Organization background

The organization is a New Zealand small indie game development company. Initially, they had 7 shareholders include 4 staff members which are all experienced software developers and 3 other investors. After the first game prototype was developed, they have entered an international game contest and eventually successfully promoting their ideas. The company investors then pursuit and framed the application to the government. As a result, the project has been funded by the government and the local government combining to support the growth of the industry in a regional area in Aotearoa. From the initial workforce of only 4 members, after 3 months, the company has scaled up to around 50 members which includes several roles such as developers, designers, artists, etc in order to make this game a reality.

Project information:

* Duration: 2 years
* Context: develop an open-world multiplayer interactive game.
* Platform available: iOS, Android, Unix, Windows
* Release plan: all platforms simultaneously. No beta release.
* Sponsor: government plus the local government.
* Workforce: 50 employees.

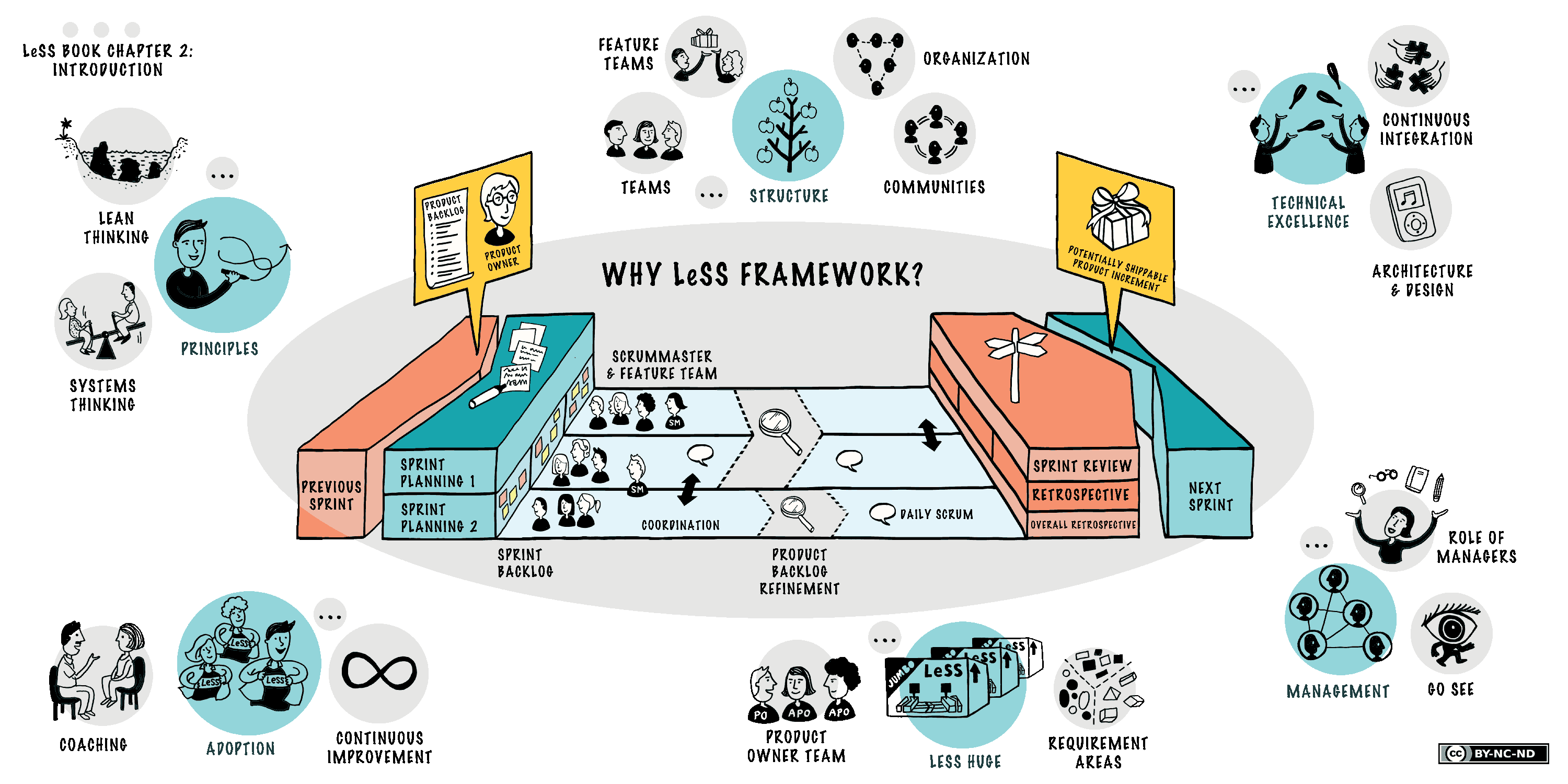
Recommendation for project approach

Agile: in term of effectively developing the software. We strongly recommend using Agile software development over the traditional waterfall methods. Waterfall development has been large condemned in the literature and research of software development practice. It has been shown to poorly suited to deal with changing requirements, produces unreliable schedules and estimates. Waterfall development only suits fit projects with little change and low complexity level (VanderLeest , Buter, & College, 2009). Consequently, Agile gradually replaces the classical plan-based methods as it incorporates many of the best practices of the last couple of decades. Agile software development method is correlated with a more balanced use of internal software documentation, contribute to knowledge sharing, increased project visibility, productivity and coordination effectiveness. Meanwhile, no correlation with increase in pressure and stress were reported (Lagerberg , Skude, Emanuelsson, Sandahl, & Ståhl, 2013). It is obvious that Agile is a superior method with an enormous amount of benefits. We want to point out some specific reasons why Agile software development should be an optimal choice for the firm project:

* Greater flexibility: even the company believes that they have a good initial of the concepts and their ideas are also promoted in an international stage. The project requirements are still very likely to change due to several factors: new employees expand the initial understanding and explore new possibilities, gamers demand latest technology in graphics, gameplay which are just introduced during the project development time, new features are required to make the game stand out among other competitors products, etc. The organization has to always be prepared and willing to adapt themselves to any potential changes. In contrast with Waterfall, Agile software development is greatly known for its flexibility when dealing with changing requirements by its concept of iterative and incremental development.
