Process Grid

This grid is provided to help you define ten key processes for your team. Less formal processes are flexible and have low overhead, but increase the probability that problems will affect project success. More formal processes have more overhead and can be irksome, but reduce the impact of problems. What is a good choice for you depends on the project characteristics as well as your team's work style and preferences.

Process Area	**ß------ Less formal -- More formal-----------------à**			
Team meetings	Held whenever there is a need, objectives not always specified	Meetings scheduled in advance on demand, specified objectives	**Regularly scheduled meetings, initial (flexible) agenda, decisions noted.**	Meetings with fixed agenda, strict time limits, meeting minutes.
Meeting format	**Free-form discussions**	Focused orderly discussions, no moderator	Meeting facilitator to keep discussions on track	Time limits on discussions, defined decision-making procedures
Version Control	**No version control. Everyone works off a single copy of the source available to all team members.**	**Use of version control tool. Individual workspaces. Checkout / update / commit from repository with change conflict resolution.**	Version control tool as before, but with tagging of significant versions / milestones for fallback and recovery.	Upfront agreement on who is allowed to modify what files & when. Key product configuration items identified upfront.
Quality management	Team members individually try to produce quality	**Some coding standards, informal reviews ("look over others' code when**	Defined coding standards, some reviews/inspections, checklists, tracking of problems to closure.	Inspections/reviews of all items. Formal release checklists with signoffs. Consistent data gathering and

	outputs	**asked”**)		analysis.
Activity planning	Team members identify tasks as they go along and get them done.	**Weekly meetings where tasks are identified and allocated to people.**	Upfront task list created, updated when requirements change. Effort estimation and effort tracking.	Upfront planning and task allocation using effort estimates and task dependencies. Periodic revision of estimates based on experience.
Activity tracking	**Progress discussed informally.**	Use of tracking tool: Weekly meetings discuss progress, plan for next week.	**Use of tracking tool: Progress tracked using tracking tool on every meeting.**	Use of tracking tool: Progress tracked against plan and at every step of the way. Re-planning when progress and plan diverge.
Integration and build consistency (coordinate changes to get consistent versions for builds)	**Team members get together informally, integrate code and debug it. Informal coordination to ensure consistent builds.**	Integration responsibility allocated. Integration approach chosen at implementation time. Coordination of changes during team meetings.	Defined strategy for integration. Activity planning done with integration in mind.	Predefined integration sequence and approach, taken into account during planning phase itself.
Testing	**Team members test their own code in their own way.**	**Test cases defined and run before each release.**	Test cases chosen carefully to ensure coverage. Unit testing by developer, separate integration and system testing.	Formal test plan, coverage analysis. Formal test tracking and bug fix tracking. Regression testing whenever changes are made.
Risk management	Reactive risk management (“handle it when it	**Informal proactive risk management (“take some**	Risk identified upfront, mitigated where possible.	Formal identification and prioritization of risks. Periodic

	occurs”)	**preventive steps”)**		review and tracking. Identified owner for each risk.
Requirements Management	Make changes to requirements as needed	Change notification: Review requirement changes with customer, notify team	**Change control: Analyze requirement changes for technical and schedule impacts. Decide whether to do now or defer.**	Maintain traceability information and versioning information for requirements. Ensure propagation of requirements changes through project documents

For each process area above, discuss the merits as a team and come to a consensus by selecting the level of process formality the team will use. Indicate the team’s selections by bolding, or otherwise highlighting the appropriate cell from **each row**. Do so by changing the cell text to **bold** and changing the **background color to yellow**.

NOTE: There is no one “right” answer for this question – an argument can be made for almost all of the 4^{10} possible combinations (though combinations where some processes are very informal and others are very formal tend to be harder to justify). NOTE: You will not be evaluated on this deliverable, the intent is for your team to put some thought into the selections and help guide the way you will perform.

For each of the ten areas, give a brief two to four sentence rationale as for why you selected the process you did. Couch your answer in terms of why you *didn’t* select one of the adjacent processes.

Process Area	Rationale
Team meetings	We want to be agile in our meeting times, lengths and locations. Meetings with fixed agendas may make it difficult when outside circumstances are influencing time constraints. We do not want to meet only when needed because we feel that meetings are the best time to communicate with one another to determine any issues anyone is having along with any successes they have had.
Meeting format	An informal, free-form discussion is the best way to allow all team members to make their voices heard and to help agree on a plan. While we plan for these discussions to stay on the topics of the meeting’s agenda, too much structure in this area will allow little room for flexibility in planning and strategizing.

Version Control	We chose to use version control because it’s an important tool that will make integration of code much easier. We simply do not want to ignore version control because without it, we feel the project would become chaotic and code would be lost. We don’t want to be strict on who can modify files and when because we want the whole team to be able to have access to all versions in case of some emergency.
Quality management	We will be following conventional coding standards and review team members’ code if necessary. More formal ways would require more effort than we can put in hence we will be choosing more flexible way to follow coding standards. Allowing team members to individually try and produce quality code will not maintain the consistency throughout the project.
Activity planning	It is important for all team members to be aware of what activities are being worked on currently and what needs to be done in the future. Trello has already proven to be a great tool for this. We have been successful in using the tool to track due dates for activities as well as log a history of tasks that have been completed. Using Trello to track our progress for every meeting also helps prepare team members for each meeting.
Activity tracking	We want to keep track of all the progress that is made on every activity. Without tracking progress, we wouldn’t be aware of what things we have done and what still needs to be accomplished. It is also a great way to keep track of due dates.
Integration / build consistency	We want integration to be a team task. This way, each member can be sure that their code is integrated correctly and functions properly when integrated. Allowing a single person to do integration may lead to problems if they do not integrate and debug properly or alter the way someone else’s code works.
Testing	Individual team members will be testing their code along the way and before each release we will run the defined test cases. It is important that the individual team members check their code before committing.
Risk management	We want to make sure to be aware of all the risks and how they may affect us. We do not want to be oblivious to them, as then they will surprise us and will force us to make changes late in production.

Requirements Management	We would be open to any changes to the requirements but we will be analyzing the impacts and then decide the course of action. Changes will not be accepted once the requirements are finalized for a cycle.