When do self-organising agile teams need managing?

Despite the agile manifesto [1] itself stating as part of its principles to 'build projects around motivated individuals', 'and to trust them to get the job done', it still acknowledges the need for a scrum master, not necessarily to act as a manager in the traditional sense, more to ensure adherence to the values and practises of scrum and to remove any impediments to the progress. The scrum master is responsible for the process used during the software development [2]. Within these subsets of the role are the more traditional management processes: scrum practises may require a level headed methodical approach when requirements are changing late in development, whereas impediments to a sprint may need a more lateral problem solving approach. Both management and teams jointly manage issues, risks, and solutions [3] however in a daily stand up the responsibility falls upon the scrum master. This pivotal role can be filled by any team member per scrum and as such, consideration must be given when choosing between candidates as to their suitability, not just for responsibility, but also their leadership style and how fitting that is to the sprint tasks at hand.

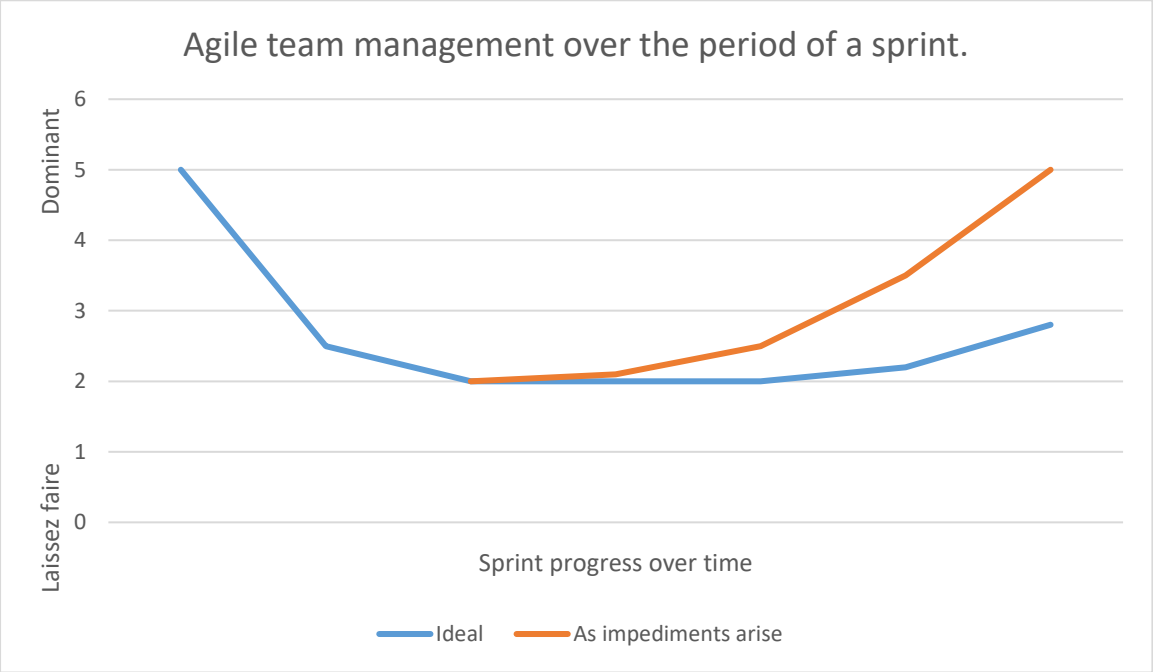
Luckily there already exists a well proven model of managing styles which we can adopt to study the repercussions of each in the context of Agile game development. Whilst the names and number of terms changes, the extreme values of the scale are usually agreed upon. From Autocratic as the most dominant, to Laissez-faire which translates to soft touch. Management styles are distinct from each other in the relative importance of desired goals [6]. Leaders and managers are concerned with two aspects of their work. The first of these aspects we term task accomplishment, organizing activities to perform assigned tasks. The second aspect we term maintenance of interpersonal relationships, tending to the morale and welfare of the people in the setting [4]. Each manager's own style is weighted on a gradient towards one of these and it is widely accepted that the optimum management team encapsulates the entire range. It is important here to differentiate between a management team usually operating over a long period and a single scrum master over a much shorter sprint.