* Better coordination effectiveness: agile teams are reported to be more aware of not only their team status but also other team’s ones as a result of working in an open office space which includes multiple teams task boards. This is a critical aspect for the company project. Since there will be multiple teams responsible for multiple platforms development. It is crucial for a particular team to has the awareness of other teams progress to avoid going too far ahead or falling to far behind the mutual progress. Keeping track of multiple team progress can quickly identify which department is requiring more assistance so corresponding actions can be taken. This helps to fulfill the firm objective of releasing all of their variants simultaneously.
* Continuous user involvement: one of the featured characteristics of Agile is the continuous involvement of client/owner. Product demonstration is shown to stakeholders before every iteration and their feedback & requirements are always collected. This process increases stakeholders interest and awareness as well as making them feeling respected and involved. Out of 4 initial members, the company shareholders are 3 investors whose involvement are encouraged and enabled the company to enter the contest. They also framed and pursued the application to government. We are sure that they play an equally important part as other initial developers. Agile software development offers an excellent opportunity for non-developers to contribute towards the project success.
* Facilitate knowledge sharing: using practices like iteration planning meeting and iteration retrospective. Project team members have an opportunity to gain insights from other functional areas besides their assigned one. The developers have learned fairly sufficient skills to be confident enough to be put in unexpected situations. Hence, it is especially helpful in case different teams demand resources swapping or assistance from other team to solve an immediate problem. We expect that these scenarios are going to happen regularly during the product development process since the organization has 4-5 different teams running concurrently. A better knowledge sharing also benefits the company in the long run. Continuously improved employees are equivalent to better overall productivity, not only for this current project but also for future ones.
* Enhance trust among members: Software developers generally does not want to be told how to do this and that, they want to be trusted on their abilities. Normally, a person cannot express all their capabilities while being doubted. However, since this is the company very first project and the majority of the employees have not had any chance to work together, skeptical feelings are inevitable. Agile methods can minimize this problem through increasing trust among team members by enhancing transparency, accountability, communication, knowledge sharing and feedback. (McHugh, Conboy, & Lang, 2012)

