게임 디자인 과정의 함정
(10 Game Design Process Pitfalls)

Ian Fisch

가마수트라 등록일(2009. 5. 7)

(http://www.gamasutra.com/view/feature/4017/10_game_design_process_pitfalls.php)

[게임 디자이너인 Fisch는 게임 제작 과정을 조사하고, 게임 생산이 바라던 대로 되지 못하는 가장 큰 열 가지 이유와 이를 해결할 수 있는 방법들을 제안하였다.]

게임의 성패를 진지하게 살펴본다면, 그 결과가 전적으로 탁월한 기술이나 새로운 아이디어에서만 기인한 것이 아님을 알게 될 것이다. 그보다는 시종일관 적절하고 역량 있는 과정 수행이 관건이다.

Spider-Man 2, *Stranglehold*, *Assassin's Creed* 와 같이 무수한 게임들이 플레이어에게 혁신적인 개념을 전달하고는 있으나, 게임플레이 10-15 시간을 채우라고 했을 때에는 그 바닥을 드러내고 만다.

반면 *Call of Duty 4*와 같은 게임의 경우, 새로울 것이라고는 거의 없어 보이지만, 페이스 조절이나 레벨 구조에 있어서는 가히 월등하며, 플레이어가 플레이 내내 빠져들게 만든다. *Portal* or *World of Goo*와 같은 혁신적이라고 여겨지는 게임들 조차 그 제작 과정이 충실하지 않았다면 성공하지 못했을 것이다.

필자는 매우 좋은 개념을 가지고 시작하였지만 이를 흥미를 돋우는 비디오 게임으로 변환하는 데에는 문제가 있었던 여러 팀들과 함께 작업해 보았다. 필자가 깨닫게 된 것은 특정한 관행이 나타나 개발자가 디자인 팀의 역량을 최대한 끌어내지 못하게 한다는 것이다.

이러한 관행들을 아래 10 가지 “디자인 과정의 함정들”로 체계화해 보았다.

1. 게임 플레이를 위해 체계적으로 시간을 떼어놓지 않음

첫 번째 함정은 사소해 보일 수도 있겠으나, 사실은 디자인 과정의 품질과 효율성에 큰 영향을 미칠 수 있다. 이상적으로는 생산 전 과정에서 디자인 팀 내 모두에게 일정한 시간을 할애하여 이제 곧 씨름하게 될 게임을 직접 플레이 해 보도록 해야 한다.

그런데 이런 상황이 체계적으로 이루어지는 것을 단 한 번도 보지 못했다. 이제 팀원들이 잔뜩 골몰하도록 재촉해야 할 프로듀서의 입장에서 "플레이 타임"은 좀처럼 쉽지 않은 결정일 수 있으며, 다섯 명으로 구성된 팀에서 "연구"에 한 주 이상을 계획한다는 생각은 다소 사치스런 플레이 세션처럼 보일 수도 있다.

그 대신 디자이너들이 밖에 무슨 일이 있는지에 대해 이미 많이 알고 있을 것으로 기대하곤 한다. 안타깝게도, 모든 게임 디자이너들이 시중에 있는 모든 게임을 다 플레이 해 볼 만큼 시간이 많지는 않다. 보통은 게임 디자이너의 나이가 많을수록, 자유 시간은 그 만큼 더 줄어든다. 더욱이, 디자이너가 남는 시간에 플레이 해 보려고 선택하는 게임은 개발 목적으로 플레이 해야 하는 게임들과 일치하지 않을 수도 있다. 예를 들어 많은 평범한 게임들에는 흥미로운 아이디어들이 포함되어 있으며, 학습 도구로서도 가치가 있다.

해당 장르의 다른 게임을 플레이 해 보는 데 시간을 바침으로써 더 나은 제품을 산출할 수도 있으며, 상당 시간을 절약할 수도 있다. 우리가 작업 중인 장르의 게임을 디자이너가 플레이 해 보는 프로젝트에 참여해 본 적이 있었는데, 이 경우 이미 수개월을 바친 것보다 더 나은 플레이어 내비게이션 시스템을 "발견"하게 된다.

또 다른 프로젝트에서는, 누군가 우리가 막 논의한 것과 정확하게 일치하는 비인기 게임에서 그와 같은 3 인칭 기법이 사용된 것을 "발견"하였다. 만약 이 게임을 플레이 해 보았더라면, 이 기법이 문헌에서 볼 때처럼 그다지 재미가 없다는 것을 미리 알았더라면 논쟁에 허비한 수주 간의 시간을 절약할 수 있었을 것이다.