We should also consider the team members when choosing a way to manage them. Studies confirm software developers typically fall into specific personality types, the most prominent being ISTJ using the Myers-Briggs Type Indicator (MBTI), introverted, highly rational and analytical 'thinkers' rather than 'feelers' [6]. Game developers are generally more abstractly artistic and thereby tend to a more extrovert nature. An extroverted team, can be difficult to manage correctly, however the highly trained, intelligent, and capable nature of software developers counter many of these difficulties. In a highly trained team for instance, little effort needs to be spent on training; a staple of the more dominant styles. In fact, with often specialised roles a team may not require any personal management. The agile manifesto [1] reflects this in its self-organising guidelines. Perhaps this is some of the essence of the Scrum methodology. Not the addition, but the exception of certain techniques and practices, given the software developing prefix. The role of the product owner defining the rules and parameters of the project, siphoned off in portions through a scrum master to a very capable team with the freedom to excel. Perhaps then it is not the team members that need managing, but the process itself.

It is given that dominant forms of supervision best suit inexperienced team members who need direction whilst highly qualified staff will respond better to a Laissez Faire approach. Given the analogy of managing a process instead of a person lends itself to the idea that perhaps dominant forms may suit a method where experience is not a factor and the Laissez Faire end of the spectrum where the team's individual expertise must be relied upon. In which case, we must look at the

different stages of a project or sprint in terms of freedom of direction and pressure to complete. Whilst this initially would seem to be a direct gradient increasing over time, there must initially be a strong element of control. In a game development cycle the start is often the planning stage where in lieu of absolute parameters the entire team is concerned with design and creativity. Be it creating the assets, or scripting behaviour, care must be taken to keep the teams vision aligned with the project owner's objective without stifling the creativity of the individuals. Whilst on the surface this may seem to warrant a soft touch, in fact absolute instructions must be given. The motivational level in respect of Laissez Faire is low because of not interference of management [7], creativity and experience must still be pointed in the required direction.

Over the course of a sprint the need for control decreases. The role of the scrum master mostly revolves around removing impediments to the process. With user stories clearly assigned and in progress the agile principle of trust must be adhered to. Unless there are changing requirements to the projects parameters a soft touch can be offered. This is the point where capable staff thrive. As the end of the sprint approaches there must be vigilant oversight to avoid overrunning or under achieving. As deadlines near, unforeseen circumstances can have drastic implications and correctly identifying and managing these requires the autocratic strength of more dominant techniques.



Conclusion.

Given that the leadership style of scrum master can be transitioned from one style to another it is possible to optimise an agile team's management through identifying the correct choice for scrum master dependant on the sprint goals and timeframe. Despite being self-organising by nature, agile teams rely upon guidance beyond user stories, and differing coaching styles must be used to adjust for late-changing requirements and changes in focus and pressures over the course of a sprint. By utilising the contrasting strengths offered by different management styles the process can be made more efficient, team morale can be improved, and overrunning time frames can be better avoided.

[1] Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Thomas, D. (2001). The agile manifesto.

[2] Larusdottir, Marta, et al. "Stakeholder Involvement in Agile Software Development." *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*. ACM, 2016.

[3] Schwaber, K. (1995). Scrum development process. In: SIGPLAN Notices, 30(10).

[4] Eagly, Alice H., and Blair T. Johnson. "Gender and leadership style: A meta-analysis." Psychological bulletin 108.2 (1990): 233.

[5] Fairweather, John R., and Norah C. Keating. "Goals and management styles of New Zealand farmers." *Agricultural systems* 44.2 (1994): 181-200.

[6] Wiesche, Manuel, and Helmut Krcmar. "The relationship of personality models and development tasks in software engineering." *Proceedings of the 52nd ACM conference on Computers and people research*. ACM, 2014.

[7] Chaudhry, Abdul Qayyum, and Husnain Javed. "Impact of transactional and laissez faire leadership style on motivation." *International Journal of Business and Social Science* 3.7 (2012).