RPG 플레이 밸런스를 위한 리스크 분석 적용
(Applying Risk Analysis To Play-Balance RPGs)

아담 카펜터(Adam Carpenter)

가마수트라 등록일(2003. 06. 11)

http://www.gamasutra.com/view/feature/131252/applying_risk_analysis_to_.php

플레이어가 게임 잡지가 최근에 출시한 게임에 "밸런스가 잘 맞추어졌다"고 어느 정도나 평가하고 있는가? 또한 얼마나 오랫동안 게임을 하고 있다고 말할 수 있는가? 실시간 전략 게임(RTS)와 롤플레잉(RPG) 게임 장르에서, 첫번째 질문에 대한 대답은 거의 할 수 없다. MMORPG의 경우에, 두 번째 질문에 대한 대답은 그래야 함에도 불구하고 그렇게 많지는 않다. 좋은 게임 밸런스는 여러 면에서, 게임 디자인의 성배이다. 회사는 엄청난 시간과 자원을 게임 밸런스를 맞추기 위해서 투자하고, 그래서 그들은 너무 어렵지도 너무 쉽지도 않은 게임을 만들고자 한다. 이러한 노력은 기업의 자원을 유의미하게 이용할 수 있도록 하는데, 디자이너와 엔지니어가 부가적인 기능을 부여함으로써 밸런스 문제를 해결하고자 노력하게 된다. 대신에, 이들은 똑 같은 방정식이나 코드 라인을 분석하는데 귀중한 시간을 소비하게 되는데, 가령 어떤 캐릭터 클래스가 계속해서 다른 것보다 훨씬 퍼포먼스가 좋을 때 이렇게 된다.

밸런스가 맞추어지지 않은 게임은 또한 다른 방식으로 개발을 괴롭힌다. 밸런스가 맞지 않은 게임은 플레이어를 좌절시키고 부정적인 여론과 리뷰를 발생시킨다. 이 두 상황 모두 게임 판매나 월 결제에 부정적인 영향을 주는데, 이로 인해 퍼블리셔와 개발사의 현금 흐름을 나쁘게 하고, 기업이 가용할 자원과 자본을 제한시킨다. 이러한 자원이 없다면, 기능을 향상시키는 능력과 현존하는 게임의 특징을 제한하게 된다. 또한, 새로운 게임을 위한 가용한 자본이 줄어든다.

AAA 게임 타이틀을 개발하는 것은 수 백만 달러가 소요되고 비용은 계속해서 오른다. 평균적으로 타이틀 하나를 출시하는데 3백만 달러가 드는데, MMORPG 개발은 1,000만 달러를 쉽게 초과한다. 이것은 모두 개발 비용에 불과한 것이고 출시 이후의 경비는 포함되지 않는다. 디자이너와 엔지니어는 게임의 밸런스를 맞추고 버그를 제거하는 등의 개발 이후에도 일을 하게 된다. 일부 게임의 경우에, 출시 이후 비용은 상당할 수 있다. 이러한 출시 이후의 비용은

비효율적인 밸런스 때문인데, 이것은 최초의 디자인이나 알파, 베타 단계에서 발생된다. 다행스럽게도, 적절한 위험 분석 기술을 이용하며, 두 가지 면에서 도움을 받을 수 있다. 개발 시간을 절약할 수 있으며, 이론 인해 비용을 줄이고, 긍정적인 구두 마케팅으로 판매와 구독자를 증가시킬 수 있다.

확률과 리스크 분석

대부분의 기본적인 수준에서, RPG는 수많은 숫자와 방정식의 집합체이다. 이러한 전체의 게임은 스프레드 시트 프로그램에 의해서만 개발될 수 있는데, Microsoft Excel과 Palisade의 @Risk와 같은 플러그 인이 그것이다. @Risk는 Microsoft Excel에 부가할 수 있는 상업용이다. 이 프로그램은 디자이너가 불확실한 것을 @Risk 기능에 적합한 가치로 바꿀 수 있게 해 준다. 이렇게 함으로써, 전체 결과의 스펙트럼이 관찰될 수 있다. @Risk와 Microsoft Excel의 이용을 통해서 중요한 게임 시스템을 모델링하고 분석할 수 있다.

물론, 하나의 스프레드시트와 통계 패키지로 만들어진 게임은 RPG가 플레이어에게 가져다 주는 의미, 콘텍스트 및 감정이 부족할 수 있다. 그러나, 이러한 게임은 플레이어가 온라인 게임 내에서 개입하고 발전해 나갈 수 있는 똑 같은 기본적인 시스템을 포함하고 있다. 전투 알고리즘에 대해 면밀히 살펴 본다면, 시스템은 어떤 상황에 대해서 좋은 결과를 결정하기 위해서 수 천번의 시뮬레이션을 실시하고 빠르게 모델링 하여 개발될 수 있다. 이러한 결과를 분석함으로써, 밸런스가 진정으로 잘 맞추어진 게임이 만들어질 수 있다.

리스트 분석 기술은 기업이 적용하고 있는 것을 상관하지 않고 이루어지는데, 바람직한 결과는 꽤 유사하다. 석유 회사가 미래의 고정 자산의 활용을 예견하려고 하듯이, 게임 회사도 주어진 게임 상황의 미래 결과를 예측하고자 노력한다. 석유 회사에서, 데이터 모델은 활용가능한 정보를 관리인에게 제공해 주어 그들이 비용/편익 결정을 할 수 있게 해 준다. 이와 비슷하게, 게임의 전투 모델은 디자이너가 적당한 밸런스가 달성 될 수 있도록 종족, 클래스 및 그룹을 조정할 필요가 있는지 아닌지를 보고 결과를 예측할 수 있도록 해 준다. 이 두 경우 모두, 리스크 분석 기술은 불확실성을 줄이고 도달할 수 있는 잠재적인 가치를 극대화 할 수 있게 해 준다.

이 글이 RPG 게임 시스템에 주안점을 두고 있지만, 리스크 분석의 적용은 모든 장르의 게임 내에서 플레이 밸런스 게임에 적용될 수 있다. 실시간 전략 게임은 특히 우수한 후보자가 되는데, 테스트 되어야 할 전투 시나리오가 다양하기 때문이다. 리스트 분석 모델링은 상대적으로 플랫포머와 사이드 스크롤과 같이 상대적을 간단한 게임에는 적용하지 않는데, 이런 게임과의 플레이어 인터랙션이 일반적으로 가치있는 모델링을 포함하고 있지 않기 때문이다. 이런 게임은 디자이너가 간단하게 시행 착오를 거쳐서 밸런스를 잘 잡을 수 있다.

이 글의 시작

(편집자 주 : 이 부문은 저자의 요청에 의해 2003년 6월 16일에 수정되었음)

이 글의 처음은 *Asheron's Call 2* (*AC2*)의 베타와 출시 단계를 관찰한 후에 시작되었다. 게임의 모조-클래스-기반 시스템과 관련된 기술을 리뷰잉하면서, 게임 시스템의 밸런스에 있어서의 불일치성이 관찰되었다. *AC2*의 프로그램 매니저인 Matthew Ford와 대면하면서, *AC2*의 개발자가 MMORPG 개발에 일반적인 수많은 분석 기술을 이용하여 게임 잴런스를 조정했다는 것을 알게 되었다. *AC2* 뿐만 아니라 Ford의 어떠한 다른 MMORPG도 이 글에 설명된 것과 같은 리스크 분석 방법을 사용하지 않았다. 나는 이 점이 매우 흥미로웠다; 이 분석 방법은 많은 소매 상품에서 발견되는 특수화, 클래스 및 영역 불균형의 정도를 설명하고 있다.

Ford와의 서신에서, 우리는 리스크 분석과 실질적인 MMORPG로의 적용을 위한 콘셉을 논의하였다. 우리는 또한 MMORPG 시스템을 테스트하고 모델링함으로써 얻을 수 있는 예측가능한 이익을 실험했다. Ford는 어떠한 MMORPG에라도 이 리스크 분석이 "전통적인 수학적 조절 방법에 귀중한 상대"라는데 동의했다. 그는 특히 게임 밸런스 가치에 관련되지 않은 변화처럼 보이는 것에 의해 기대하지 않은 결과가 만들어 질 수 있다는 것에 가치를 부여했다. Ford는 또한 이 방법이 개발 후기의 게임 밸런스를 정제하는 것 뿐만 아니라, 게임 디자인 초기 단계에서 실험적인 게임 다이나믹스를 반복하는데 좋은 방법이 될 수 있다는 것에 동의했다.