Agile Framework: One important aspect with this scenario is that the organization has recently up- scaled from a 4 men vision to a 50 employees organization. In order to effectively and efficiently complete the project in a timely manner we would recommend implementing an agile framework. The framework that we suggest is Large- Scale Scrum or LeSS agile framework. A framework is a basic outline of how an enterprises teams and workloads are managed. It is a set of principle’s and rules which help to create a work standard throughout an organization. LeSS or Large- Scale Scrum is an effective agile framework. There are several benefits to choosing LeSS framework for this particular project:

* LeSS accommodates 2- 8 teams- For this particular project there are currently 45- 50 people employed. Standard LeSS framework is designed to accommodate this size due to the basic Scrum structure. Other frameworks are designed to manage much larger, enterprise level organisations which would not be appropriate for this project.
* There is one product- level Sprint amongst all SCRUM teams- This framework design has all SCRUM teams sprinting in parallel, which fits perfectly with how this project should be implemented. One of the objectives of this project is for the game to be released at the same time amongst all platforms. This requires that all platform developments must be incremented at a similar pace. This framework introduces a one- product level sprint amongst all teams, which means that overall development of the game on all platforms moves more or less at the same speed. Which would lead to a simultaneous release.
* Each team is self-organizing- This aspect of LeSS framework allows a team the flexibility to form their own working patterns and workload arrangements. Especially in the context of this project where there are multiple programming languages and multiple platforms, the ability for a team to organize themselves would help reduce complexities which would otherwise occur if trying to organize all teams in the same way. This also seems to work well among smaller sized organisations.
* There is one product owner with one product backlog- This idea is very similar to the basic SCRUM ideology but just on a larger scale. It would benefit this project because as a small independent game company, it is possible to have 1 person with the ability to have a focus on both the customers needs and the needs and ability of the team. Other frameworks would most likely separate the two, increasing communication loads, making the project overall less efficient.
* A lot less managing staff compared to other Agile framework- Due to this organization having a relatively lower number of staff than large scale enterprises, LeSS helps to accentuate this characteristic as a strength. Lower numbers mean less complexity and less managing staff since there are less people to manage. Other large scale frameworks work hard to provide structure in all areas, if implemented in this project it would be in danger of ‘over organizing.’
* LeSS builds upon a “basic” Scrum idea- As mentioned before, LeSS is easy to implement as it follows the same principles of a small SCRUM team environment.
* Flexibility over procedures and rules- This benefit will help especially when working on multiple different platforms simultaneously.
* Recommended for fast growing, smaller organizations.



Scrum: there are lots of way to operate Agile methods for a project called Agile methodologies. There is no static implementation of Agile into every software development projects as Agile methods are designed for customization, organizations are offered a set of principles and practices based on their culture and value ( Soundararajan & Arthur, 2011). Knowing what Agile methodology best suits the firm project is important. In our opinion, Scrum offers all the best principles and practices that the firm needed to successfully deliver the product on a timely manner. There are many Scrum practices that we introduce in our Project Practice Guide in which we also have explained the concepts of them. We strongly recommend the firm to look at it and pick up all the practices that found out to be most feasible and useful. Here we are going to outline some practices that is particularly crucial in the way they are positively influence the development of the organization project as well as how to effectively set up them which were grouped in 5 Scrum phases. We also specify some potential modifications to the practices to optimize them for the company project.

1. Initiate

* Scrum team: Scrum teams function better in small size. There must be separate teams for each of the game versions because they require different source code as well as skill sets from assigned developers. Additionally, Scrum teams are appeared to be cross-functional teams which may be a challenge for some firms. However, this is a good match for this specified project as the organization possess 45-50 people who play different roles. Thus, the company is suggested to divide their workforce to 4-5 Scrum teams which are responsible to develop distinct variants of the games such as Windows, Unix, iOS, Android. Each Scrum team has 9-10 people and has to include all the possible roles for instance developers, testers, designers, artists, etc. identify who can be a Scrum Master in each team is also essential. More information about the roles within a Scrum team can be found in our suggested Project Practice Guide.
* Product backlog: based on the project requirements, each Scrum team should be able to create a product backlog. Even the game function required are the same for all Scrum teams, but separate product backlogs make it easier to setup teams and activity on Trello, etc. Additionally, due to the programming language nature, same tasks in different programming platforms may have different difficulties which leads to the shift in user stories prioritization of different teams. How to effectively gather and manage project requirements will be presented in a separate section “Requirement Collection and Assessment” in this report. In order to prioritize the stories, a method which is commonly used is MosCoW which categorizes these stories as Must Have, Should Have and Could Have.

