Sprint Report

So let us break down one report and what patterns we can use to provide enough information and follow the example explained up until now in this article.

1. Start with the **Project's Name**, **Sprint Number** and the **Period of Execution**. Clean and an easy way to enumerate the document and inform the reader if it is the most current one.
    1. Optionally provide a Table of Contents to digest the document.
2. The first and most important topic should be a general overview of the current status of the project. We usually call this **"The Roadmap"**. We can use a screenshot of the roadmap set in "Quarterly" view, displaying a couple of previous sprints, the current one and one or two subsequent. Display the Epics, ordered by priority and by engagement (the finished are first, the ongoing are following, the new ones are at the end).
    1. One can also write below the roadmap the **"Affected Epics"** by the current sprint to reduce and ambiguities to the reader. For some of the stakeholders this is enough and here is where they stop. They need to know that we are working on what was agreed, where we are at the moment and if we are keeping the estimated delivery plan. This can remove many following engagements in communication between the delivery team and stakeholders..
    2. Following are the affected epics are the **"Major Milestones"** accomplished by the sprint. A milestone is different from an epic in a way that it can be a part of an epic or can span several epics (tracked as phases). It is an agreed upon achievement that brings value to the client. If it is healthy it represents a step in the desired direction, or it can be an end of the road signpost. But sometimes it can show the complete opposite, that we are heading where we do not want to go, and can stop us before me spend more resources and make more damage - agile at its finest.
    3. Lastly is the **"Sprint Goal",** a short term mission statement that can declare the sprint a success or a failure. It is a mutually agreed statement by the team that we need to accomplish at the end of the sprint, so it is important to add it here. It shows the stakeholders that we understood what is most important during this iteration and that we have all worked towards completing it. If there are multiple sprints goals set for the ongoing sprint, add them separately and tag their state (Completed, Partially Completed or Not Completed) or mark them with colours accordingly (ex. Green, Yellow, Red).
3. Next we have **"The Status"** topic. This is a short descriptive summary of the overall status of the project or product. How is this different from the milestones then? Well, the status can give updates of certain aspects that may not be considered milestones, or it may give a reason why certain milestone has not been achieved, where we are currently at with our development effort and if we have certain blockers or impediments. The status bar can also have different accent colours depending of our understanding of the health of the project.
    1. The status can have multiple bullet points that are written in third person and are generalised descriptions written on a high level of abstraction for specific change efforts, i.e., they are accomplishments by the team or certain individuals and can be used as a type of praise for exceptional efforts and investments.

    2. It can also include information for the environment of a certain epic or a feature that has not completed its expected lifecycle (ex. it is found waiting feedback on UAT instead of moving to PRODUCTION as planned) and why it is found there and what it needs to be done for it to move forward. This is considered waste in the lean philosophy, and reduces the overall development throughput, so it is paramount to be communicated with the stakeholders and is of highest importance, so that we can quickly remove it as constraint or bottleneck to the delivery process.

4. What follows is the **"Delivery"** report. It is a simple table displaying the team velocity for the past 3 sprints, averaged on the last 5. It also displays the period that these sprints were completed in.
    1. The averaging on five sprints is a bit confusing here because we are displaying only the last three, but experienced scrum masters will immediately note that averaging on the last three might not give correct approximations (Cone of Uncertainty), and displaying all five only makes more noise. That noise comes a bit later if the reader is interested to dive deeper into the report.
5. So we reach every manager's hell: **"Risks"**. This is where the most energy is invested by each manager, tracking and following the effort as the sprint progresses, creating a holistic view and presenting it in a risk matrix. It basically is really simple because there is no standard approach and it is just individual assessment of risks (and opportunities) that have been noticed during the sprint period and might affect the upcoming one or few iterations.
    1. We start with a simple risk description, or a summary of the reasoning why we consider the certain point as a risk.
    2. Next we have the probability of it happening in the expected period (Low to High).
    3. We then note the severity of the damage it might cause if it happens (Low to High).
    4. And we consider a status of what we intend or try to do regarding it (Avoid, Mitigate, Transfer or Accept).
    5. Lastly we have a short description of an action plan regarding the status, explaining how we are to follow through with it.
6. Reaching the core of the report and starting a new phase, a more in-depth analysis of the team progress spanning multiple past sprints (up to 10) and displaying their KPIs. Simply, we note the sprint name/number (duration), total story points committed, story points completed, story points not completed, the calculated completion rate (completed divided committed), story points added during the sprints (indicating scope creep - or we failed in our risk management), and lastly we can show our bug bash progress (or how much effort was invested in bug management, either committed at the beginning of the sprint or found in our regression testing phase).
    1. A good practice is to revisit our initial estimations at the end of each sprint, updating them in a separate field our estimation after the fact, double check if we have a correct estimation feeling of our story points scale as a team and if we can improve upon it. Comparing the initial value with the new value can give us a discrepancy percentage, that we can follow in subsequent sprints to see if we reduce it, therefore increasing the completion rate. If the discrepancy widens it is an indication of "pumping up" effort points so it seems like we are delivering much more value than we actually are.
    2. An anti pattern can occur at this point. The manager is tempted to track the logged times for each item and try to correct the estimations using man-days.

   These two values should never be put in the same basket and be interchangeable. Much has been written for this and it is not a topic for this time. However, tracking time for each item is an engineer’s complete waste of time and it does not improve the team’s estimation precision, since it is not something they have concluded as a team effort, rather it was imposed by the manager. If you want to track time, track uninterupted flow.

   3. Providing a detailed table, showing each story, task and bug that was committed at the beginning of the sprint, and ones added during its duration marked as red, can give the interested reader a chance to follow the status of an epic broken down to its subcomponents. A table of items that have not been completed or have been removed from the sprint can follow next.
   4. Lastly we have the analysis of the discrepancy (if any), or a description for the reasons of changing the sprint plan. This is in a way a more detailed dive in for the status of the sprint, that can connect several points, risks, milestones and goals into a whole, giving a clearer picture of why something has happened that it was not supposed to occur - a failed delivery plan, affecting the overall release plan.

7. And last we give a slight glimpse into the **“Plan for our next sprint”**, writing down the delivery plan into a neat table of a task breakdown structure, our stories, tasks and bugs with our initial estimations.
   1. The structure of the table can follow the previous table of completed items so for the next report one can simply copy the planned items into the completed (and hopefully not) incomplete items.