More on the language of game design and research.

In 1999 a game designer Doug Church (Looking Glass Studios, Eidos Interactive and Valve) published a paper suggesting a clear and well-defined language for videogame design. Unlike other technical disciplines, videogames have yet to formalize a shared language. Communication is important in all disciplines, but essentially videogames came down to being described as being 'fun' or 'not fun'. There is no singular 'language' that incorporated all of the elements of the design process. Shared language is a critical tool for analysis and understanding, and creates cognitive shortcuts when talking to other designers.

In my book I focus on one model for defining player motivations and behaviors. This paper examines two additional industry-used models. Formal Abstract Design Tools (FADT) proposed by Doug Church and Mechanics, Dynamics and Aesthetics (MDA) proposed by Robin Hunicke, Marc LeBlanc and Robert Zubeck and based on Church's work. To understand and study videogame requires a formal language (define the nuts and bolts) as well as an abstract one (describes the concepts and aesthetics) that is relevant to the discipline.

Formal Abstract Design Tools (FADT).

When designers want to talk about a 'cool' element or well-designed piece of a game it seems obvious to look at an improved artificial intelligence (A.I.) algorithm or a sleeker motion capture application and agree it's a step forward for the gaming industry. There are tangible measurable outcomes from these advances. What is much harder to track is the interconnectedness of game design across genres and platforms and then be able to analyze the effect of a small evolution in a system in one game and maps its influence on other games.

Formal elements in other disciplines such as engineering, are precise and can be explained and understood by other engineers (an example of a formal element would be *Player Reward Systems*, designers understand what these are) but the abstract side of the language is harder to grasp because it is based on the underlying ideas and concepts of the game's mechanics (for example an idea for an incremental system that makes the player more powerful based on the player's combat style which is exhibited through a more powerful gun or magical sword, is an abstract concept because it is not fixed like a '+2 Damage Attack Sword'). This may seem confusing, but this is precisely the problem game designers have, they are building fantasy worlds which are markedly different from designers building simulations of our own world. Even in real-world scenarios or simulations there issues exist because videogames are not mirrors to real life, they are systems designed primarily for entertainment and are designed with interaction and a player in mind.

The Formal principles of Videogame Design.

This is where the theory of game design and how it's communicated becomes a central theme. We all know what a videogame is, until we actually attempt to describe what *all* videogames are. A platformer game is not the same as a first-person-shooter which is not the same as a sports game and so on. Doug Church's *Formal Abstract Design Tools* have been developed to create a common language for designers and developers based around what all videogames have in common and what defines the medium:

- Player Intention.
- Perceivable Consequences of interaction.
- Story/Narrative.

Player intention is realized by the player being invited to plan and play a path through a game and to become invested in exploring and conquering the world they're in (the magic circle as discussed in the book). Intention is the game talking to you as the player and giving you options. You understand these options in context of the game and implement a plan of action based on what you understand of the game world and its limitations. For example the player in *Batman Arkham Origins* (Rocksteady Studios 2012) comes upon a game area containing a large group of foes and begins a fight sequence but is quickly overpowered and fails. The game has told the player that just wading in and performing actions in that way results in a fail state. The intention of the player is to then plan how best to complete the task using the fighting techniques they have picked up so far, and to try different approaches based on the feedback they receive from the game (perhaps using combination attacks, or physical elements from the environment). As a game designer it is important that the player understands why they succeeded or failed in any encounter. In *Arkham Origins* it's easier to defeat foes if the player 'chains' attacks together and there is specific on-screen feedback that supports this premise, it becomes obvious to a player that this tactic will ensure a more positive outcome if they try this the next time.

Intention is an abstract principle, because there's no obvious correlation between chaining punches and kicks together in a brawler-style game and the mechanics of a football or soccer videogame. The important part for the player is that they understand that chaining attacks meant they could create a win state. A football or soccer game could use a very similar mechanic to chain moves, dodges or passes together to create a better chance of scoring a goal. If the player fails to score they will analyze their game play and know where they went wrong, perhaps in timing or in missing a button press. In this regard two completely different games use *player intention*.

Perceivable Consequences is the game world reacting to you as a player. This tool applies across many genres of games but is most commonly used in Role Playing Games (RPGs). A perceived consequence could be as simple as the unlocking of a secret area by hitting a switch, or feedback from the game on completion of an action (unlock chest, receive gold). Consequences, like intentions, are focused on feedback that is consistent within the game world. The question the developer asks when applying perceivable consequences is 'What would be it be reasonable for the player to expect when faced with this scenario?'