1. Plan and Estimate

* Sprint Planning: the earlier sprints should be spent on developing fundamental functionalities. Which are categorized as Must Have features according to MoSCoW. Therefore, the sprint goals for these sprints could be “Implement the ability to create a character”, “Implement the ability to customize character looks”. In term of effort estimation, each development team can apply “Planning Poker” method to estimate how long does it take to complete a single user story. The full Scrum team will take part in this game, hence, all the results are based on team joint decision. Once the Scrum team has fully clarified the stories which need to be completed in the sprint along with their estimated effort, create a Sprint Backlog and includes all the assessed user stories. Resource assigning to tasks is also an important activity, this should be discussed by all Scrum team members before making the decision. Avoid forcing developers to do tasks, always have to ask for their opinions first.

1. Implement

* Sprint Execution: this is when the development team starts to develop and test features from the sprint backlog. It includes Daily Standup and Daily Scrum which is the meeting that everybody has chance to share the status of their work they are doing or get help if anything is blocking their process. This process can be further optimized in our organization, we expect the organization to be capable of setting up an open working space includes different task boards so everything is clearly visualized to facilitate the project visibility and knowledge sharing among all employees. There is one drawback of Scrum is the high chance of observing bottleneck when there is a process cannot satisfy enough amount of works need to be processed. This circumstance is likely to happen for the firm since the number of testers almost always less than developers in any team while the amount of user stories to be developed and tested is always equal. As a result, testing could potentially be a bottleneck. In order to eliminate this problem, we suggest the firm to adapt some Kanban principles. Kanban methodology was investigated to have many considerable benefits of motivation and controlling over the project activities (Ikonen, Pirinen, Fagerholm, Kettunen , & Abrahamsson, 2011). Most prominently, the limit to Work-in-process (WIP) principle. The firm is going to setup the limit work for each process, the work limit number of each process is vary. How to set up these limits comes with experimentation. However, here are some common formulas that the firm can apply to effectively set up the limit:
* The amount of all user stories (exclude the done) in the broad: equivalent to the number of Scrum team / 2. Hence, this number could be 5 for the firm development teams since there are around 10 people in each team.
* The sprint backlog (to do column): stories wait for their turns to be processed. Half of the possible user stories on board. This could be set up to 2 since there are only 5 user stories simultaneously stick on the board.
* The development column: stories being developed. This number can be set up to half number of developers. There are around 5 developers in each team, hence, the limit may be 2 or 3.

There are several benefits of having a small queue. It forces developers to work together on the same task. This makes user stories can be done much faster and also facilitate knowledge sharing and collaborating skills. Additionally, developers will less likely have to switch task which may wastes up to 20% of their effort just switching to the new task. Importantly, the Scrum team has to ensure that the work flow is visible using a Scrum broad.

1. Review and Retrospective

* Sprint Review: the key characteristic that the firm should focus on the Sprint Review is stakeholder involvements. This is a great opportunity to gather feedback from the 3 company important shareholders along with others as well as presenting any demands for changes in the budget. The project team has to ensure that they still observe that the project progress still serves the business objective of the company. Key stakeholders satisfaction is always an important criteria to measure the success of a project.
* Sprint Retrospective: the sprint retrospective is an appropriate way to close a finished sprint and head to the next sprint. Every Scrum team should be able to identify what went well, what was the problem and what action will need to be undertaken to improve the productivity in next sprint during the sprint retrospective. It is encouraged that only Scrum members participate in this Scrum ritual. Any kind of external people involvement should be performed in the sprint review instead.

1. Release: release the games which covers all accepted deliverables in all 4 platforms. Besides, documenting a lesson-learned report to close out the project. The lesson-learned report should specify if the project has met the triple constraints of time, cost, and scope. Identify what has gone well, what has not and what should be done differently in the next project.