Nintendo/Retro Studios의 *Metroid Prime 3: Corruption*

디자이너가 모두 관련 게임을 플레이 해 본 경험이 있을 때, 서로에게 아이디어를 표현하는 능력은 훨씬 더 향상된다. 어느 한 디자이너가 상대방에게 “오른쪽 스틱을 눌러 토글이 가능해지는 경우 말고, *Metroid Prime*에서와 같은 락 온 시스템(lock-on system)을 사용해 볼 수도 있다”고 말할 수 있을 것이다.

상대방 디자이너가 *Metroid Prime*을 플레이 해 보았다면, 무슨 말인지 즉시 이해할 수 있을 것이다. 락 온 시스템이 무엇인지 설명할 필요는 없을 것이다. 그게 무슨 느낌인지, 말로 설명할 수 없는 무엇인가를 이해할 수가 있다.

전에 *Gears of War* 의 수류탄 투하 시스템이 문서 상으로 설명하기에는 너무 복잡하다고 생각한 팀장과 작업한 적이 있다. 결국 일단 앉아서 직접 해 보고는, 그 아이템을 포함시키는 것을 승인하였다.

디자이너 팀장이 디자인 팀의 나머지 팀원들과 함께 이 연구 작업에 직접 참여하는 것은 중요한 과정이다. 디자이너 팀장이 “덜 중요하다”고 생각할 경우, 팀의 다른 팀원들도 그처럼 여길 것이다. 더욱이 팀장이 주요 의사 결정권자이기 때문에, 다른 디자이너들과 동일한 어휘로 말할 수 있는지 여부가 중요하다.

일정 기간에 걸쳐 체계적으로 게임플레이를 해 보기 위해서는 일종의 설득이 필요할 수도 있다. 프로듀서는 눈으로 식별 가능한 결과물이 있을 경우 이를 시행해 보기가 더 쉬울 것이다. 이를테면, 각 디자이너가 매일 퇴근 전에 팀장에게 노트를 제출해야 할 수 있다.

2. 문서상의 디자인 작업에 너무 큰 비중을 두는 일

수학과 학생들은 카오스 이론의 기초적인 아이디어 중 하나인, 날씨와 같은 시스템이 너무 복잡해서 정확한 예측이 불가능할 수도 있다는 사실을 알고 있을 것이다. 상호작용하는 변수들이 너무 많아, 그 중 하나가 중요한 변화를 초래할 수도 있다. 게임에서도 같은 느낌을 받는다.

디자이너는 특정 레벨을 어떻게 플레이 할 지 머리 속으로 그려볼 수 있다. 하지만 상호작용 하는 수 백 가지의 요소들, 이를테면 환경, 제어, 게임 역학, 물리학 또는 AI 등이 각각 전체 재미 요소에 큰 영향을 미칠 수 있다. 이것은 필자가 문서상의 디자인에 너무 많은 비중을 두지 않는 한 가지 이유이다.

두 번째 이유는, 필자의 경험에 비추어 볼 때 최상의 아이디어는 실험과 행복한 우연적 상황에서 발견된다는 것이다. 문서상 디자인에 기초해 레벨을 구성하다가도 종종 AI 가 독특한 방식으로 환경과 상호작용하는 것을 알게 되었을 때에 그것이야말로 소소하지만 멋진 경험이 아닐 수 없다.

이제 이 사건을 대부분의 플레이어가 접할 수 있는 것으로 만들기 위해서 레벨을 수정하게 된다. 이처럼 게임의 시스템과 상호작용하여 재미요소를 발견하는 형태는 디자이너가 원본 문서 디자인에서 자신의 레벨을 수정할 자유가 있을 때에만 가능한 것이다.

안타깝게도, 디자이너가 항상 이러한 자유를 갖는 것은 아니다. 디자이너가 3D 로 하듯이 서류상으로 레벨을 제작하는 데 동일한 시간이 주어지는 팀에서 작업한 적이 있었다. 일단 문서 디자인이 끝나면, 큰 규모의 회의에서 이를 발표하고 분석 및 승인 과정이 이루어진다.

디자이너가 3D 구현 중에 큰 변화를 원할 경우, 이는 문서 작업에서의 오류를 수정하는 것으로 여겨진다. 문서상의 디자인을 재편집해야 하며, 부서장으로부터 다른 승인을 더 거쳐야 했던 것이다.

