailor-made one size fits all kind of solutions. The Scrum of Scrums between the vehicle software and supervisory software proved to be effective as we found that both teams were unblocking each other at least 50% faster. But once we achieved a steady state in that project, the frequency was reduced from twice a week to once a week and then stopped. The scrum masters continued to align and bring impediments to relevant teams. For example, if the vehicle software team had a dependent story on the supervisory software team, the scrum masters would align and make sure that the teams are unblocked within a day or two.

7.3 Experience at the Customer site

Most of our engineers got the opportunity to work at the customer site. This helped them understand the pain points of the customer. For example, if our AGV sensors detect an object on its path, it is supposed to stop, abiding by the security feature. However, there were occasions when there was material or packaging that was on the route that led to the unnecessary stopping of the vehicle and creating queues. Training the factory workers helped solve this problem easily. The occurrence of such situations was reduced by 85% by educating the factory workers.

8. CONCLUSION

It took us 8-10 months and relentless efforts to move to a workable, agile state within the software department. Our teams could comprehend customer requirements better and react faster to changes made in the layout of the factories and warehouses. Our customers were deploying our products faster and were happy with the fast response to feedback from our Software teams. Our existing customers rated us higher in the satisfaction score as their issues were resolved faster and well within the agreed SLAs. We made direct and constant contact with our customers or their representatives, which helped us build strong relationships with them. For large-sized projects, with lots of interdependencies between vehicle software and supervisory software, we even considered trying SAFe but decided to pause to move to that stage because of changes in the structure of the company and new leadership. It was a conscious choice to keep delivering our AGVs with our earned learnings so far.

9. ACKNOWLEDGEMENTS

I would like to thank my colleagues and teammates who helped me grow and learn on my Agile and SAFe journey. I big shoutout to those who were ready to take the challenge with me and deep gratitude to those who challenged the process and kept me on my toes. I am grateful to my Shephard, **Ken Power** who helped me shape my thoughts better and pushed me to share more with the readers. I would also like to thank my mentors and coaches who invested time and constantly shared their knowledge with me.