There are some tools that can help the development team set up Scrum on the cloud, which is really helpful when the team has to work remotely. In our project tool guide, we suggest Workfront and Trello for Agile project management software (PMS). Microsoft Project 2016 is also a PMS but it is not designed specifically for Agile software development. Depending on the firm current financial analysis, the organization can choose to use whether Workfront or Trello. Workfront is a high-end PMS tool which can support both common project management and agile project management aspect. Regardless of its price and some potential risks of being a relatively new software to everyone, it seems a worthy investment for long-term. On the other hand, if the firm cannot afford such a high-end tool, a combination of Trello which is completely free and Microsoft Project 2016 is another decent option. In this report, we are going to demonstrate Trello as a tool to set up Scrum methodology. It is a more feasible and safer option right now as the company was just founded and this is also its very first project.

* Numbers of board: there will be 4 Trello boards for each Scrum team which responsible to develop the games platform of iOS, Android, Windows and Unix respectively. Members only works on their allocated board but their memberships are included on other broads as well. This helps everyone to keep track of other department works which aids 4 departments to move forward at a same pace.
* Trello columns: compulsory columns on the Trello board are Product backlog to contain and priorities user stories, Sprint backlog to contain set of stories for a specific sprint, Documentation to contain all project management-related documents such as team contract, burndown chart, etc. In development, done development, testing and done columns to control and monitor the flow of user stories movement.
* Trello cards: each user stories card is supposed to have an acceptance checklist, resource assigned, estimated story point and a priority brand. Additionally, the number of cards available at the same time for each column are limited except for the done one due to Kanban WIP limit practice.
* Communication & Collaboration on Trello: team members are encouraged to make project-related discussion right under specified cards. Trello provides a handy and robust communication tool that the firm should be able to utilize it. Since this is the first time the company deals with the communication of around 50 people. Project managers should specify some communication regulations on Trello and put them under the Documentations column such as: the discussion topic must relate to the project, members are advised to only make comments on the cards they are assigned, avoid making nonsense comment, etc. These rules can help the firm to better manage the communication of employees as well as reduce the chaos of having so many communication channels.

Version control: GitHub should be the top choice for the version control as we have suggested in our Project Tool Guide. The learning plan and learning resources for anyone who is not yet familiar with GitHub is made available in our Tool Learning Package. In this report, we firstly want to emphasize 2 options of code modification on GitHub:

* Direct Contributions: users push their commits directly to the repository by the using push command and the changes will be merged instantly.
* Indirect Contributions: users commit and push their changes on a remotely created branch, create a Pull request so everyone has a chance to review, compare the changes before deciding whether to merge the test branch to the master branch or not. This feature opens the concept of Pull-based development on GitHub. Additionally, this can facilitate communication among developers.

The way code modification has been proposed does greatly influence the success or failure of a build, pull request are much more likely to result in a successful build than direct commits ( Vasilescu , Schuylenburg, Wulms, Serebrenik, & Brand, 2014). However, study found that in the period from February to August 2012, 315,522 original repositories received a single commit. From those, 53,866 (17%) received at least one pull request, while 54,205 (18%) used the shared repository approach, having received commits by more than one developers and no pull requests. This indicates the preference trend of direct contribution over indirect contribution. (Gousios, Pinzger, & Deursen, 2014). We suggest the firm not to follow this trend. Instead, carefully preparing all the developers to embrace the concept of Pull-based development as it has been proven to be superior in various aspects than making direct commits.

* Setup GitHub repositories: the firm need at least 4 repositories on GitHub. One for each platform version of Windows, Linux, iOS and Android. Developers are provided access to their assigned repository only. Each developer will have to clone and work locally on their own repository.
* Pull-based development: developers are forbidden to make any direct changes to the repositories. Any commits have to be presented using a pull request. The changes will be merge only after considerations of all other developers. They are expected to use GitHub for every Sprint Daily standup.
* Collaboration: developers are encouraged to make discussions directly on GitHub about each other works. Discussion contents must be relevant and on topic.