디자이너는 처음부터 이를 제대로 했어야 한다고 생각했다. 이 과정의 최종 결과로, 게임에 적합하지 않은 환경이 만들어 진 것이다. 예를 들어, 어떤 환경에서는 플레이어가 수 마일 떨어진 곳에서도 적을 보고 쓰러뜨릴 수 있었다. 다른 경우에는 적들이 바로 눈 앞에 올 때까지도 보지 못 했었다.

3. 동료 평가를 대수롭지 않게 여김

독자가 게임 회사에서 일해 본 적이 있다면, 아마도 한 두 차례 회의에서 “더 많은 사람이 게임플레이를 해야 한다”는 말을 들은 적이 있을 것이다.

이러한 요청은 보통 작가나 프로그래머에게 하게 되는데, 필자는 디자이너가 서로의 작업에 대해서는 관여하지 않는 버릇이 있음을 알게 되었다. 내 작업에만 너무 집중하는 함정에 빠진 나머지 타인의 작업 분량을 플레이 해 보는 일은 무시하게 되는 것이다.

정기적으로 다른 디자이너의 레벨을 플레이 해 봄으로써 아이디어의 교류가 가능해 지며, 보통 더 나은 게임을 만들게 된다. 종종 디자이너가 다른 디자이너의 레벨을 플레이 해 보고, 특정 요소가 멋지다고 생각하고는 몇 가지를 개선하여 자기 레벨에 포함시키기도 한다.

건전한 경쟁은 각자가 서로의 아이디어를 취하여 이를 개선시켜 보려는 시도 가운데 이루어진다. 자신의 것 이외의 레벨을 플레이 해 봄으로써, 디자이너는 또한 자기 개인적인 관점에서만 보는 것이 아니라, 전체적으로 게임이 어떻게 플레이 될 것인지에 대해서 더 좋은 감을 얻게 된다.

동료 평가는 이처럼 팀장이 엄격하게 강요하지 않으면 무시되기 쉬운 요소이다. 필자는 팀이 매주 동료 평가 세션을 가져 모든 디자이너가 자신의 작업을 내놓고 팀원들의 작업량을 오후 내내 검토해 보게 할 것을 권하는 바이다.

동료 검토의 날은 데드라인에 너무 임박해서는 안 될 것이다. 예를 들어, 중대 시점이 보통 금요일에 있다면, 동료 검토는 화요일 오후 즈음에 계획해 둘 수 있을 것이다.

4. 프로듀서로서의 재능만 보고 선발되는 결정권자

디자인 팀에는 여러 가지 다른 역할이 있으며, 다양한 종류의 디자이너들이 있다. 필자의 경험으로는, 팀장의 역할을 담당하도록 승격된 디자이너들이 거의 전적으로 행정적 직무를 수행하게 되는데, 이를테면 어떤 적의 종류를 어떤 레벨에 포함시킬 것인지를 지정하는 스프레드시트를 제작하는 일 등이 이에 포함된다.

컨텐츠 제작을 수행하는 이들은 보통 그 일에만 머문다. 레벨 제작 우위의 작업을 하는 디자이너는 보상을 승진으로 받는 일이 거의 없으므로, 더 큰 게임 디자인 결정권을 갖게 될 기회는 없을 것이다. 두 종류의 디자이너 모두가 동일하게 중요하며, 여기에 불균형이 있을 경우 프로젝트에 악영향을 미칠 수 있다.

이러한 불균형의 이유는 분명하다. 팀장이 되려면 자신의 일 보다는 팀원들의 작업에 대해 책임감을 가져야 한다. 사교성도 좋아야 한다. 팀장이 하루 여덟 시간 내내 편집만 하고 있다면 이를 입증하기란 쉽지 않을 것이다.

반면 전체 팀이 수반되는 직무를 수행하고 있다면 그러한 자질을 입증하기가 한결 더 쉬워질 것이다. 예를 들어, 디자인 관련 요청 사항을 시기적절하고 체계적인 방식으로 프로그래밍 할 수 있을지 확인할 책임을 맡았다고 해 보자.

불행하게도 디자인 관련 결정권자가 흥미를 돋우는 컨텐츠를 직접 만들어 보지 않았다면 프로젝트 수행이 어려워질 것이다. 어떤 아이디어를 포함시킬 것인지 여부를 결정함에 있어서 현명한 선택을 하기가 어려울 수 있다.

설상가상으로 실무에 거의 시간을 내지 못할 경우, 어떤 과정이 좋은 컨텐츠 제작에 기여할 수 있는지에 대한 직접적인 지식이 없을 것이다. 예를 들어, 자신의 업무에서 동료 평가로부터 유익을 얻어 본 적이 없기 때문에 그 중요성을 인식하지 못할 수 있다.