의미있는 게임 시스템

RPG를 디자인 할 때, 수 많은 시스템이 개발되어지고 균형을 맞추어야 한다. 수 천번 게임 진행중에 언급되는 능력, 스킬, 스펠의 조그마한 차이가 실제로 게임에 적용되어 질 때 수용될 수 없는 결과가 초래된다. 일부의 시스템은 RPG에서 포함되어야 하는 밸런스를 필요로 한다 :

- 플레이어 캐릭터 종족
- 플레이어 캐릭터 클래스
- 플레이어 대 환경 충돌 (PvE)
- 플레이어 대 플레이어 충돌 (PvP)
- 플레이어 진보 및 스킬 향상 비율
- 스킬을 증가시키는데 필요한 비용과 시간
- 자원 제약

이 리스트는 성공적인 RPG에서 밸런스가 맞추어져야 하는 시스템의 일부이다. 일을 좀더 복잡하게 하는 경우라면, 게임 시스템이 항상 별개의 요소가 아니기 때문에, 게임 시스템이 부지불식간에 다른 것으로 불균형을 이루게 되도록 노력하게 될 수도 있다. 대부분의 시스템은 밀접한 관련성이 있고 플레이어 종족이나 계급과 같은 기본적인 단계에서 이루어지는 조정은 게임의 모든 면에 영향을 미칠 잠재적 가능성을 가지고 있다.

모델 개발

리스크-분석 기술을 게임 밸런싱에 적용하기 위해서, 기본-케이스 시나리오가 필요하다. 전투 시스템의 경우에, 기본 케이스는 이전에 결정된 레벨/클래스/종족 등의 전형적인 플레이어 캐릭터를 나타낸다. 평균 플레이어 캐릭터가 시뮬레이션 될 수 있도록 하기 위해 데이터를 수집하는데 주의가 필요하다. 밸런스가 게임을 플레이 할 수 있을 만한 단계 이전에 조절하려고 한다면, 디자이너는 개인적인 경험과 그들이 완성된 게임에서 정확하게 보고자 하는 것을 반영한 기본 케이스를 개발하기 위한 미래의 기대를 모두 이용해야만 한다.

인풋 파라미터의 조건

일단 베이스 케이스가 결정되면, 모든 상수, 변수 및 방정식이 시스템에 포함되어야 하며, 필요한 조건은 다음과 같다 :

상수. RPG에서, 상수는 모델에서 변하지 않는 가치를 나타낸다. 상수의 예는 캐릭터 레벨, NPC 타이머 및 최초 종족 통계가 될 것이다. 게임이 밸런스를 맞추고 있을 때 상수가 조절될 것이다. 그러나 어떠한 확률 분포도 이들에게는 적용되지 않을 것이다.

변수. 이것은 플레이어 간에 달라 질 수 있는 게임 내 가치를 말한다. 이것들은 스킬 포인트, 같은 NPC 몬스터의 복제 경우 및 크래프트 자질 분포의 할당을 포함한다. 변수는 확률 분포 기능을 이용하여 고려된다. 노멀, 로그 노멀, 카이 스퀘어 및 많은 다른 분포가 게임내에서 랜덤으로 요소가 나타나도록 하는데 이용될 수 있다.

방정식. 이것은 행위의 성공과 실패를 결정하고, 보호 스킬과 블로킹 퍼센트 간의 수학적 관계를 설정하거나 치거나 피해를 주는 우선적인 통계의 효과를 결정짓는 계산법이다.

모델을 디자인할 때, 주의사항이 있다 : 모델은 데이터 만큼이나 좋아야 한다. 부정확한 데이터나 방정식이 모델에서 대체될 때, 모델이 부정확하거나 관련성이 없을 것이라는 사실에 기인하여 결과가 도출된다. “쓰레기 투입구, 쓰레기 배출구”는 상투적인 격언이다. 그러므로 플레이어의 성향과 최소/최대 잠재성을 이해하는 것 만큼 게임 시스템에서 가용한 상태의 범위를 이해하는 것이 중요하다.

기본적인 플레이어 캐릭터와 기본적인 몬스터 사이의 성공적인 개발을 위한 모델링을 하고 있다면, 다음의 각 사항을 명심할 필요가 있다 :

모델 필수조건 :

우선적인 통계

헬스 토탈

마나 토탈

갑옷 형태

갑옷 수준

갑옷 흡수

무기 형태

무기 데미지

공격/방어 보너스

공격/방어 벌점

피사기/치기/막기/때리기 퍼센트

위의 것은 일부에 불과하다. 게임 내 시스템이 위와 같은 특징에 영향을 받거나 다른 요인에 의해 영향을 맏는 다면, 모델은 이러한 요소에 대해서 인풋을 필요로 하게 될 것이다. 간단한 게임은 의미있는 결과를 도출하는데 몇 가지의 인풋을 필요로 한다. 반대의 경우도 마찬가지이다. 복잡한 게임은 모델을 개발할 때 좀더 많은 노력과 조종을 필요로 한다.

좀더 정확한 모델을 만들기 위해서, 위의 필수요소의 대부분은 변수의 형태로 모델에 투입되어야 한다. 변수의 이용은 상수와 반대로, 관찰되어져야만 한다. 예를 들어, 만약 플레이어가 그들의 X, Y가치 사이의 우선적인 통계를 다양화 할 수 있다면, 모델은 이를 설명해야만 한다. 모델의 요인 변화 때문에 디자이너는 상수 평균 가치에 의해 제공된 결과를 제한할 수 없게 될 것이다.

모든 상수, 변수 및 방정식이 정비되고 스트레드시트에 인풋 값을 넣게 되면, 이제는 시뮬레이션을 실시할 때이다. 시행이 더 많이 될수록, 여러분이 갖게 될 분포는 좀더 균형이 맞을 것이다. 그러므로 여러분은 좀더 정확한 결과를 가지게 될 것이다. 그러나 예외적으로 복잡한 모델의 경우라면, 시뮬레이션 수행에 좀더 시간을 들여야 한다. 일반적으로, 5000회를 반복하는 것이 적당하다. 이것은 0.0002%의 효과를 보여주는데 그치는 정도이다. 만약 고려하지 못한 결과가 관찰된다면, 시뮬레이션 반복 회수는 결과 곡선을 부드럽게 만들기 위해서 증가되어야 한다.

디자인 객관성과 "만약 – 한다면" 시나리오 결과 비교

시뮬레이션이 완료되었다면, 결과가 분석되어야 한다. 리스크 분석 소프트웨어의 최고 장점 중의 하나는 특정한 결과를 위한 누적 확률 그래프를 볼 수 있다는 것이다. 누적 확률 그래프는 결과물 Y가 적어도 X퍼센트의 횟수로 일어날 것이라는 퍼센트를 보여준다. 기본 플레이어 캐릭터와 기본 NPC 몬스터 간의 전투의 경우에 있어서, 디자이너는 각 회당 히트 포인트 변화를 관찰할 수 있다. 그림 1은 샘플 누적 확률 분포를 보여준다.

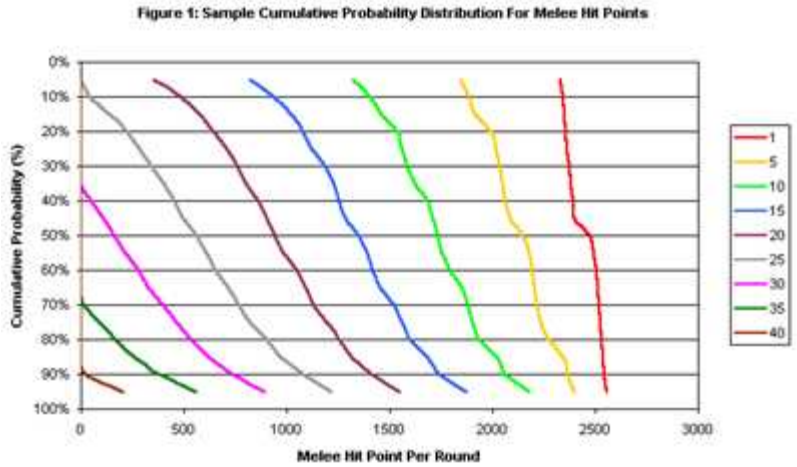

그림 1

각 회마다 죽을 가능성을 표로 나타낸 것은 디자이너가 회당 기준점으로 전투의 사망률을 관찰할 수 있도록 해 준다 (그림 2 참조). 자연스럽게, 가지고 있는 가치가 1이라고 할 때 죽을 확률을 빼면, 생존가능성이 도출된다 (예 : 1.0-0.4=0.4 또는 40%의 생존 가능성). (대부준의 게임 전투가 그러하듯이) 죽을 때 까지 싸운다면, 플레이어 캐릭터의 생존은 적이 먼저 죽어야 가능하다. PC 대 NPC 전투의 경우에, NPC 죽음의 가능성과 PC 죽음 가능성 1을 빼면, 승리 또는 실패를 볼 수 있다. 이러한 두 선이 교차하는 지점은 전투의 결과를 기대해도 좋을 지점이다. 만약, NPC 죽음과 플레이어 죽음의 가능성을 포함하고 있는 지점에서, 25 라운드에서 60% 지점에서 선이 교차한다면, 플레이어 캐릭터는 승리할 가능성이 60%라는 것이다. 이 교차점은 전투 시뮬레이션의 바람직한 결과물로 비교될 수 있다. 바람직한 결과물은 아래와 같을 것이다 :

