Scrum 911! Using Scrum to Overhaul a Support Organization

Bhaven Sheth
bhaven_sheth@mckinsey.com

Abstract

The Support 2.0 Team at our IT organization has recently completed Phase 1 of the Support 2.0 project. Phase 1 was a four month endeavor with a primary focus of collecting data to understand why support was so costly and identify the root cause for each support request. The team strongly believed that this initial phase would be critical to making significant impacts to the quality and supportability of applications.

This experience report describes the challenge for the Support 2.0 Team and shares the successful adoption of agile practices in redefining the support organization. In the spirit of the agile transition to software development one year prior, our IT organization decided to rethink the approach to application support. By embracing agile principles and utilizing agile concepts, such as a cross-functional team, a team collaboration approach with frequent reviews to inspect and adapt as needed, etc, the business was able to reap immediate benefits.

It was a challenging journey for the team, where the mix of onshore/offshore support engineers and an integrated client / vendor relationship within an agile team and focused framework helped to transform the support organization from an operational cost center to a value added thought partner. This resulted in a significantly improved culture and effectiveness of the support organization. This unique organization of an agile, cross-functional and team-based approach to handle support requests improved the customer experience, reduced support costs, and ultimately provided greater opportunity for the business to fund new development efforts.

1. Background

Our IT organization is responsible for all technology functions and services for internal operations, consultants and external clients. This includes application and infrastructure development and maintenance, IT security, support, and more for many applications across various groups. Within our IT organization resides the Support Organization, which is distributed into many groups to support IT’s infrastructure, applications, databases, environments, etc. The focus of this experience report is the 3rd Level Application Support Team (aka Support 2.0 Team), which primarily handles application related and analytical troubleshooting issues for 100+ applications.

The Support Organization is divided among various groups (Infrastructure, Database, and Application) in a 3-tier escalation model:

- 1st Level → Initial triaging
- 2nd Level → General How To’s, Infrastructure, Data Management
- 3rd Level → Application related and analytical troubleshooting

These three groups were heavily divided, and had different objectives, different Service Level Agreements (SLAs) and rarely communicated effectively with each other. Many support requests were passed from group to group with no clear resolution. This led to frequent conflict and blame between the groups. The lack of team collaboration and effective communication created an unhealthy culture and working environment between the groups, which impacted the responsiveness to the customers.

3rd Level, which supports applications developed by the Application Development group, is completely outsourced and fully controlled by the vendor. There
are 80 total support engineers with a mix of ten onsite in New York, five in Cincinnati and 65 offshore in India. The distributed nature of the team contributed to many of the challenges faced by the 3rd Level group such as team isolation and lack of team communication, which led to lost opportunities to learn from each other. Additionally, the team was not encouraged to solve problems right, but to solve them quickly. Therefore, the mundane and repetitive nature of the work led to poor motivation and an unhealthy culture. The end result was frustrated customers who do not receive the type of quality service that they deserved.

2. Reduce Support By How Much!!!!

Despite not having insight into the cost drivers of the support organization, leadership felt that the costs for 3rd Level application support were too much (out of proportion compared to the industry when comparing between development costs and support costs). Since support was fully controlled by the vendor, Leadership at our IT organization had little visibility to the following:

- Why we needed 80 support engineers to support our applications
- What types of requests we received
- What is the frequency of the requests we received
- What is the effort to resolve each support requests
- What are their strengths and weaknesses of the support engineers

As the Application Development group continues to build new applications, our support vendor led us to believe that we would need additional support engineers to support the new applications, which subsequently raises support costs. This seemed misguided, especially since the Application Development group has been building quality, relatively bug-free applications using Agile for the last year and a half. Therefore, one would think the rate of increasing support resources should decrease over time. As a result, the Leadership Team felt compelled to make drastic changes to better understand our actual support needs.

Leadership reduced the number of contractors from 80 to 50, but added eight Full-Time Employees (FTEs) to the support team (Architects and Project Managers with good problem solving and organizational skills) and drive the initiative described in this report. The new team was initially tasked to determine how to handle all incoming requests with a smaller team. To help them, Leadership helped the team to shape the goals for the next four months. The goals that were collectively agreed upon were:

1. Identify a root cause for every ticket
2. No increase in customer noise / complaints
3. Make the support organization a fun place to work

Understanding the root cause was a natural fit to gather the appropriate data. By understanding the root cause of every ticket, the team would eventually be able to identify patterns and potential areas of improvement. Once the information is provided to Program Managers, they would be in a better position to prioritize and make appropriate decisions, such as which defects to resolve first. However, identifying the root cause of every ticket required discipline from the team. Many of the team members could permanently fix the issues themselves. However, there were some low hanging fruit that the team would be able to resolve, and their diligence to identifying the root cause would eventually enable larger more significant changes in the future.