Quality assurance:

It is essential that the development teams have to ensure that all the all the product deliverables function the way that they are supposed to do. The cost of project failure is really expensive. Several quality assurance practices that we have mentioned in our Project Practice Guide really help improving quality of the firm product. Besides, we want to focus on these following approaches related to code quality:

* Pair programming: this is related to code quality assurance, 2 programmers working together at a same time and place. This practice enables continuous code quality testing by one programmer. Also facilitate knowledge sharing by exchange opinions between two programmers. Pair programming is feasible and suits well the concept of Scrum while everyone is working in an open-space office. It can be performed easy by the firm since all firm development teams mostly do not work remotely.
* Code review: this should be done in any Sprint Daily standup. A piece of code written by a developer will be checked for quality by another Scrum team member. Pull request on GitHub optimize this practice. It is a next better option in addition to pair programming especially when development team has to work remotely. There are several tools which can ease this process by automatically track for bad-written code or bugs, we suggest developers to use FindBug. It is free and compatible with most IDEs. More information on this tool is referred on our Project Tool Guide.
* Coding standard: since this is a very first company project, state a code standard is crucial as it helps avoiding many potential coding problems later on. The majority of developers are working together for the first time and each of them may has their own style of coding. Code standard can enable consistency and better understanding of code across the project members. The firm has to introduce their own specified code standard to developers before the start of any actual development. Some example code standard rules that the organization can make is:
* All classname must start with a capital letter
* All method/functions must start with a lower case letter
* All constants must have underscore between words.

Testing is an obvious activity for quality assurance. It ensures the product created does properly what it supposed to do. There are so many scenarios that makes a software do not work, we need to cover all of them by having different levels of software testing. Quality assurance is important to deliver the best products that the firm is capable of. There are 2 types of testing which are manual and automated. The firm need to be able to implement properly both of them. Information of manual and automated test along with different type of test is presented in our Project Practice Guide. Below are some approaches we recommend for testing along with reasons:

* Behaviour Driven Development (BDD): to ensure that the code always do their job right, BDD is an effective approach. It makes the code much cleaner because it only contains what it supposed to deliver. This results in much faster product delivery. Since the firm is trying to develop an interactive game, there will be a lot of unit tests and acceptance tests to be covered, ensure that the written code passed both of them is important.
* Test first: create test first before writing any code. It is a good practice deriving from XP (Extreme Programming) methodology to help developers to really consider what needs to be done and have immediate feedback while working. Moreover, developers design will be affected by the thinking of test everything of value to customer. This mindset is beneficial for the firm business as one of the decisive factors of project success is user satisfactions. Additionally, it improves developers responsibility, any code they have written have to pass all specified tests.
* Test Driven Development (TDD): a game normally covers lots of functionalities which make testers have to do a large amount of work. The firm can prevent large workload to testers by minimize potential problems can occur in each function. Using TDD make sure that done user stories have always passed all specified unit tests.

Managing change

Changing requirements is a natural characteristic of an agile software development project. Change could be made in any process or anytime where and when appropriate. Fortunately, agile software development method is very flexible. It embraces and welcomes the idea that requirements can evolve during the project development. Managing change is inevitable in Agile since there are so many reasons that requirements are not stable from the start to the end in a project. Here are some common ones:

* Team mistake: the team failed to identify an important requirement that was later discovered by a stakeholder. This possibility is likely to happen since the firm needs to have stakeholders involved frequently in the Sprint Review.
* Political issues: there could be a power shift among different stakeholders. Therefore, the user story prioritization also needs to be changed. Again, continuous stakeholder involvement can help the firm to quickly react to this problem.
* Legislation changes: state laws and federal laws change during the development could possibly affect the organization project. This is mostly due to external factor, we are quite sure that the firm will be informed about the changes and take correct actions accordingly.
* Market changes: sometimes changes are done due to the competition among different companies. In order to be success, the organization have to be aware of market changes.
* Stakeholder requests: new feature that stakeholder wants to add on during the project development. This potential problem needs to be handled nicely to avoid negatively affecting the currently undertaken process.

Since we have suggested the firm to use Scrum in particular. Handling changes in Agile and Scrum particularly could be done often and elegantly. Stakeholder actively involved can helps the firm to quickly get new requirements and prepare for it. Any changes should be documented as a change request form. The decision of either reject or approve the changes is evaluated by the whole development team after reviewing the change request form. If the majority of the team member does not agree, the change should not be taken place. Once new request is approved by the team, it is captured as user story and added to the Product Backlog at any time. However, the Sprint Backlog should always remain frozen. Product backlog grooming will happen before next sprint execution to analyses all user stories. This results in the whole user stories remaining will be assessed and prioritized again. The key factor that we want to remind the organization is avoiding making changes during an iteration. Nonetheless, Scrum concept really reduces the possibility of this to happen. There is a fixed number of functions which were decided to be built in each sprint and this number cannot be modified. In case a change that requires code changes, the organization development team should not do that straight away. Instead, they should look for the source change top-down from requirements and do the all changes at that path first. This approach ensures that the organization always carter for all possible changes from high to low level. Informative content about product backlog grooming and middle iteration changes is presented in our project practice guide.