· 특수한 스킬이나 스타일을 사용하지 않고 같은 레벨의 몬스터를 물리칠 확률이 75%
· 최적의 레벨 스킬과 능력치를 이용한 같은 레벨의 몬스터를 물리칠 확률 95%
· 스킬이 최고치에 도달하기 이전에 평균 60시간의 시간이 필요함
· 최적의 스킬/스타일을 이용하여 다른 클래스를 물리칠 확률 40%

만약 바람직한 결과와 시뮬레이션 결과가 동일하다면, 시스템은 균형이 잘 잡혀있고 디자이너는 다음 시뮬레이션을 나아갈 수 있다. 만약 시뮬레이션 결과와 바람직한 결과물이 동일하지 않다면, 시스템의 일부에 수정을 가할 필요가 있다.

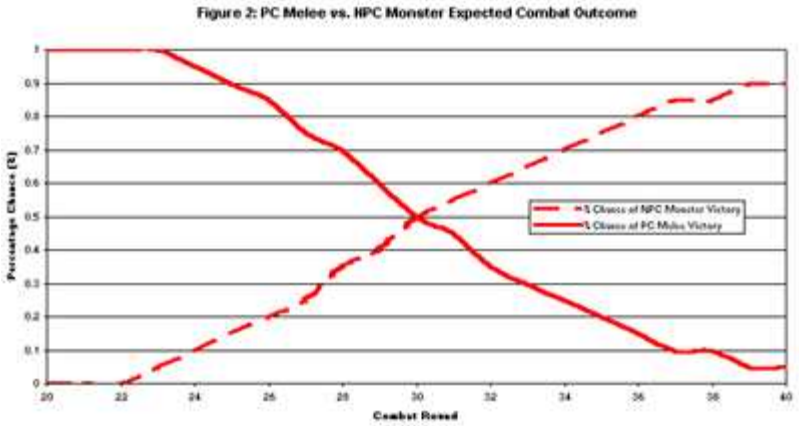

그림 2

이런 식으로 전투 시스템 밸런스를 맞추고자 할 때, 절차는 경제학에서 사용되는 수요-공급 곡선을 연상시킨다. 수요-공급 곡선에서, 특정한 요소가 다양한 방향으로 곡선을 이동시킨다. 승리와 패배 공선에서도 똑같이 적용된다. 다양한 상수와 방정식에 의해서 하나 또는 두개의 곡선이 수평적으로 이동할 가능성이 있다. 바람직한 결과물로 교차가 발생할 때까지 인풋이 조절되어야 하고, 게임 시스템은 균형을 이룰 수 있다.

효과적인 모델의 이점은 특정한 클래스와 접촉자의 밸런스를 맞출 수 있도록 해 줄 뿐만 아니라, 제약없는 "만약에" 시나리오를 탐구할 수 있도록 해 준다는 것이다. 가령, 플레이어 재주가 "치는" 행위, 피해를 입히는 행위 및 막기 가능성에 영향을 준다고 가정해 보자. 만약 디자이너가 플레이어 손상을 클래스의 재주를 증가시킴으로써 증가시키고자 한다면, 그는 "치는" 행위와 막기 퍼센트에 미치는 영향에 관해서도 고민을 해야만 한다. 만약 디자이너가 종족의 기본 재능을 증가시키고자 한다면, 그는 이 변화가 모든 다른 클래스에 미치는 영향력에 대해서 고민할 필요가 있다. 그러나 만약 모델이 확고히 개발되었다면, 클래스의 재능을 증가시키거나 종족 재능을 손보는 것은 간단한 작업이 된다. 수정한 다음 시뮬레이션을 다시 하고, 재능 증가가 다른 계급, 종족 또는 NPC와 관련된 밸런스를 훼손하는지 아닌지를 결정하면 된다.

소매 단계 디자인에서 모델 이용까지

In each stage of development, the benefits of models change. In the design, or pre-alpha phase, the use of models accomplishes two goals:

개발 각 단계마다, 모델 변화를 통한 이득이 생긴다. 디자인 또는 알파 이전 단계에서, 모델 사용에 대한 두 가지의 목표가 있다 :

1. 전투, 스킬이나 제조 시스템은 쉽게 바꾸도록 한다. 이렇게 하면 이전 디자인과의 비교를 용이하게 하여 시뮬레이션을 해 보기 쉽다. 그리고 변화의 결과를 양적으로

보여줄 수 있다.

2. @Risk에 의해 발생한 그래픽 아웃풋은 회의를 하는 동안 논점에 집중할 수 있도록 해 준다. 각기 다른 팀 구성원이 MPC 몬스터와 플레이어와 클래스와 같은 결정적인 시스템을 개발하고 있을 때, 똑 같은 기준점을 공유하는 것이 매우 중요하다. 현재 시스템 디자인의 그래픽적인 표현을 공유하는 능력은 아주 중요하다.

알파와 베타 단계에서, 모델에 의해서 도출되는 이론적인 결과는 실제 플레이어 데이터와 비교될 수 있다. 플레이어의 기본적인 통계, 스킬 포인트 할당, 무작위적으로 발생하는 아이템 배분과 같은 변수 분포가 정제될 수 있으며 모델의 입력 요소가 적당해 지도록 한다. 게다가, 이 단계의 개발 중에, 플레이어의 성향에 관한 모델 내에서의 가정이 타당할 것인지 타당하지 않을 것인지를 알 수 있다. 게임 결과에 따른 실제 모델과 비교함으로써, 모델은 정제될 수 있고 훨씬 정확해 진다.

구매자를 대상으로 하는 테스트를 할 때, 모델에 의한 결과는 게임 내 로그와 비교될 수 있다. 너무 강력한 클래스에 관한 플레이어의 불평은 가장 큰 이점이 된다. 시뮬레이션 된 모델 결과와 게임 로그를 비교함으로써, 디자이너는 캐릭터 클래스가 의도한 대로 기능을 하고 있는지 아닌지를 판단할 수 있다. 게임 내 결과가 시뮬레이션 결과와 다르다면, 모델은 코드 내 존재하는 버그를 찾아낼 수 있게 해 준다.

누가 모델을 디자인하는가?

게임 인터랙션을 모델링하는 임무를 할당하기 위한 논리적인 방법이 두 가지 있다. 첫 째는 개발 팀 스스로가 그들 만의 모델을 만들기 위해 다양한 시스템 디자이너를 두는 방법이다. 두번째는 전체 게임에 대해 납득할 만한 모델을 개발하는 책임을 지는 한 명의 팀 구성원을 두는 방법이다. 후자의 경우, 모델은 디자이너가 제공하는 다양한 시스템의 정보에 기반을 두게 될 것이다. 두 가지 모두 장 단점을 가지고 있다 :

스스로의 모델을 만드는 시스템 디자이너의 장점과 단점:

+ 모델러는 모델화 된 시스템을 잘 이해하고 있다
+ 결과물을 관찰하면서 인풋을 조절하고 바꿀 수 있는 능력을 의지대로 할 수 있다
+ 인풋 변인이 곡선을 어떻게 움직이는지를 잘 이해한다
- 디자이너가 Excel과 확률 분포에 익숙하지 않을지도 모른다
- 평형적이고 관련되지 않은 시스템이 같은 모델 안에 자리잡을 수도 있다
- 리드 디자이너가 수 많은 자원을 검토해야 할 필요가 있다

모든 시스템을 모델링 하는 한 사람을 할당하는 것에 대한 장점과 이점 :

+ 모든 게임 시스템을 유기적으로 담고 있는 모델을 만들 수 있다
+ 한번에 모든 시스템의 결과물을 시스템 디자이너가 볼 수 있다

+ 전체 게임에 영향을 미치는 변화가 한 장소에서 분석될 수 있다
+ 디자이너는 수평적 시스템 디자인 분석에 관해서 한 사람에만 말 하면 된다
- 모델러와 디자이너 같은 완벽한 오픈 커뮤니케이션 필요
- 모델러는 시스템 디자이너가 수용할 수 있는 수준 이상으로 밸런스 인풋으로 마무리 지을 수 있다
- 오직 한 사람만 게임 시스템이 어떻게 모델화 되었는지를 잘 이해한다

어떻게 팀을 구성할 것인지는 리드 디자이너가 결정할 몫이다. 두 셋업 모두 긍정적이고 부정적인 측면을 가지고 있기 때문에, 디자이너는 팀의 리더십 스타일과 기업 문화에 가장 적한 것을 선택해야 한다.

