

Session Title: From Counter-Strike to Left 4 Dead: Creating Replayable Cooperative Experiences

Format: 60 minute lecture

Audience Level: All

Description:

This session will review the high-level design of Left 4 Dead, how it evolved from Counter-Strike, and the importance of procedural systems such as the "AI Director" in creating replayable and compelling cooperative experiences.

Takeaway:

Lessons learned from Counter-Strike and Left 4 Dead with respect to authoring replayable cooperative game experiences.

Outline:

Brief History of Left 4 Dead

1. Worked on Counter-Strike franchise as Turtle Rock Studios
 1. Counter-Strike Bot (2003)
 2. Counter-Strike:Xbox (November 2003)
 3. CZ (March 2004)
 4. CS:Source (October 2004 -)
2. Discovered early on (2003) that a few Counter-Terrorist players against a team of Terrorist bots armed with knives was a lot of fun
3. After prototyping a few concepts for our first original game, we returned to the "horde rush" game concept and began what was to become Left 4 Dead in early 2005
4. By 2007 the game had started to attract attention. Our 12 person dev team was spread thin, and by the end of the year Valve acquired Turtle Rock Studios, ramping up development resources by an order of magnitude.
5. Valve released Left 4 Dead in November 2008

Some Lessons from Counter-Strike

1. Positive
 1. Teamwork and cooperation
 1. Difficult to succeed alone, cooperation and teamwork directly improve success rate
 2. Random players on the internet will work together if the game mechanics encourage it
 2. Severe penalties can be useful game mechanics
 1. If the player perceives them as "their fault"
 2. If the player can predict/expect them
 3. If they are easily discoverable
 4. If they drive players toward desired core behavior
 3. Natural pacing of CS is "spiky"
 1. Periods of quiet tension punctuated by unpredictable moments of intense combat
 2. Constant combat is fatiguing
 3. Long periods of inactivity are boring

- 4. Unpredictable peaks and valleys of intensity create a powerfully compelling experience
- 4. Emergent game play creates infinite replayability
 - 1. Same scenario, often the same map, yet different and compelling experience each round
- 5. Co-op Against Bot Teams is Viable
 - 1. Players play alone with bot teammates and enemies
 - 2. Players play co-op games with friends against all-bot teams
 - 3. Allows for players of all skill levels to enjoy CS
- 2. Negative
 - 1. Steep learning curve
 - 1. Overwhelming to new players
 - 1. Large list of similar equipment choices
 - 2. Relative effectiveness of equipment is unclear
 - 2. Rules must be learned by trial and error
 - 1. How to gain money
 - 2. How to buy equipment
 - 3. What the goals of each scenario are (ie: bomb defusal)
 - 4. Where the goals of each map are located
 - 3. Unforgiving of minor mistakes
 - 1. BOOM! Headshot.

Left 4 Dead Design

- 1. Build Upon Lessons Learned from CS
- 2. Keep it as simple as it needs to be, but no simpler
 - 1. Clear overall goal (escape)
 - 2. Easily discoverable core gameplay (pick up guns, shoot zombies, work together, don't die)
 - 3. Clear equipment choices
 - 4. Discoverable map routes
- 3. Explicit Cooperation
 - 1. Push CS teamwork to the next level, require explicit cooperation
 - 2. Demand seems to exist for co-op, but very few games actually require it
- 4. Replayability
 - 1. Avoid explicit scripting and designer-placed events
 - 1. Inherently less replayable
 - 2. Labor intensive
 - 3. Traditional scripting is very difficult with more than one player
 - 2. Invest in AI and procedural algorithms
 - 1. Greatly multiplies small team output
 - 2. Inherently replayable

Co-op

- 1. Friends Amplify Fun
- 2. Co-op Increases "Net Fun" of entire group
- 3. You Can't Do Co-op Halfway
 - 1. If players can "win" by going solo and ignoring/sacrificing teammates, they will
 - 2. The challenge is to structure game dynamics such that players perceive cooperation as the best strategy

Replayability

1. The "AI Director" is a collection of systems that create a dramatic, replayable experience.
 1. Procedural population of enemies and items
 2. Player emotional intensity management
 3. Dynamic Music management
2. Robust non-scripted AI actors
 1. Randomized placement requires AI actors to have robust navigation and behaviors

Procedural Population

1. When each map is started
 1. Using the Navigation Mesh, the primary Escape Route is computed through the map
 2. Bosses (Tanks and Witches) are placed on/near the Escape Route
 1. Move down the Escape Route by a randomized distance
 2. Select a nearby nav area
 3. Place a Tank or Witch marker in that area
 4. Repeat until end of Escape Route reached
 3. Randomly distribute wandering zombies throughout the map
 1. Increment "zombie counter" for desired area
2. During the Game
 1. Maintain an "Active Area Set" (AAS) around the Survivor team
 2. As wandering zombies leave the AAS, delete them and increase the zombie counter in the area where they were
 3. As areas enter the AAS, create

Intensity Management

1. Simulate how "stressed" each Survivor player is
 1. When they are injured, increase "intensity" by an appropriate amount
 2. When there are no nearby threats, decay intensity over time
2. Use intensity metric to shape the Survivor team's experience
 1. If any Survivor's intensity reaches maximum, calm things down for a variable duration (ie: a "relax period")
 2. After the "relax period" is over, return to "build up" mode
3. Randomization is important
4. Pacing is modulated, not difficulty
5. How pacing is relaxed
 1. Mobs are not created
 2. Wandering zombies are not created until max Survivor intensity drops below a threshold
 3. Specials are not created (Hunter, Boomer, Smoker)

Dynamic Music Management

1. Musical score is created dynamically for each player
2. Crossfade/blend core music based on Survivor's "intensity"

1. Silence at the right time
2. Music builds as combat becomes intense
3. Serendipity is Good
 1. As long as the "misses" aren't so bad they stand out, the "hits" will be powerfully remembered
4. Music associated with events
 1. Mob rush created
 2. Special infected spawned
 3. Tank approaching
 4. Witch nearby
 5. Player death
 6. Player pounced/choked
 7. Player incapacitated
5. Music associated with the environment
 1. Moving from a confined space into a wide open area (reveal)
 2. Specific areas, such as safe houses
6. Music tied to Director state
 1. Finale stages
 2. Team success/failure

Other Observations

Everybody Knows Real Zombies Don't Run

1. Design decisions are driven directly by gameplay. Mobs of running melee attackers were exciting and fun. The most compelling story was selected that fit our preexisting game mechanics.
2. Precedent - recent zombie movies have explored "fast" zombies
3. Polarizing discussion points can be good community drivers and feed word-of-mouth PR.
 1. Recurring community rants for/against fast zombies
 2. Whether people talk to their friends about awesome "fast" zombies, or complain that we screwed up, they are still talking, and Talking is Good.
 3. "The AWP suz0rz"
4. Don't try to create controversy, but don't avoid valuable game mechanics because of it, either.

Players Personify the Director

1. Serendipitous events considered to be the work of the Director

Players of varying skill levels work in L4D

1. Players who dont have traditional FPS skills are still useful to the team
 1. Healing
 2. Rescuing from Incapacitation
2. L4D doesn't rely on precise aim

Women and Zombies

1. We have seen more women play Left 4 Dead for longer amounts of time than we have with Counter-Strike and Team Fortress 2.
2. A goal other than "domination" is appealing
3. Teamwork and cooperation is more valuable than twitch aiming and obscure game knowledge