Stakeholder Analysis: knowing how to categorizes the stakeholders into different groups is essential. It helps the firm to capture all key stakeholders that are important to the project, as well as find an efficient way to manage this group of stakeholders. We are going to evaluate some stakeholders group that the firm need to be aware using Stakeholder Typology. Information about Stakeholder Typology can be found in our Project Practice Guide.

* Dependent Stakeholder: the players of our game when it is available in the market. This is the main source that the firm will get financial income from and also a decisive stakeholder groups that determine the success or failure of the project. Players always want the game publisher to actually listen. Having a large player base is hard, but to manage a large player base is even harder. A nice Q&A, service support team can increase the satisfaction of this group. Some rewards program to loyal and contributive players, frequent interesting events, game updates based on players demand are also good strategy to manage this stakeholder group.
* Dormant Stakeholder: the government which are also sponsors for the firm project. This is a big advantage that the firm should capture. Having a sponsor which is a dormant stakeholder mean the sponsor will less likely actually care about the project. Many projects world-wide have trouble dealing with sponsor claims when they keep urging the process, demanding new features, etc. Additionally, the government can also provide good conditions for the firm. In return, the firm should maintain a good relationship with the local government and try to fulfill their goals of promoting the region.
* Definitive Stakeholder: normally the sponsor of a project. However, since the organization sponsors fall under Dormant Stakeholder category, there is no actual definitive stakeholder for our project. Again we want to emphasize that this is great advantage.
* Demanding Stakeholder: a group of gamers that always demand ridiculous changes is a clear example of this stakeholder groups. They can keep yelling the company to make changes but their opinions are not supported as well as does not represent the majority of gamers demands. The organization can just ignore them. However, separating demanding stakeholder and dependent stakeholder is the process that requires a reasonable amount of attention and effort as the cost of misclassification is really expensive.

Requirement Collection and Assessment:

Since the firm idea has been promoted in an international contest, the firm should have a clear idea of what should be expected to see in the game. Nonetheless, it is a long way from initial concept to the day the game will be released, many changes could be undertaken in requirements due to various factors. Additionally, the organization may also want to expand the scale of the game which leads to more requirements needed. Here are some suggested methods for collecting requirements:

* Interviewing
* Prototyping
* Observation
* Questionnaires and surveys
* Benchmarking

Interviewing, questionnaires and surveys are appeared to be the easiest methods to carry out but still able to produce good results. These methods should be first applied on the company new developers to clarify the feasibility of the requirements as well as gathering new ideas for the game. In addition to the developers, gamers should also be examined as they are the application users eventually. The majority of gamers opinions provide lots of insights and directions for the firm. Gathered requirements are recorded using use case. Use case do not only help the firm in identifying stakeholders by specifying actors, they also aid the development teams in finding acceptance test which is later will be useful in creating user stories.

Regarding to requirements quality, the company must have both functional/non-functional requirements. Requirements will be assessed in Scrum methodology in form of user stories through activities such as user stories creation, user stories prioritization, effort estimation, user acceptance tests, etc. Being able to deliver all the user stories on time is a goal that any project heads to.

Collaboration & Communication:

Communication is another challenge that the firm has to face. Was initially communicating between only 4 members but now that number has increased up to 50. Communication inside the firm can be divided into 2 categories: Face-to-Face and virtual.

* Face-to-Face: direct communication which was facilitated through Scrum rituals such as Daily standup, Sprint retrospective, Sprint review, etc.
* Virtual: is made on the tools supporting the software development that we have suggested such as GitHub, Trello, Workfront, etc.