컴퓨터 모델의 제한점

Unfortunately, the use of spreadsheet models and the @Risk add-in does not guarantee balance in a game. Player interaction models are simply a method by which real game results can be predicted. It's up to the designer to analyze the simulation results and determine whether they are acceptable. As previously mentioned, a simulation's results are only as good as the model that produced it. The two main drawbacks of spreadsheet models are:

불행히도, 스프레드 시트 모델과 @Risk 이용은 게임 밸런스를 보장해 주지 않는다. 플레이어 인터랙션 모델은 단순히 실제 게임 결과를 예측할 수 있다는 것에 의한 방법론에 지나지 않는다. 시뮬레이션 결과를 분석하고 수용할지를 결정하는 것은 디자이너의 몫이다. 이전에 언급했듯이, 시뮬레이션의 결과는 그것을 만들어 낸 모델 만큼만 좋을 뿐이다. 스프레드 시트의 주요한 두 가지 단점은 다음과 같다 :

1. 모델러가 플레이어 성향과 플레이 패턴에 익숙하다
2. 올바르지 않은 인풋과 가정을 모델 안에 포함시킨다

첫번째 단점의 경우, 모델을 디자인 한 사람이 디자인하고 있는 시스템 뿐만 아니라 캐릭터의 스킬과 능력을 이용하고 있는 플레이어에게 익숙하다는 것은 부득이 한 것이다. 주어진 시나리오의 결과를 예측하기 위해 모델을 시도한다. 만약 플레이어나 플레이어 그룹이 개발자가 기대했던 것과 완전히 다른 방식의 상황에 접근하고 있다면, 모델은 유용하지 않다.

이러한 좋은 예가 Mythic의 *Dark Age of Camelot* 이다. 전통적인 플레이어 대 환경 레벨링 그룹은 무사, 마법사와 치료사의 조합이었다. (“레벨링 그룹”은 경험치를 얻어서 캐릭터의 레벨을 증가시키기 위해 몬스터를 죽이는 목적으로 함께 플레이를 하는 다수의 플레이를 지칭하는 용어이다.) 전통적인 그룹은 한 명의 플레이어가 PvE 전투에서 이와 비슷한 그룹을 이용한다면, 쉽게 모델화될 수 있고 결정될 수 있다. 그러나, 마법사 클래스의 인기가 상승 및 렛 풀링 기술은 레벨 역학을 유의미하게 바꾸었다.

(*Dark Age of Camelot* 에서, Hibernia 영역 내에서 마법사 클래스는 펫 클래스이다. 플레이어

캐릭터는 플레이어에 의해 쉽게 조종될 수 있는 NPC 펫을 소환할 수 있다. 마법사는 “데미지 쉴드”라 불리는 주문을 할 수 있다. “포커스 쉴드”는 마법사가 움직이지 않고, 해를 입히거나 어떤 다른 주문을 외우려고 시도할 때 마법사의 펫이 이를 유지할 수 있도록 할 수 있다. 포커스 쉴드는 마법사의 펫을 활동적이게 해 주는 반면, 펫에게 해를 입히는 PC나 NPC는 포커스 쉴드에 의해서 해를 입게 될 것이다. 또한, 포커스 쉴드는 NPC 몬스터를 가해하는 “argo”를 엄청나게 발생시킨다. "Agro"는 NPC 몬스터가 다른 플레이어게 가지는 상대적인 공격성을 일컫는 용어이다. 일반적으로, argo 가치는 만들어 지고, 힐링 주문이 PC에 의해 잘 이루어지는 것 만큼 NPC에게 입히는 데미지 역시 증가한다. 게임의 AI는 NPC가 굉장한 argo 수준으로 플레이어 캐릭터를 공격하도록 만든다. 마법사와 그들의 “렛 풀”과 포커스 쉴드의 경우에서, 마법사의 NPC 펫은 항상 argo를 유지히킨다.)

데미지를 흡수하고 argp를 발생시킬 수 있는 몇 개의 난투 캐릭터로 전통적인 그룹을 조합하는 대신에, 전형적인 그룹 조합은 수 많은 마법사를 포함시키는 바뀌었다. PvE 전투에서 다루어지는 데미지를 마법사가 대부분 발생시킬 수 있다는 점 때문에, 몬스터는 기대한 것보다 훨씬 빨리 죽었고, 플레이어의 레벨업 비율이 증가했다. 이런 것과 같은 시뮬레이션을 포함하는 오리지널 모델은 없었다. 그러므로, “전통적인” 그룹 모델이 타당화되지 않은 상태에서 마법사가 디자인되었다 할 수 있는데, 마법사 레벨링 방법이 완벽히 다르지 않았기 때문이라 할 수 있다.

플레이어 성향이 이와 같이 변하고 있을 때, 모델은 새로운 플레이 패턴을 고려해서 재조명되어야 한다. 이런 식으로, 조절은 게임 시스템이 오리지널 디자인 목표를 유지하면서 이루어질 수 있도록 해야 한다.

정확하지 않은 인풋의 경우, 이것은 몇 가지의 상황을 만들 수 있다. 모델에 변수를 삽입할 때, 모델러는 평균 플레이어를 위한 결과를 도출하고자 한다. 만약 모델러가 생각하는 “평균”의 개념이 정확하지 않다면, 모델에 의한 도출된 그 결과는 타당하지 않을 것이다. 그러므로 알파와 베타 단계에서 진정한 플레이어 캐릭터를 획득하는 것이 중요하며 그 데이터를 모델의 인풋 분포와 비교해야 한다. 만약 실제의 데이터가 모델의 인풋과 다르다면, 이러한 인풋은 다시 조정되어야 하며 시뮬레이션은 기대되는 결과물에서 발생한 변화를 체크하여 다시 수행되어야 한다.

실질적인 모델 디자인

RPG 시스템을 개발할 때, 모델은 게임 시스템 그 자체 만큼이나 복잡해야만 한다. 시스템 복합성을 무시하면, 게임 시스템 밸런싱과 개발의 필수요소는 똑같다. 어떤 리스크 분석 원리가 게임에 활용될 수 있는지에 관한 방법론을 설명하기 위해, 나는 예를 들어보겠다. 극단적으로 간단한 전투 시스템이 있는데, 이것은 모델화 되고 나서 밸런스가 맞추어질 것이다. 두 예의 시나리오는 모델의 기본 케이스 역할을 하는 첫 번째 시나리오를 살펴볼 것이다.

이 시스템의 기본 케이스는 한 손 검/방패와 두 손 검 휘두르기 대전이다. 목표는 이 같은 듀얼 상황이 발생했을 때 시스템을 조절하는 방법을 결정하고 시스템을 모델하는 것이다. 두 캐릭터 모두 이길 확률이 동등하다. 두 번째 시나리오는 두 타입(첫 번째 시나리오 상의 타입)이 같은 레벨의 NPC 몬스터에 대항하는 것이다. 목표는 두 명의 플레이어 캐릭터 각각이 그들의 디자인 템플렛을 바꾸지 않고 몬스터와 같은 밸런스를 가지는 것이다. 전반적인 목적은 NPC 몬스터에 대항하여 이길 확률이 75%를 갖는 그런 캐릭터를 디자인하는 것이다.

이 글의 공간적인 제약 때문에 샘플 전투 시스템은 의외로 간단하다. 그럼에도 불구하고, "치기", 공격 위치, 다양한 스피드 무기, 치명타와 같은 기능에 영향을 주는 다수의 요인을 이용한 좀더 복잡한 시스템이 밸런스를 맞추는데 어려움을 주지 않는다.