좋은 팀장을 선발하는 일도 쉽지 않다. 만약 팀이 책임감 있는 팀장, 좋은 컨텐츠를 만드는 요소에 대한 좋은 감각이 있는 팀장이 있다면 그 팀은 운이 좋은 것이다. 이 두 가지 요소를 각각 갖추기 힘들 것이므로, 같은 사람이 반드시 두 가지를 모두 갖추어야 할 필요는 없다고 말하고 싶다.

디자이너 팀장은 디자인 과정을 관리하고, 다른 디자이너들의 작업을 검토하고, 툴 세트를 가지고 실무에 참여해 보면서 시간을 보낼 수 있을 것이다. 한편 직무 목록이나 주간 보고서를 만드는 등의 행정적 직무는 디자인부서와 긴밀하게 협력하는 프로듀서에게 맡기면 될 것이다.

5. 플레이스 홀더를 이용하지 않음

플레이스 홀더(위치 표시자) 데이터와 코드가 이 업계에서 많이 사용되고 있으나, 플레이스홀더의 사용은 일단 생산을 시작하면 급격히 감소하는 경향이 있다. 이해할 만한 일이다. 개발자가 출판업자를 위해 일하고 있을 경우, 진행 상황을 보여줄 수 있는지 여부는 매우 중요하다. 멋지고 번쩍이는 검증용 샘플코드를 보다가 유치한 애니메이션과 단순 음영 배경으로 돌아가는 것은 분명 역행하는 것처럼 보일 수도 있다.

내부적으로는 팀원들, 특히 작가와 애니메이션 작가가 보통 차후에 대체되어야 하는 저 품질의 데이터보다는 최종적인 데이터를 가지고 작업하기를 더 선호하는 법이다. 프로듀서는 보통 일을 두 번씩 하기보다는 한 번에 끝내기를 더 선호할 것이다. 안타깝게도,

플레이스홀더의 사용을 절감할 경우 최종 게임의 품질은 낮아지고 디자인 과정도 진행이 느려질 수 밖에 없다.

그 기본 시나리오는 다음과 같다: 디자이너는 게임에서의 데이터가 3D 모델, 애니메이션, 게임 플레이 기법 등의 형태로 나타나기를 원한다. 플레이스홀더를 몇 시간 기다리기 보다는, 최종 데이터를 기다리며 한 주를 보내야 할 것이다. 그가 받아 본 것이 상상했던 것 만큼 재미있지 않을 경우, 한 주 간의 작업량만 낭비한 꼴이 된다.

이런 일이 발생하지 않게 하기 위해, 불필요한 요소는 없애고 최상의 아이디어만 남길 생각으로 긴 승인 과정을 수행하곤 한다. 실제로 그러한 승인 과정에서 독창성을 잃지 않는 한 가장 위험성이 적은 아이디어 요소만을 남기고 상당 부분은 제거될 것이다.

따라서 플레이스 홀더로 아이디어를 시도해 보고 계속 진행하기 보다는, 디자이너는 플레이스홀더로 실험하여 발견할 수 있었을 것 보다 보통 훨씬 더 흥미가 떨어지는 최종 데이터를 받아 볼 때까지 한 주 간을 내내 기다린다.

플레이스홀더 없이 작업하는 것은 여러 면에서 시간 낭비이다. 수백 개의 3D 메쉬로 구성된 레벨은 단순한 BSP 로 구성된 것보다 로드하고 테스트 하는 데 더 많은 시간이 걸린다. 몇 개의 단순한 모양이 아닌, 100 개 단편으로 구성되어 있기 때문에 수정하는 데도 훨씬 더 오래 걸리기 마련이다.

경험이 적은 미션 디자이너는 재미있는 게임플레이에 주력해야 하는 상황에서 실내 장식이나 조명 기술을 뽐내느라 시간을 낭비할 것이다. 단순 음영의 텍스처, 플레이스홀더 3D 객체, 및 꼼꼼한 작가의 날카롭고 까다로운 지적 등은 미션 디자이너가 재미있는 환경을 만들어야 할 때 건축학적으로 필요한 요소들이다.

플레이스홀더가 많은 게임의 디자인 과정에는 요구되는 것도 많다. 디자인 팀이 “최종” 레벨에서 원하는 것에 대해 결정을 내리는 동안 작가들이 다른 작업들을 진행할 수 있게 할 만한 스튜디오가 필요할 것이다.