Early videogames traps would often appear from randomly or from anywhere and contain sudden death (Battletoads, 1991, Rare and Teenage Mutant Ninja Turtles, 1989, Konami). This was often due to bad game design of harked back to the arcade game model which required that the player insert more money to continue as a business model. Perceived consequences in game design can be used intentionally or unintentionally as a way of 'punishing' players and as such needs to be implemented properly. Today the developer should telegraph the trap in some way so that the player can build an 'intention model' and begin to understand the perceived consequences of interacting with the trap. So even if there is sudden death, the player should understand that there is a reason they die, and they need to find or use other game elements (a rope, a hidden door) to route around the trap or negate it in some way. The perceivable consequence is that the player knows there is a trap, and can die from it, but then understands that they are performing an incorrect action so they have to find a way around it and can begin to explore their options within that scenario. For example in Tomb Raider 3 (Eidos Interactive, 1998) there is a boulder trap, if you run in a direct straight line Lara will always get crushed, it's hard to see but off to each side is another corridor, stepping left or right avoids the boulder. The player knows their actions are incorrect because of the fail state, so the game teaches them that there must be another way, and there is. Enabling the player to make a decision based on the knowledge they have of the game thus far (intention), builds an intuitive model of the game-space based on feedback from their interactions (perceived consequences). Consistency across these two principles creates a positive experience, but the caveat is that this can create tensions with story and narrative structures.

The Intention/Consequences model works well in action and platform style games. When applied to open-world or RPG style games it does not seem to apply as well. It does, but in a more nuanced manner. When in an open-world or fantasy game players know that they have to interact with everything, pick up everything and talk to everyone to advance the story or objective. The player is trying to discover the intentions of the designer and move their game forward (Intention/Consequence). It may be that the discovery of a certain book enables the player to access a certain area, and so on. The most obvious form of Story/Narrative in the FADT model is the 'on-rails' narrative structure. In games where there is really no choice or branching narrative (for example the recent Lara Croft reboots) it is the combat system that

contains the main intention/consequence. As a player you have no control over the narrative, but you can control the combat and how you approach survival or death.

Sports games merge all three aspects of the FADT because there is no narrative other than the one the player is creating simply by playing. What is different is how the intention versus perceived consequences operate. There is a system in sports games is bases on statistics and algorithms, so the same actions can have different results. The pass may work five times, but not on the sixth. This is understood by the player and the consequences are understood to have been impacted by the game's AI or other players. The FADT model does not ignore story, but is it far more focused on the systems that the player interfaces with over narratives in order to better understand how the player would think their way through the games problems.

The Mechanics, Dynamics and Aesthetics model (MDA).

The other side of creating a language for videogames is in creating a language with wider appeal for critics, academics and creative people related to the videogame industry. The MDA model expounds on the formal/abstract principles by including aspects of the building blocks of game design and then applying them across disciplines.

The MDA model is focused on achieving and understanding the *experience* of gameplay. The formal/abstract principles are focused on building the game, the mechanics/dynamics/aesthetics look at the game being played. The model's creators understand that experiences cannot be designed but instead are emergent, developing out of the rules and mechanics of the game, and that's an important definition. For example, when playing The Last of Us (Naughty Dog, 2013), no player can pinpoint which mechanic it is that makes them want to be more (or less) protective over Ellie. Which mechanic gives them a sense of panic or despair when wandering though the ruined world. If you were to talk to any other players of a game you really loved, the chances are they would have subtle or perhaps profoundly different experiences to your own. Even though the game is exactly the same for each player mechanically and narratively, the experiences can be very different. Within the MDA model this is defined as Second Order Design.