디자인 파라미터

기본적인 시스템에서, 아래의 상수, 변수와 방정식이 활용된다.

상수 :

- 캐릭터 레벨
- 캐릭터 스킬
- 갑옷 요인
- 기본 무기 데미지
- 기본 무기 변차

변인 :

- 기본적인 통계 (힘, 체력, 민첩성, 지능)

방정식 :

- 두 번째 통계 (히트 포인트)
- 데미지 흡수
- 회피/방어/공격 성공률
- 평균 무기 데미지
- 평균 무기 변차

캐릭터 레벨은 모델에서 50으로 정의되었다. 두 캐릭터에 각각 부여받은 스킬은 기본 무기 형태와 기본 방어 메케니즘으로 제한되었다. 한 손 검을 쓰는 캐릭터의 갑옷 요인은 200으로 맞추어졌다. 최초의 무기 데미지와 변차는 24.3과 1.5였다. 두 손 검을 사용하는 무기의 경우, 데미지와 변차는 31.3과 1.5였다.

기본 캐릭터 템플렛에서, 정의된 변인은 기본 캐릭터 통계치였다. 시스템은 각각의 4가지 주요 통계치가 기본 가치를 90 점을 가지는 지점에 적용되었다. 플레이어는 4개의 상태 중 각 한 가지 마다 최대 25점을 가질 수 있는 부가적인 60점을 할당받을 수 있다. 플레이어 선택을 시뮬레이션하기 위해서, 트런케이티드 카이 제곱 함수가 이용되었다. 평균 값은 스킬 마다 15점으로 맞추어졌고, 중앙치는 14.86이고 14.04의 모드였다. 분포는 각 통계마다 1점으로 낮은 쪽이 불완전했다. 높은 쪽은 통계마다 25점이었다. 정확한 60점이 4개의 우선 통계치에 할당되도록 하기 위해서, 분포는 세번 만 적용되었다. 민첩성은 전반적인 통계치에 적합하도록 함수화되도록 남겨졌다.

시뮬레이션에 사용된 방정식은 캐릭터의 주요 통계치와 스킬에 관련되었다. 타격 포인트는 캐릭터 레벨과 체력의 함수에 기반을 두었다. 데미지 흡수는 갑옷 요인 (상수)와 캐릭터의 다양한 체력양과 직접적으로 연계되었다.

“타격”은 다음과 같이 계산되었다. 피하기는 캐릭터의 민첩성과 직접적으로 관련되어 있어서, 평균적으로 10.5%의 피하기 찬스를 가질 수 있었다. 막기는 방어 기술의 80%와 관련되었고, 회피는 기본 스킬 레벨의 60%에 맞추어졌다. 피하기, 막기 및 회피 확률은 합쳐졌고, 그리고 나서 각각의 캐릭터가 데미지를 입는 퍼센트를 결정하기 위해서 1에서 뺐다.

가장 마지막의 두 방정식은 무기 데미지와 데미지 변차를 다루는 것이었다. 기본적인 통계치 변경 요소는 무기 데미지와 변차에 영향을 미치도록 개발되었다. 변경 요소는 힘과 지능의 제곱 평균의 제곱근에 기반을 두었다. 차례대로, 실질적인 무기 데미지는 우선적 통계 변경치의 함수였고, 무기의 기본 데미지와 캐릭터의 무기 스킬도 마찬가지였다. 무기 변차는 기본적인 통계 조절치와 변차와 관련있었다.

전투를 모델링 할 때, 일률적인 1에서 1사이의 분포가 이용되었다. 이것은 타격이 성공적인지 아닌지를 결정하는데 필요한 정보를 제공했다. 동일한 분포는 또한 각각의 전투마다 다루어지는 데미지를 결정하는데 이용되었다. 이 분포는 평균 무기 데미지에서 무기 평균 변차를 뺀 것과 평균 무기 데미지에 무기 평균 변차를 더한 것 사이 값 내에 위치하도록 했다. IF 상황은 데미지가 주어졌을지 아닐지를 결정하여 타격/타격 실패 비율을 결정할 수 있도록 하였다. 마지막으로 각 캐릭터의 타격 포인트는 매회 마다 추적되었다.

무기 데미지와 변차가 이 모델의 균형을 맞추는데 조절되는 유일한 요소였기 때문에, 이들의 최초 값은 도피 확률과 흡수 요소에 의한 무기 배가된 무기 데미지와 캐릭터의 히트 포인트로 나눈 값에 의해서 결정되었다. 평균 전투 라운드의 수는 30회로 지정되었다.

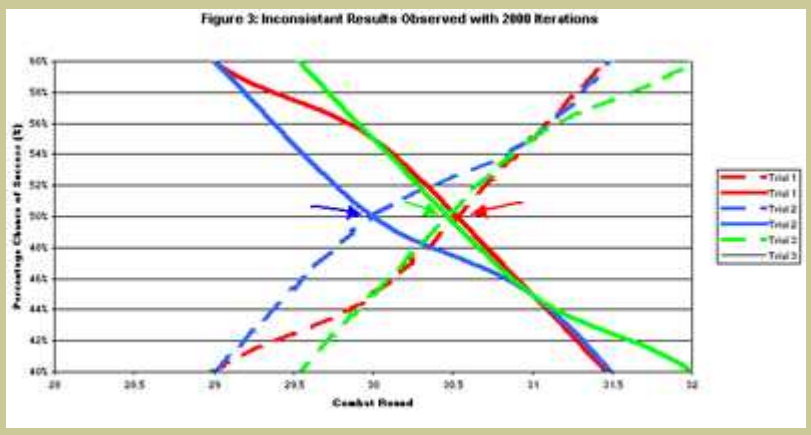

Figure 3: Inconsistent Results Observe with 2000 iterations. Click on Image to Enlarge.

결과 분석하기

처음, 모델을 2,000회 반복했지만, 시뮬레이션 결과로 수용할 수 없는 변차를 나타내는 영역으로 샘플을 제한하였다. 관찰된 변차는 그림 3에 나타나 있다. 좀더 일관되고 자연스러운 결과를 발생시키기 위하여, 몬테 카를로 샘플링으로 5,000회에서 10,000회의 반복이 필요했다. 결과적으로, 모든 시뮬레이션은 10,000회 반복되었다.

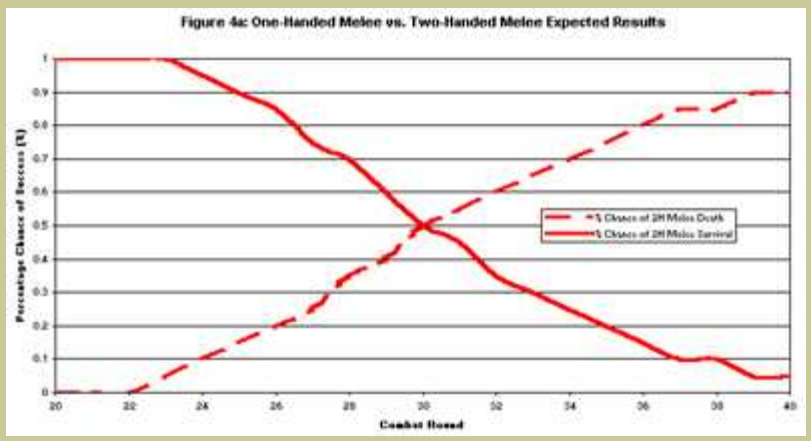

Figure 4a: One-Handed Melee vs. Two-Handed Melee Expected Results. Click on Image to Enlarge.

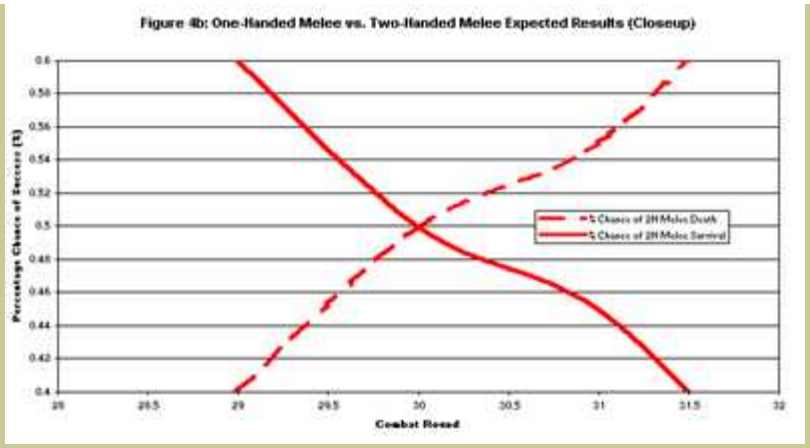

Figure 4b: One-Handed Melee vs. Two-Handed Melee Expected Results (Closeup). Click on Image to Enlarge.