The second goal of preventing an increase in customer noise and complaints was also intriguing. There was no metric used to track this goal, but rather would be based on qualitative feedback from end users to Program Managers. Why not take the appropriate steps to reduce customer complaints? Two reasons: First, it was a daunting task simply to service the same volume of incoming requests and identify the root cause with a significantly smaller team. Second, this was another opportunity to gather the appropriate data about the pain points faced by the end users. Leadership felt that trying to take the additional steps to improve the customer experience would take the focus away from understanding and prioritizing the pain points.

The third goal was actually thought to be the most challenging. Is it possible to make the support organization a fun place to work? Throughout the industry, our organization notwithstanding, the support teams have always been seen as second class citizens compared to development teams. There is no motivation or growth plan for the support engineers. They simply pick up a ticket from their queue, process the required request, and close the ticket. How could we get these unmotivated, yet knowledgeable, engineers to begin to think out of the box and get to the root cause of every ticket? How can we get them to
challenge the requests instead of mindlessly processing them? With an unhealthy environment and culture, such a change seemed extremely drastic and unheard of. However, the Support 2.0 Team knew that it was important to significantly improve the environment and culture of the support organization so that the changes they eventually employed are sustainable and generates outside interest to work in the support organization.

3. “Goals are Wildly Unrealistic!”

One month into the project, the Support 2.0 Team was significantly overwhelmed. The backlog of requests was growing, there were many complaints from users, and it seemed unrealistic to be able to identify the root cause of every ticket. Several team members verbally and openly expressed their concerns about rethinking these goals. Instead, the team agreed to embark on a radical approach in an attempt to achieve the aggressive goals. The large support team was divided into sub-teams and they agreed to collaborate and to meet periodically to review progress and make adjustments when required. This began the era of the implementation of agile practices to support.

4. Agile Approach to Handle Support Requests??

The team understood the benefits of agile, and felt that many of the practices could help within support. However, they wanted to avoid “trying to fit a square peg into a round hole.” All of the FTEs have worked on development projects using agile for at least the past year, so they realized that it would be unwise to fully apply all of the agile practices. Instead, they embraced agile principles and specific practices that initially made sense, and then incrementally adopted more practices as appropriate. Some of the initial practices that were adopted were team co-location, organization into smaller sub-teams, daily stand up calls, regular sprint reviews and retrospectives.

First, the team decided it would be best to co-locate where possible. Therefore, those in New York secured a conference room for the duration of the four month project. Those in Cincinnati and India did the same. Being able to work together was a significant boost to productivity as we were able to tackle problems together and learn from each other in real time. Without the co-location, none of the benefits achieved would have been possible.

Secondly, the team re-organized into smaller teams based on application technology (Java, Notes, etc), application family (People Systems, Financial Systems, etc), or a combination of both. Thus the eight FTEs and 50 contractors were broken up into smaller teams of one to two FTEs and seven to eight contractors. The smaller teams enabled better communication flow and stronger team collaboration. Instead of individually working on support requests, they shared the knowledge and information about the requests that they serviced. In addition, they reached out to each other when challenges arose, which eventually built a stronger and more well rounded team. Because of the improvements in team dynamics, they no longer built a stronger and more well rounded team. Of the improvements in team dynamics, they no longer feared reaching out to others for help. They all now approach support as a team event and help and learn from each other where possible. This built a stronger, more cohesive and motivated team.

Each sub-team also participated in daily stand up calls, where they would collaborate to discuss the tickets they handled yesterday, the tickets they plan to handle today, and any major roadblocks, thus a slight twist on the normal development project daily stand up call. Additionally, throughout the day, any major issues or urgent tickets would be collaborated by the team to collectively discuss who would be involved and how to proceed. This frequent collaboration enabled the FTEs to better understand their respective sub-teams and identify teams and individuals strengths and weaknesses. This knowledge was extremely important because another month into the project, the team was asked to make further reductions in headcount. Having a better understanding of the teams strengths and weaknesses enabled the team to make more informed decisions about the feasibility of additional reductions and who would be released. There were concerns that such additional reductions would de-motivate the team, but on the contrary, the stronger support engineers seemed to strive in the new collaborative environment and were less concerned, as they were getting the appropriate recognition for their efforts, while it became obvious who were not holding their weight.