Communication is the key factor to success in any project, especially in Agile and Scrum particularly. The balance use of face-to-face and virtual communication is worth achieving in this project. It is recommended that the firm should value both equivalently. In order to prevent chaos communication from happening, virtual communication regulations are specified as we have mentioned above to make discussions always stay on the topic. However, the company should not make one mutual for all teams. Empowered teams are found to be feel less stressed in Agile software development (Laanti, 2013). Each development team may have their own preferred way of working, thus, giving them some freedom to specify their own terms using a team contract. This can make employees feel more comfortable with the feeling of being respected and not being forced. Same concept is applied to all the Scrum rituals, all members no matter what role they take are encouraged to actively participate and express their opinions. Additionally, the company better make meetings short but full of informative contents rather than long but irrelevant and not direct to the point.

Planning & Tracking:

The whole concept about software development planning is related to sprint planning which has been discussed above in the Scrum section. To do planning and keep track for the whole project which consists of 4 small sub-projects (Windows, Android, iOS, Unix) running at the same time. Some traditional project management practices are helpful. These practices are not practical without the help of some project management tools (PMS) which were suggested in the project tool guide such as Microsoft Project 2016 and Workfront. Regarding the PMS choice, Workfront is more advanced and has the advantages of being a purely Software as a Service. On the other hand, Microsoft Project is a cheaper installed software and normally has all functions needed for a PMS. The choice of whether purchasing Workfront or MS Project is based on the company financial situation as well as current familiarity of employees to PMS. We recommend the firm to check our project tool guide which is written to effectively aid the organization in making the right tool decision. List of potential project management practices is presented in the project practices guide, the firm should refer to them. Moreover, it is important to know the order to carry out these practices, we suggest the firm to follow this order:

1. Gantt chart: the first chart needs to be prepared. Each development team can create their own Gantt chart to specify different activities will be carried out in each sprint along with milestones to be reached. All activities are time-boxed which makes it also matches the concept of sprinting in Scrum. A technique which the firm should use that can help duration planning easier is PERT. It is especially useful in case the task possesses a high degree of uncertainty which is a high potential problem of the firm project. Development teams can use the estimated durations specified in Gantt chart to apply them on effort estimation of user stories.
2. Network diagram: generally, it will be automatically generated following the Gantt chart. This is also the prerequisite for the critical path. This diagram visualizes all the tasks inserted in the Gantt chart. The firm can print it out and put them in the common task board where it can be seen by everyone.
3. Critical path analysis: generated to identify which is the fastest way possible to complete a project. Identify which tasks are top priority and which tasks can be delayed in order to take correlate actions. The development teams should give the priority to do all tasks which line on the critical path.
4. Tracking Gantt chart: this chart is frequently updated along the project execution. A great technique to keep track of what work have been done and the actual speed of project execution. Tracking Gantt chart should be shown to the development teams in every Sprint retrospective to analyze how well the last sprint was carried out. Lots of lessons can be learned by reviewing the tracking Gantt chart.

Managing project risks is another important thing that the firm should pay attention to. Any good project has plenty of risks, without risks a project has no value at all. It is a nature characteristics of a project. The organization needs to minimize the risks affection as much as possible, ideally to 0. In order to do that, risk management should be done to increase project success chances to identify risks and specify specific strategy to deal with them. Risk register is a good practice to address all the potential risks by providing description, root course and possible response to deal with the risk. Similar to a Work Breakdown Structure, project risks can also be visualized at a hierarchical structure using the Risk Breakdown Structure. After risks identifications, risk should be assessed to decide the impact level of them on the project as well as corresponding endeavor to minimize bad effects. Some Qualitative and Quantitative analysis are efficient in ranking and determining the impact level of identified risk. Our suggestion is using Expected Monetary Value analysis and Severity for ranking. Potential responses are all based on the assessment result, most of the project risks can be mitigated by Agile software development practices. Hence, they need to be better executed and managed by the firm to minimize potential project risks.

Conclusion

Technology is changing rapidly day by day and so does software engineering. The number of software development practices and methodologies is evolving rapidly which can make some development teams feel hard to keep track of what should they can apply for their project. There are so many software development practices and methodologies available and they mostly claim to be better than others. However, applying a software development method just by hearing that it is the best or most widely used in a project without proper assessment is a very common mistake that many firms make. In this report, we do not just recommend practices because they are claimed to be efficient. Instead, all of our recommendations are supported with clear rationale of the project attribute and software engineering literature along with empirical evidences. We have judged and evaluated all the methodologies and practices cautiously to make sure that they are effective, feasible and suitable for this particular project.