15 Principles for High Quality Game Production

By Chris Klimecky

1 - Fun Through Iteration: Fun is the highest priority in our games and requires many iterations to get it right. Finding "The Fun'' and refining it into a satisfying gameplay experience is not a single spec to screen process. Starting with the establishment of a Plan of Record (POR) to point us in a direction, we plan for iteration, estimate for iteration, and scope with the time it will take to iterate in mind, adjusting the POR as we learn more along the way.

2 – Respect Our Player Community – A game has little meaning on its own without its players. We respect our player community, communicate regularly (good or bad news), and develop with our players in mind. This does not mean doing everything they want, but we should take their feedback into consideration, hear as diverse a range of voices and perspectives as we can, measure/analyze player data, and communicate decisions clearly (including a respectful "no"). Good humor and a "personal, human voice", as opposed to a "corporate voice", help to build the relationship and cultivate a two-way street of trust and respect. This includes admitting mistakes and course correcting publically as warranted.

3 - Collaboration is Key: While important in concept, mindset, and normalized behavior, it is also a deliberate part of our process. Meetings that brainstorm and solve problems, document/spec reviews, code reviews, art/audio reviews, monthly lead team project reviews, daily syncs and alignments, sprint plannings and retrospectives...these are just a few examples of the many ways that we connect, give and take constructive feedback, incorporate new ideas, grow together, and enhance each other with intent in order to get the best games out of all of us.

4 - Strive for Alignment: Making games with a team is a significant challenge. We recognize and respect the complexity of our interconnectedness. From the individual to strike teams to interdisciplinary teams to project teams to PM Central Tech/Marketing, etc. affecting each other, we remain aware of the ripple effect and continually strive for alignment in a complex and evolving environment. We only succeed if we all succeed.

5 - Plan for Flexibility: The word "agile" gets thrown around a lot these days, almost to the point of negative reaction due to its overuse and loss of real meaning. We do not adhere to a strict agile methodology, which is actually more constrained than most other production methodologies. Instead, we plan for flexibility, using Sprints to bucket and prioritize tasks for focus, measure progress/velocity, plan for the short-term, and learn/improve our day-to-day efficiency. In the long-term, we plan for systemic iteration, user research/feedback, and development of the "magic" that often can only be found toward the end of a production cycle when all the planned parts have come together.

6 - Creativity Led, Technology Based: Focusing on the player experience is at the heart of our game development. The creative ideas and gameplay that come from the minds and efforts of the creative team are the driving force behind our games. That said, game development is inherently software development, a decidedly engineering and technology-based effort. Support for a continually improving architecture-based engineering effort is paramount to the success of a game studio. Better engineering produces better games by empowering designers and artists with the building blocks to create great player experiences over the long term.

7 - Evolution of Process and Growth Mindset: Change and uncertainty are not a bad thing. In fact, we accept and embrace it - we believe we can get better together. We have process not for process's sake, but in support of the efficiency and well-being of our teams. That means we must evolve our processes as the team and project evolve. A consistent cadence of retrospectives and candid feedback help us learn, keep the good stuff, and improve what isn't working anymore. A growth mindset supports the awareness and flexibility to continually make this journey as a team.

8 - Production With Empathy: We hire smart people. However, even smart people can make mistakes when they are burned out or otherwise not thinking clearly. Good mental health, aside from being good for people in general, is an efficiency issue with regard to its importance for production. We take mental health seriously, reasonably scope for deadlines, collaborate for manageable workloads, and exhibit flexibility when the scope-capacity relationship is not playing out properly. There is no planned team crunch. While individual passion and effort are celebrated, our focus is on efficient use of work time rather than extra work hours.

9 - Transparent Communication: Honest, direct communication is our baseline and our objective is to regularly relay information that reflects our current understanding of reality. We don't play games with estimates/scheduling or guarding information - we aim for transparency and candor at all times, even if it may be uncomfortable. Regardless of venue, we strive for proactive and candid communication that gets to the heart of issues and the information people need to function efficiently and effectively. We constantly reinforce an environment in which speaking truth is safe, regardless of the audience (especially in "speaking truth to power" situations).

10 - Focus = Quality: Whether it's individual focus, team focus, feature focus…all the way up to company focus, we create better games when we pay attention to focus. This can take many different forms such as scoping decisions or release platform decisions or specific team focus days. Focus allows individuals to create flow in their creative process and teams to iterate more in the areas that matter most. Our teams and our games are better with focus.

11 - Cutting is not Failure: It is valuable to be a team of high ambition and we wouldn't want it any other way. But when the reality of a situation shows that we can't deliver on those ambitions, cutting is the right decision. Though it can feel like a failure, it is actually the opposite - we have succeeded in focusing the project and reserving time to iterate in areas that have higher return on our work investment. The game will be better for it.

12 - Secondary Systems (Front-End and Back-End) are Not Third Class Citizens: We take front-end functionality (and UI), credits, accessibility, telemetry, localization, and mod support seriously. Plan for them up front, iterate early and often, and build systems throughout the game that support them. While not primary, they are required, and ignoring them just makes them more difficult to develop later in the process.

13 - QA is an Integral Part of the Team: Quality Assurance is an important part of the game development process, they are key members of the dev team, and QA is a respected, viable career path. From code branch management to BVT, gameplay feedback to tools testing, automation to TRC/TCR compliance and much more, QA is vital to the team's efficiency. Early investment in QA saves engineering time, increases iteration speed, and informs greater understanding of risk on the project.

14 - Great Art, Animation, and Audio Require Great Tools: Tools for workflow, tools for pipelines, and tools for optimization are the support that artists need to get their creations into the game and functioning well. This necessitates consistent and dedicated engineering support both from individuals on game project teams and through a Central Tech group.

15 - Investing for the Long-Term Rewards: We invest so that our games live well beyond their ship date. It doesn’t mean there aren’t constraints or deadlines (no unlimited budgets yet!), but long-term thinking and the support and expense that comes along with it requires specific attention and awareness for decisions throughout development. What choices can we make now that we will thank ourselves for later? How can we architect better for…

a. Short build times when the project gets huge?
b. IT infrastructure?
c. Efficient asset creation and pipelines?
d. Live ops?
e. Mod support?
f. Porting?
g. Archiving that would be understandable by those who might port to other platforms in the future or access for any other reason?
h. Systems that can be moved to Central Tech and used for future projects?
i. Profiling and automation tools?
j. Etc…

These are just a few of the questions we can be asking ourselves that, when decisions are put into practice, will make life easier and development more efficient over the lifespan of a project.