각 시뮬레이션에 수행된 반복 횟수가 증가함에 따라, 확률 곡선의 변차가 사라지고 시뮬레이션 결과는 일관성 있는 가치를 가지게 되었다. 한 손 사용과 두 손 사용 시도에 대한 배교 결과는 그림 4a와 4b에 나타나 있다. 이 두 수치에 나타나 있는 정보는 주어진 라운드에서 전투에 이길 확률을 설명한다. 그림 4a에서 알 수 있듯이, 라운드 22이 이전에, 각각의 캐릭터가 죽을 확률은 거의 없었다. 스물 세번째 라운드에서, 한 손 사용 캐릭터는 죽을 확률이 5%인 반면에, 양손 사용을 하는 캐릭터는 0%로 남아있다. 24라운드에서, 한 손 사용 캐릭터의 죽을 확률은 10%로 증가하였고 두 손 사용 캐릭터는 죽을 확률이 5%인 것으로 관찰되었다. 30회 째 라운드에서, 50%에 도달하였다. 두 선이 교차하는 지점이 밸런스가 맞는 지점이다. 이 지점부터, 기대되는 결과가 획득된다. 만약 두 선의 교차가 60%에서 발생한다면, 두 손 사용 캐릭터는 60%의 승리 확률을 가지고, 반면 한 손 사용자의 성공 확률은 40%가 된다.

두 곡선 사이의 유사점을 살펴보는 것이 중요하다. 한 손 무기와 두 손 무기 사이의 전투에 관련해서, 두 곡선의 일반적인 기울기는 상대 방에 대해서 부정적인 값이다. 이와 같은 경향은 항상 보여지는 경우가 아니다. 모델에서 인풋을 조절함으로써, 기울기를 바꿀 수 있다. 이 예는 그림 5에 나타나 있다. 이 그래프에서, 두 선의 합일점은 30번째 라운드, 50%에서 발생한다. 그러나, 두 곡선의 기울기는 더 이상 상대방에게 부정적인 값이 아니다. 두 손 사용 무기는 한 손 무기 사용자를 40번째 라운드에서 100% 죽일 수 있다. 그러나, 한 손 사용자 무기는 죽기 이전에 75%의 최대치를 이끌어 낼 수 밖에 없다. 평가치의 낮은 끝쪽에서, 한 손 무기는 두 손 무기를 더 일찍 죽일 수 있는 기회를 가질 수 있음이 관찰된다. 그러나, 이 두 가지는 밸런스가 맞지 않는다. 15에서 40라운드 사이의 각각의 두 곡선의 영역을 계산해 보면, 이 영역은 한 손과 두 손 무기 각각 9.4와 8.5의 값을 가진다.

이것은 한 손 무기가 전투에서 이길 확률이 더 크다는 것을 나타낸다.

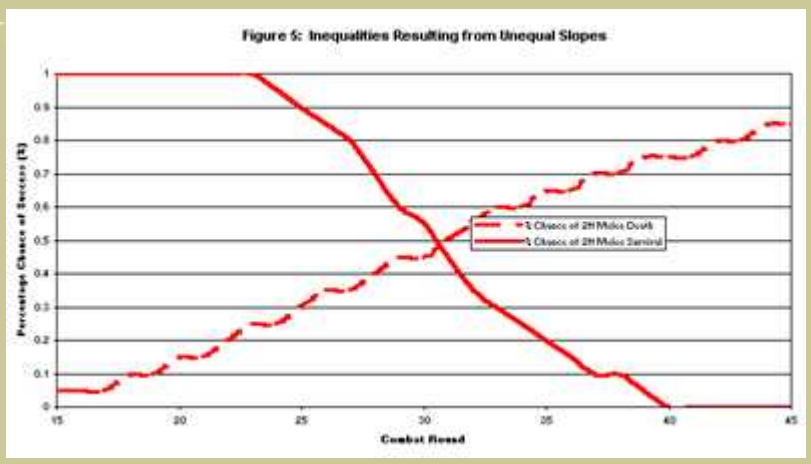

Figure 5: Inequalities Resulting from Unequal Slopes. Click on Image to Enlarge.

기울기를 수정하는 능력, 즉 이길 확률은 다른 형태의 클래스를 위한 스킬과 능력칠ㄹ 디자인할 때 장점이 된다. 전투 캐릭터를 마법사에 비교해 본다면, 궁수나 다른 범주의 캐릭터, 변수와 방정식은 데미지와 다른 흡수율의 범주에서 균형을 맞추어갈 수 있다.

시나리오 2 : 한 손 및 두 손 캐릭터 대 NPC 몬스터

기본적인 시나리오는 정의되었고 두 클래스는 분명히 밸런스가 맞추어졌다. 두 캐릭터 형태는 NPC 몬스터와 같은 레벨로 비교될 수 있다. 여기서 디자인 목표는 각각의 캐릭터가 NPC를 약 75% 확률로 이길 수 있게 하는 것이다. 이러한 결과는 캐릭터 템플렛을 바꾸지 않고도 가능할 수 있어야 한다는 것이 중요하다.

NPC 몬스터의 디자인은 대부분 상수값을 가지도록 되어 있다. 하나의 방정식을 사용하여 변수를 넣지 않는다. 몬스터와 관련되고 조정할 수 있는 통계치는 히트 포인트, 갑옷 레벨, 회피, 평균 데미지 및 평균 데미지 변차이다. 유일한 방정식은 데미지 흡수에 관한 것이고, 이것은 몬스터의 갑옷 요인에 비례한다.

기본적인 결과를 예측하기 위해서, 표 1의 데이터가 발생되었다. 표 1은 몬스터를 이기기 위해 필요한 평균 라운드 횟수를 나타낸다. 몬스터를 죽이기 위한 한 손 및 두 손을 사용하는 캐릭터의 가능성응ㄹ 모두 포함하고 있으며, 몬스터의 이길 확률 또한 볼 수 있다. 이 표에 인풋 값을 조정함으로써, 몬스터의 최초 값이 발생된다. @Risk 시뮬레이션은 기대되는 전투 결과물을 결정하기 위해서 시행되었다.

Table 1: Approximate Solution to PC vs. NPC Combat

	HPs	Damage	Absorbance	To Hit %	Avg. Dam.	Avg Rounds
1H	2525	200.0	20.5%	49.5%	78.7	32.08
Mob	2900	178.0	10.0%	68.5%	109.7	26.43
						5.65
	HPs	Damage	Absorbance	To Hit %	Avg. Dam.	Avg Rounds
2H	2525	200.0	20.5%	59.5%	94.6	26.69
Mob	2900	214.0	10.0%	68.5%	131.9	21.98
						4.71

Table 1: Approximate Solution to PC vs. NPC Combat

이것의 목적은 25% 부근이나 가운데에서 한 쌍이 선이 교차할 수 있도록 해야 한다는 것이다. 이런 식으로, 두 개의 캐릭터는 NPC 몬스터를 죽일 확률을 똑같이 가지게 된다. 그림 6에서 볼 수 있듯이, 세번째 시도는 최고의 결과를 보여준다. 그림 6에서, 두번째 시나리오에서 각각이 가지는 이길 확률의 차이를 관찰 할 수 있다. 평균적으로, 한 손 무기는 5번의 라운드를 통해 몬스터를 물리 칠 수 있다. 이것은 디자인 이슈를 낳는다.

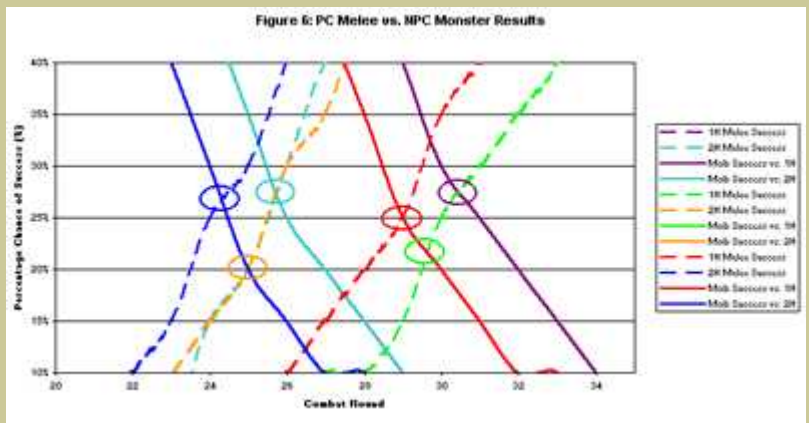

Figure 6: PC Melee vs. NPC Monster Results. Click on Image to Enlarge.

