

# Accurate Prediction and other Organizational Myths

by  
Starr Long

# Minimize Pre-Production!

- Contrary to standard organizational methods
- “No plan survives contact with the enemy”
  - Giant, extremely detailed, Design Documents done during preproduction are a waste of time
  - Only do enough pre-production on the game design to build an overall schedule
  - Only do detailed designs in conjunction with the programming team as they are implementing a given system

# Minimize Pre-Production!

- Step 1: Create a concise basic feature list
  - List of features with a short phrase describing it
    - Example: Basic AI: attack, defend, retreat, flock
  - Should also include things your game explicitly **WILL NOT** do
    - Example: No arbitrary item placement
- Step 2: Create short descriptions of each feature (max one page)

# Minimize Pre-Production!

- Step 3: Build out engineering schedule
  - Based on the feature list and one pagers
  - Include “maintenance” time.
    - 50% average over allotted time for unforeseen issues, bug fixing, etc.
    - Maintenance time is the per feature/task cushion.
- Step 4: Build Design and Art schedules based on engineering schedule
  - Make any changes to Tech schedule based on feedback from Art and Design

# Minimize Pre-Production!

- Create milestones and deliverables that have clear overall goals
  - Use meaningful milestone names vs. old definitions of Alpha, Beta, etc.
    - Example: Milestone 2: Walk & Talk: Characters will be able to walk around a game map & talk to other players
  - What is the game play like at the end of the milestone?
    - Example: At the end of milestone 3 the player will be able to create a character and equip weapons.

# Accurate prediction is a myth!

- Budget time per feature, don't allocate time based on design
  - EXAMPLE: Budget 6 weeks for Character Inventory, any features that fall outside that six weeks are cut/postponed
- Prioritize sub-features / sub-systems within each feature / system
  - Minimum required for ship, wish list for ship, etc.
  - Use this prioritization to determine which features get completed within budgeted time

# Accurate prediction is a myth!

- Only do detailed scheduling and milestone descriptions for a given milestone during the preceding milestone
  - Needs & tasks will change as the product progresses so fleshing out details too early just creates rework.
  - Constantly reevaluate your schedule
    - Estimates are valuable for guiding the larger motions of the group, regular analysis of *actual* costs contribute greatly to more accurate prediction in each future phase
    - On TR the Art Director regularly reviews the actual costs of each art asset after it is complete

# Discipline!

- Keep to a reasonable Team size:
  - More than 25-30 on a team is very risky
  - Large teams have trouble communicating and staying in synch
  - With larger teams Managers spend too much time managing people vs. managing the project
  - Start small and bring on team only as needed
- Throwing more bodies at a problem rarely solves it.
  - Balance regular full time with contract/temp resources to better match production spikes

- Don't expect Managers to contribute content
  - This is a slowly dying myth in our industry
  - Don't try to base your schedule on content from managers
  - Tech director will rarely write code, Art Director won't be painting textures
  - Managers will be scheduling, developing technology, directing the team
  - Leaders lead, production resources produce

# Discipline!

- Core hours
  - I know everyone will be available for discussions & meetings at a certain time each day
  - I suggest 9 AM with 8 hours of working time
    - This gives two blocks of time for real work to be done.
  - Time for the industry to grow up
  - Contrary to popular belief getting to work on time AND in the morning does NOT prevent creativity
  - Now that we are getting older more and more of us have families and would like to see them in the evenings.
  - The actual times are irrelevant, consistency is the key.

- NO CRUNCH
  - Extended mandatory overtime NEVER makes a better game
  - On TR we are doing limited overtime (2-3 weeks max) towards specific goals like demos, milestones, etc.
- Tools (Editors, exporters, etc.)
  - Allocate at least 1-2 full time experienced resources just to tools
  - On most products I have worked on this was always lower priority than getting game code working. This is a HUGE mistake

- Art Pipeline Structuring extremely important very early
  - Definition: Getting art into the game
  - Find the right tools (try very hard not to write them yourself)
  - Make sure those tools work well as a long-term, extendable solution
  - Then DO NOT CHANGE IT
  - Resource Management Tools are critical part of the pipeline

# Discipline!

- Maintain constant high level of communication:
  - The entire team should always be aware of the current status of the project
  - Regular reports (daily, weekly, monthly, etc.)
  - Regular meetings
    - Meetings should be as short as possible
    - Meetings should always have stated goal, an agenda, and notes/action items should be taken
  - Internal website with links to current documentation
  - Get out from behind the desk

# Discipline!

- Documentation, Code Comments, etc.
  - The time of the hacker is over.
  - Any programmer could take over any other programmer's work just by looking at documentation and comments
- Features will be cut
  - You will need to do it, get ready

# Play Your Game!

- Stable, Fast, & Fun: In that order
- Weekly Play sessions as soon as possible
  - Make sure team provides feedback for these play sessions & you track that feedback
- Create actual game environment as early as possible
  - Fixing bugs always higher priority than new features.
  - Always have a working version
  - Automated Daily Builds

# Structure!

- Establish a clear hierarchy from the beginning.
  - Make sure everyone knows who to go to for decisions
- Organize by department with leaders of each (art, programming, design)

- Strike Teams
  - Once basic structure of game is complete move to strike teams
    - Retain dept. managers for resource allocation, etc.
  - Strike teams are temporary
    - Reorganize based on needs
  - Goal oriented
    - Weekly demonstrable goals, one large goal
  - Cross discipline
    - At least one member from each department
    - Strikes avoid slogging through process, they are nimble and dynamic, they promote accountability which usually equals results.



# QA & Support: Test Early, Test Often!

- Involve QA & Support from beginning
- Have QA test every milestone deliverable, even if you are developing internally
- Require sign off for all deliverables
- Make details like code comments and documentation required for deliverable sign-off
- Have QA test each daily build
- Give QA promotion control for builds

# Conclusions

- Minimize pre-production
- Budget time vs. attempting to accurately predict
- Establish and maintain a disciplined environment
- Play your game early and often
- Establish and maintain a clear yet flexible team structure
- Test, test, test