Second-Order Design:

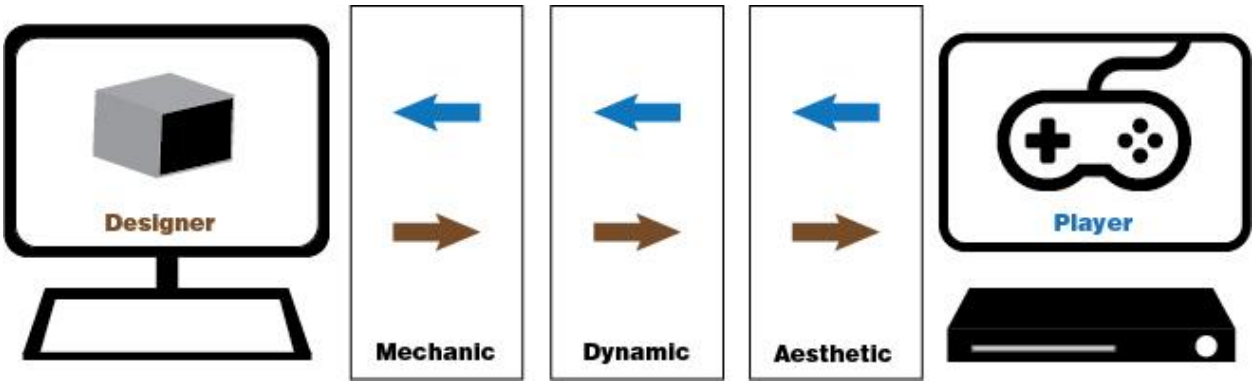
A game enables and creates the experience for the player, but does not fully control it. This is emergent gameplay and is known in the discipline as a second-order design problem. In certain games such as World of Warcraft (Blizzard Entertainment, 2004), Eve Online (CCP Games, 2003) or other MMOs the games designers do not define the path towards the end solution (players have to save the world, and only in this way), but instead create a game that through its mechanics, dynamics and aesthetics will enable the player to create their own solution. This

may be a solution which the designer does not have any control over that solution (for example the player could use an unconventional gambit to achieve a result) and that is fine.

Games are very complex and essentially the designer builds a game with rules and scope and then lets the player loose in that environment. Game design is not focused on the *idea* of a game it's about *implementing* that idea and understanding that a large portion of any positive experience of a game (Dynamics and Aesthetics) is decided by the player and can be very subjective. As we discussed before, this is *emergent* gameplay. In early FPS multiplayer games if players always re-spawned in the same area, players realized that they could stay here and just attack anyone else. It gave rise to the play dynamic of 'camping'. This is an unintended consequence of solving the problem of 'how players re-enter the game once they die', no one on the design team foresaw the evolution of camping because no one had played that way in development testing. The fix was to randomize the spawn points, but this was in reaction to the players using the game and coming up with their own strategies or second order design based upon the player's actions and experiences.

Layers of Experience based upon play.

FADTs and MDA models teach us to embrace two viewpoints, the designer who creates the mechanics and then goes on to develop the aesthetic (the A in MDA); and then the player who comes at the game from the opposite direction. The player begins with the aesthetic and effectively works backwards towards the mechanic. The aesthetic is the first impression of the game (the 'On-boarding' building block) because it's immediate and easy for the player to grasp. Within seconds of engaging with a game, players will create opinions on the game and



begin to describe their relationship with it based on their internalization of the game's 'feel'.

As a player moves on past the Aesthetic they encounter the Dynamic and their experience is more subtly informed. The Dynamics (or interactions) within the game space, such as how the

character moves, if the multiplayer combat seems fair and balanced and so on. This is the point at which the player is creating a deeper relationship with the game based on their time in the game (experience). However, the perception of the game can be positive or negative based on the Dynamic, if it's forcing the player to play in a specific way, or if the game reacts against a player's action because it's too strictly designed, a positive dynamic can be disrupted. The analogy is, that with the MDA model the designer must design for the player, not for the designer, because what one designers thinks is 'fun' another player may not. The role of the MDA is to keep the designer on track thinking about and designing for the player.

Conclusion:

FADTs and MDA models focus on language, analysis and start us thinking about the inter-connectedness between players, designers and mechanics. How we communicate about games is important, it helps to better define the medium with every small step. Better models for communication means better games in the same way that a shared language in engineering means better bridges. For a designer taking on-board these different models does not mean applying every aspect, it's being aware of the intricacies of these relationships between games and their players. As a designer you can being to understand that a change to the Mechanic or the Dynamic alters the Aesthetic in a meaningful way and this can color the player's experience positively or negatively. There is a lot that cannot be controlled. You need to continually test and iterate your game and apply the communication models that make the most sense to you, to get the most positive reactions from players. These models are an aid in keeping your development process on track and give you the ability to amylase and think critically about the functionality of your game rather than just assuming if you find it fun, others will too.