The Support 2.0 Team also organized bi-weekly sprint reviews, where the team would discuss the progress towards their goals. For instance, each sprint review the team would explain the percent of root causes identified and the plan to increase it each week. Additionally, they would discuss specific issues from customers and determine an action plan if required. The team also discussed positive feedback from the end users for the support team, which was critical as the team was recognized for their efforts in front of the Leadership Team. The FTEs went out of there way to ensure that the support engineers that did well were
congratulated and recognized. This helped to increase the motivation, even when we were making further reductions.

Furthermore, the team would discuss any patterns that emerged. Some examples include significant amount of missing admin functionality in a particular application, specific areas of failures within databases or servers, etc. This was the appropriate forum to raise such concerns to the appropriate actions could be taken. The team would build upon each sprint review to ensure that they are addressing the goals, the needs of the Leadership team and adapt accordingly. The Support 2.0 Team also organized retrospectives to continue to learn from their experiences. Each sub-team held their own weekly retrospective, and then the team as a whole met bi-weekly to discuss summarized feedback from the sub-team retrospectives. This was a pretty standard retrospective about what worked well, what could be better and action items, and was fairly collaborative and effective.

5. Results / Benefits

The agile approach applied to the support organization enabled the team to make significant progress towards their goals. For the first goal, although they were unable to identify 100% root causes for every support ticket (only 75%), the mentality of root cause identification empowered the team to completely re-think their approach to handling support requests. No longer do they process every mindless request that comes their way. Instead, they begin to ask questions: Is this really a support task? Can I push back on this request? Is there another group that can handle this request? This helped the team be more efficient in their processing. Additionally, the team was now able to identify patterns and major areas for improvement. Due to this investigation, three major changes ensued regarding the development of applications at our IT organization. 1) Three major projects were kicked off with the sole responsibility of fixing the high priority issues identified by the Support 2.0 Team. The expectation is that these projects will run for the entire year, thus eliminating a large portion of future support tickets and support costs. 2) Many, if not all application development projects, now allocate a certain percentage of points to fix defects. In the past, feature stories generally received a higher priority than bug fixes, but this mentality has changed. 3) Support Team members are included on development projects to help developers think about ways to ensure quality and supportability of the applications going forward. This collaboration happens from the start of projects. All of these efforts are helping to reduce support costs and improve the quality of applications.

The second goal of maintaining customer noise was surpassed. At the start of the project, support requests were thought of as a “black hole”. End users would submit requests, but they would not have any relevant information about the request. They did not know the status of the request, or who was working on the request, or when it would be completed. At the end of the project, end users started to gain confidence that their specific support requests would be met. Additionally, they were very appreciative of the prioritization discussions and of the large amounts of efforts the team put in for the critical issues. Finally, since the team collaborated frequently and often, a sub-team naturally formed with the individuals with the right skill set when major issues occurred. The team can now independently self organize and form groups seamlessly to tackle the major challenging issues, a huge win for the team.

The team also surpassed the third goal, as the culture and environment significantly improved. No longer is support seen as the “red-headed step-child” or a place where no one wanted to work. There were a number of requests from developers and FTEs to be involved with the support team since they understood the ability to make an impact. IT resources of all types (Scrum Masters, Product Owners, Developers, etc) recognized the value and impact by improving supportability of applications and is a great place to make an impact. This drastically changed the culture and environment of the support organization that we now feel is sustainable over time.

Another benefit is that barriers were removed between development teams and support teams. By collaborating and working together, the culture of support issues changed. No longer did the teams point the finger at each other for critical issues, but instead chipped in and set appropriate expectations to ensure that the right people are involved to resolve the appropriate issues. Everyone in the Application Development group now recognizes that production issues are an IT problem, not a support issue versus a development team issue.

Finally, our IT organization was able to achieve their main purpose of gaining a better understanding of the major cost drivers, which put the leadership team in a better and more comfortable position to make any future budgetary decisions.
6. Conclusion

Agile principles and practices are not limited to application development projects. It is possible to use the same agile principles and many of its practices to create an effective and collaborative support organization. Such principles can also help transform the support organization from an operational cost center to a value added thought partner. Today, the Support 2.0 Team at our IT organization is reached out to early in projects to help develop and build quality applications so that support volume is low or non-existent.

As a result, these methods have improved the responsiveness of the support organization, the long health of existing applications, the quality of application development and ultimately helped to reduce support costs.

Without the improvements created by the support organization, our IT Group would have had no budget for new application development, thus another huge win for our sponsors.