이것에 대해 떠올릴 수 있는 질문은 하나의 클래스가 다른 클래스와 비교해서 몬스터를 죽이는데 더 많은 시간을 필요로 하는 것이 수용될 만 한 가 아닌가의 문제이다. 만약 그럴 수 있다고 한다면, 디자인은 완벽하다고 간주될 수 있다. 만약 그럴 수 없다고 한다면, 하나의 캐릭터는 반드시 수용 가능한 차이가 어느 정도인지를 고려해야만 한다. 일단 이 수준이

결정되면, 모델의 인풋은 바람직한 결과를 얻기 위해서 조정될 수 있다. 일부의 경우에, 오직 몬스터의 모델만 수정하는 것은 이길 확률의 차이를 줄이게 될 것이다. 그러나, 대개는 모든 캐릭터와 몬스터를 조절하는 것이 필요할 것이다.

좀 더 복잡한 시나리오

이 두 모델은 디자인과 응용에 있어서 극도로 기본적이다. 좀더 현실적이고 실질적인 RPG 모델은 무기 속도, 전투 영역, 스타일, 특수 공격, 버프 대 버프가 없는 상태 및 그룹 전투 상황 관 같은 변수 요인을 고려할 필요가 있다. 이 모든 게임 특징을 설명하는 모델을 개발하는 것이 불가능하지 않다. 이것을 시뮬레이션 할 방법에 대해서 좀더 논의해 볼 것이다.

전투 속도 변수

대부분의 RPG 게임에서, 무기 형태는 다른 속도에서 전투할 수 있도록 고안된다. 게다가, 버프(캐릭터의 상태를 증폭시켜 주고 플레이어의 속도를 증가시키는 스킬/주문/아이템)와 같은 요인과 캐스팅 타임 단축 능력은 공격이 적용되고 주문이 걸리는 비율을 좀더 조절해 줄 수 있다. 결과적으로, 위에 제시된 전투 라운드를 모델링하는 가장 간단한 접근법은 의미있는 모델과 결과를 도출하는데 효과적이지 않다. 감사하게도, 이 변인을 설명하는 기능적인 모델을 디자인하는 것이 가능하다. 이 변수 공격에 관한 방법론은 세 가지가 있다.

캐릭터 공격 시간에 대한 차이를 설명하는 첫 번째 방법은 1초당 평균 데미지를 개발하는 것이다. 만약 무기의 평균 데미지가 200포인트라면, 무기는 매 3.5초당 한번씩 휘두를 수 있다. 그러면 간단한 수식으로, 1초당 평균 데미지가 57.14 포인트가 된다. 이런 식으로, 데미지의 비율이 정확하게 계산된다. 위의 예시에 활용된 라운드는 캐릭터와 몬스터 간의 평균 공격 속도를 나타낸다. 반응 무기 스티일이 부족한 전투 시스템을 개발하게 될 때, 무기 속도의 차이를 고려한 이 방법은 선호할 만 하다. (반응 공격 스타일은 무기에 따른 특수한 스킬이나 공격을 가지고 있어, 플레이어가 그들의 적에게 반응할 때 이용될 수 있다. 플레이어의 공격을 적이 막는다면 “스타일 a”를 이용하고, 적이 공격을 회피한다면 “스타일 b”를 활용한다는 식이다.) 그러나 반응식 전투 스타일이 있을 때, 이것의 기증성과 데미지는 좀더 정확하게 모델될 수 있도록 측정될 필요가 있다. 서로 연계되어 있는 무기 스타일의 경우에, 도출되는 결과는 스타일이 게임 메캐닉이 허락하는 것 이상으로 수행되지 않는 것을 확신할 수 있도록 분석이 불가피 하다. (연계 스타일은 연속적으로 함께 연결되어 있는 스타일을 말한다 : “스타일 a”는 반드시 먼저 수행되어야 하고, 그리고 나서 “스타일 c”와 마지막으로 “스타일 d”로 끝낸다. 만약 한 개의 스타일이 실패하면, 연속되는 스타일은 수행될 수 없다. 모델은 선행된 스타일의 성공과 실패를 확인하여 “스타일 d”가 수행되지 않도록 하는 것을 체크해야만 한다.)

두번째 전투 속도 변인을 다루는 방법은 전투-라운드 시나리오를 평균 무기 속도와 전투 시간에 기반을 두고 계산하고, IF 상태에 기반을 두는 것이다. 전투는 현재 전투 시간이 무기의 속도에 의해서 공평하게 나누어 질 수 있는지 아닌지를 알아보기 위해 모델 체크를 하도록 구성된다. 완전한 결과가 도출될 때, 캐릭터가 공격을 할 것이다. 만약 결과값이 완전하지

않다면, 여러분은 적당한 시간이 지나지 않았고, 행동이 수행되지 않을 것이라고 생각할 수 있다. 이 방법은 모델 정확성이 탁월할 때 가능하다. 그러나, 무기 속도 변수를 고려하는 방법은 모델의 크기에 의해 잠재성이 제한 받을 수 있다. 긴 전투의 경우, 모델 내 전투 횟수는 상당히 의미있다. 이것은 시뮬레이션 운영 시간을 증가시키고, 결과를 분석하는데 더 많은 노력을 기울일 필요가 있게 된다.

세 번째 방법은 상대적으로 간단하고, 정확성의 정도 때문에 선호된다. 앞서 제시된 기본 케이스의 예와 같이, 전투 라운드는 이 시스템에 의해 이용된 측정 단위이다. 주요한 차이점은 결과가 전투 라운드로 보여진다는 것이고, 라운드는 다른 모델과 비교되기 전에 시간 값으로 변화되어야 한다는 것이다. 이 시스템의 단점은 무기의 속도가 다양할 때 제약이 발생한다는 것이다. 무기 속도가 다양할 때, 예를 들어 3.5초에서 4.2초 사이일 때, 두 상태 중의 하나는 기준이 되어야 한다 : 둘 중의 하나의 평균 무기 속도가 전투 라운드가 시간 기준에 맞추어 변형될 때 반드시 가정되어야 하거나 시간 분포의 변수가 모델의 전투 결과에 포함되어야 한다.

전투 범위

전투 영역을 제대로 설명하기 위해서, 공격 당한 캐릭터나 NPC의 움직임 속도가 필요하다. 모델은 PC와 NPC 사이의 거리를 지속적으로 잴 필요가 있다. IF 상태를 이용함으로써, 여러분은 모델에서 최소한으로 요구하는 영역 보다 더 큰 PC와 NPC 사이의 영역을 가능 한 한 오래 이용할 수 있는 구조를 만들 수 있다. NPC 몬스터를 공격하는 궁수의 경우, 세 가지의 부가적인 전투 라인이 시뮬레이션을 위해 필요하다. 첫 번째 라인은 NPC의 위치를 궁수와 관련지어 측정하는 것이다. 두 번째와 세 번째의 라인은 궁수에 의해 받은 데미지의 크기와 범위를 나타낼 것이다. 이 두 데미지는 라운드 추기에 NPC의 위치를 체크하는 IF 상태를 포함하고 있다. 만약 NPC의 위치가 최소한의 공격 거리를 넘어선다면, 공격이 수행될 것이다. 만약 거리가 최소한의 구체적 거리를 넘어서지 않는다면, 제대로 된 공격이 수행되지 않을 것이다.

스타일, 주문 및 특수 공격

RPG 게임의 전투 스타일과 특수 능력의 증가는 전투 모델링 시스템이 그들의 데미지와 효과를 고려하고 있었음을 내포한다. 만약 특수 능력이 전투 시스템에서 무시된다면, 게임 내 결과는 극단적일 수 있다. 모델에 특수 공격을 적용할 때, 그 능력이 캐릭터에 의해 완벽히 수행된다고 가정해 보자 – 모든 스타일이 가장 적절할 때 이용된다. 이 유형을 모델에 디자인함으로써, 최적의 시나리오가 만들어진다. 차례로, 이것은 플레이어가 그들의 실제와 같은 스킬을 증가시키는 인센티브를 준다. 플레이어는 캐릭터의 능력을 이상적으로 활용하기 위해서 목표를 가지게 될 것이다.

플레이어의 경향에 관한 상당한 이해를 가지고 있는 사람은 이상적인 특수 공격 모델을

디자인할 것이다. 파워 게이머의 플레이 스타일에 익숙하지 않은 사람이 모델을 디자인한다면, 완벽한 시나리오는 만들어지지 않을 것이다. IF 구문은 AI가 반응 스타일이 활용되지를 결정하는데 필수적이다. 반응 스타일을 활용하는 것이 가능하지 않을 때, 가장 효과적인 스타일이 이용되어야만 한다 (예를 들어, 가장 큰 데미지). 여기서의 아이디어는 플레이어가 최대의 수준의 스킬과 효율성을 추구하고 있음을 가정한다는 것이다.

그룹 전투

정확하게 그룹 전투를 시뮬레이션하는 모델을 고안하는 것은 최적의 그룹 기능의 방법을 이해하고 주의 깊게 계획하는 것을 필요로 한다. 클래스 조합과 사이즈를 고려한 그룹은 효과적으로 모델화 될 수 있다. 그룹 전투를 모델링 하는데 있어서 주요한 압박감은 게이머의 플레이 패턴과 모델러의 친숙도 및 모델 개발에 소요되는 시간이다. 만약 여러분이 이 두가지 조건을 극복할 수 없다면, 여러분은 8대 8의 편안한 모델링을 해야 할 것이다.

플레이어 대 환경의 그룹 전투를 모델링 할 때, 최적화 된 플레이어 대 환경 그룹이 어떻게 작용할 것인지를 이해하는 것이 필요하다. 전통적인 그룹에서, 세 개의 주요한 캐릭터 형태는 전사, 힐러 및 마법사이다. 게임 디자인에 따라서는 부수적인 캐릭터 형태가 존재할 수 도 있고, 심지어는 필요하기도 하다. 부수적인 캐릭터는 플레이어 버핑, NPC 디버핑을 주어 그룹을 향상시키기도 한다. 그룹 내 다양한 역할과 다른 클래스가 결합될 수 있는 수를 증가시키기 때문에, 다른 그룹 구성 사이에 힘의 불균형을 시뮬레이션 하는 것이 중요하다.

표 2 : 캐릭터 정의

탱크. 전투에 특수화 된 캘릭터. 탱크 플래스의 대표적인 예는 전사, 워리어, 군인, 레인저 등이 있다. 갑옷을 입고 히트 포인트를 가진다.

힐러. 다른 캐릭터의 건강을 증진시키고, 다른 캐릭터에게 주문을 걸어 힐링을 시도하는 캐릭터. 드루이드, 위생병, 성직자 등이 대표적이다.

마법사. 주문을 통해 직접적으로 위협적인 데미지를 입히는 캐릭터. 위저드, 메이지, 스켈-캐스터, 매직 이용자 등이 대표적이다.

그룹 전투 모델의 발전은 좋은 Microsoft Excel 스킬을 가진 사람에 의해 이루어질 수 있다. 또한 플레이어의 경향에 밝고 의사 결정 구조를 만들 수 있는 리스크 분석을 적용하는 방법을 잘 이해하고 있는 사람에 의해서도 이루어진다. 그러나, 그룹 시뮬레이션을 시행하는 방법론에 대해서 논의방법을 이 글에서 다루기에는 너무 길어 질 수 있어 어려울 것 같다.

토네이도 차트와 민감성

게임 시스템 디자인에 모델을 이용하는 가장 중요한 점은 모델의 결과에 영향을 주는 개인의 인풋을 어떻게 결정할 것인가에 관한 능력이다. 변수 민감도는 “토네이도 차트”를 이용하여

결정될 수 있다. 토네이도 차트는 핵심 변인이 모델의 결과에 어떻게 영향을 미치는지를 분해하여 그래픽으로 나타낸 것이다. 토네이도 차트에 나타난 효과는 최초 값 (기본적인 통계치)과 계산을 통한 결과값을 포함하고 있다. 그림 7은 한 손 사용 대. 두 손 사용 시뮬레이션의 결과를 나타내는 변인의 효과를 보여주는 토네이도 차트의 샘플이다.

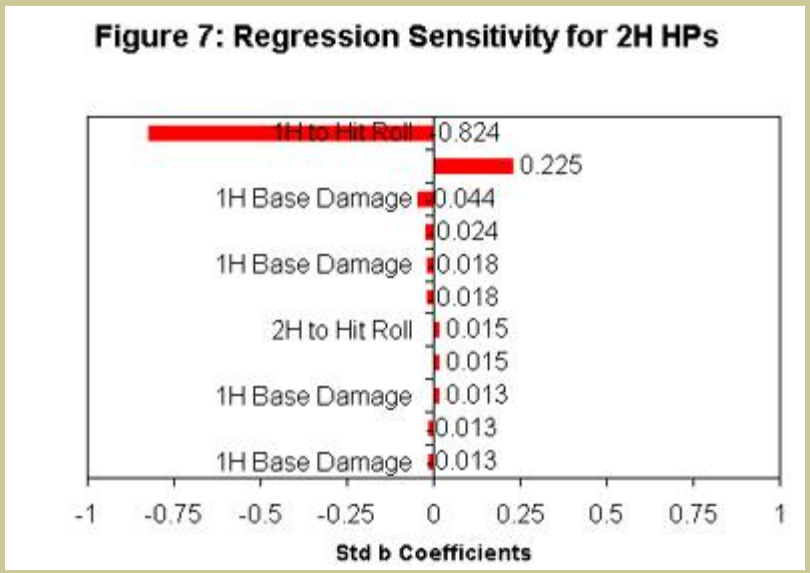

Figure 7: Regression Sensitivity for 2H HPs

부가적인 엑셀 특징

Microsoft의 Excel과 Palisade의 @Risk는 게임 밸런스를 맞추는데 도움을 주는 특징을 가지고 있다. "목표 찾기" 기능에서 좀더 복잡한 "해결책"까지의 다양함 영역을 가지고 있다. 방법을 반복함에 따라, 이러한 특징은 자동적으로 핵심 상수를 조절하여 만족할 만한 수적인 결과를 얻을 수 있도록 해 준다. "목표 찾기"는 인풋 셀의 값을 조정하여 바람직한 구체적인 값에 도달한 아웃 풋 셀을 구체화 할 때 까지 반복된다. "해결책"은 좀더 복잡한 목표 찾기의 버전으로 아웃 풋 셀을 평준화하거나 극대화 하거나 최소화하기 위하여 인풋 셀을 여러 번 수행할 수 있다. 이 두 도구 모두 플레이-밸런스 게임을 위한 모델을 이용하는데 최상의 도움을 줄 수 있다.

시스템 모델의 전반적인 이점

게임 시스템을 시뮬레이션 하기 위한 신뢰성 있는 모델의 사용은 값진 도구가 될 수 있다. 모델이 잘 디자인되었을 때, 게임 밸런스의 유의미한 흠을 찾아 낼 수 있다. 게다가, 모델은 다양한 다른 그룹을 적절히 다룰 수 있도록 디자인을 도울 수 있다. 이 글은 RPG에서의 클래스 밸런스와 시뮬레이션을 위한 방법에 초점을 두었지만, 리스크 분석 기술의 응용은 클래스에 국한된 것이 아니다. 숫자와 방정식으로 이루어진 모든 게임 시스템이 스프레드시트와 확률 패키지를 통해 시뮬레이션 될 수 있다.

결국, 모델과 시뮬레이션의 이용은 게임을 좀더 성공적이고, 이에 따라 좀더 많은 이윤을 낼 수 있게 만들어 줄 것이다. 이윤 추구가 주요한 목적이다. 향상된 수익은 게임 디자인 기업에 가용할 만한 자원을 증가시켜 줄 것이다. 이것은 다시 현재 상용화된 게임의 품질을 향상시키고, 새로운 게임 생산을 위해 필요로 하는 재원을 마련하게 해 줄 것이다.

저자 주: 이 글에 사용된 모든 엑셀 모델과 수치는 이를 살펴보고자 하는 사람들에게 제공가능하다. 게다가, 나는 게임 시스템 밸런스와 디자인을 위한 수학적 모델에 관한 실질적이고 이론적인 논의에 항상 관심이 있다. 만약 여러분이 Excel 파일을 받기를 원하거나 위의 기술 적용에 관해 어떠한 의견이 있다면, feralone@attbi.com나 feralone@bellsouth.net로 내게 메일을 보내라.

Resources

Palisade's @Risk: http://www.palisade.com