또한 고위층의 사람들(외부 출판업자나 내부 스튜디오 팀장)이 그저 표면적인 것이 아니라 플레이스홀더로 채워진 재미있는 레벨의 가치를 인식할 수 있을 것이라는 믿음이 필요하다. 이러한 믿음이 없을 경우, 외부 플레이테스트를 통해 정련된 그래픽을 보여주는 대신 “재미” 정도를 측정해 볼 수도 있다.

6. 스토리가 게임 디자인을 제어하도록 허용함

텔레비전이나 영화에서, 작가는 대본을 작성하여 이를 프로듀서에게 건네준다. 그러면 팀원들이 대본을 시청자들에게 스토리를 전개할 맵으로 사용한다. 이 과정은 게임의 주요 목적이 플레이어에게 스토리를 말해 주는 것일 경우, 비디오 게임에도 적용해 볼 수 있다.

많은 JRPG 및 어드벤처 게임뿐 아니라 *Final Fantasy* 시리즈와 같은 게임은 탁월한 스토리만으로도 성공할 수 있음을 보여주는 예들이다. 게임의 주요 목적이 뛰어난 대화형 상황 속으로 플레이어를 밀어 넣는 것이라면, 이 방법은 부적절할 것이다.

게임이 *Halo*, *Call of Duty*, 또는 *God of War* 등과 같을 경우, 스크립트가 게임플레이를 중심으로 틀을 구성해야 하며, 그 반대가 되어서는 안 된다. 레벨 디자인, 플레이 기법, 페이싱 등의 요소가 가장 중요하다. 이러한 요소들이 탄탄하지 않다면, 스토리만으로는 그 공백을 매울 수 없다. 스토리가 중요하지 않다는 이야기가 아니다. 오히려 작가는 플레이어에게 가장 매혹적인 대화형의 상황을 경험하게 해 줄 목적으로 스토리를 변경할 만한 유동성이 있어야 한다.

게임 디자인의 다른 요소들보다 스토리에 우선순위를 두는 것은 문제를 야기할 수 있다. 그 고전적인 예가 바로 2003 년판 *Enter the Matrix*이다.

게임의 플롯과 라이브 액션의 단편들이 영화 The Matrix Reloaded 의 안팎에 짜 넣도록 만들어졌고, 영화 기술팀에서 대거 참여하였다. 그 결과 스토리 관점에서 레벨이 설정되었으나 게임플레이 관점에서는 아무런 목적성이 없게 되었다. 더욱이 개발팀은 세 가지 다른 게임 형태, 곧 차 안, 도보, 및 호버크라프트 안 장면을 적당히 구성하는 데에만 신경 쓰면 되었는데, 스토리에 따르자면 어쩔 수 없었다.

전에 할리우드 작가가 우리에게 건네 준 대본에 따라 3 인칭 액션 프로젝트를 작업한 적이 있다. 흥미로운 스토리이기는 했으나, 앞서 디자인한 게임플레이 요소와는 상당한 마찰이 있었다. 게임 역학에서는 거대 총격전이 있는 현장을 배경으로 하는 반면, 이 대본에 따르면 몇 명의 적들과의 접근전으로 레벨이 구성되어야 했다.

이 게임은 플레이어가 명령을 전달할 파트너를 염두에 두고 설계된 것이었으나, 대본은 플레이어가 게임의 후반부의 상당 부분을 혼자 싸우도록 요구하였다. 말 그대로 학습곡선에 역행하는 요구였다. 이 프로젝트는 결국 취소되었는데, 주로 설계 팀이 할리우드 대본을 따르도록 강요를 받았기 때문이었다.

액션 게임의 스토리는 게임 작가가 담당해야 한다. 게임 작가는 전에 게임 대본을 써 보았거나 최소한 본인이 게이머인 경우이다. 게임 작가는 전시간 팀과 함께 있으면서 그의 스토리가 왜 계속 수정되어야 하는지를 이해하게 될 것이다.

디자인 팀이 공항 편 레벨을 게임 말엽이 아닌 초엽부터 보다 이치에 맞도록 구성하였더라면 그다지 혼란스럽지는 않았을 것이다. 게임 대본을 작성하는 일은 영화 대본 작성과 같지 않다. 그저 할리우드 작가에게 대본을 써 달라고 수표를 내미는 것만으로는 충분하지 않다

7. 디자이너에게 충분한 툴을 주지 않음

디자이너가 게임 툴을 가지고 할 수 있는 일을 제한하는 것과 관련하여 많은 그럴 듯한 주장들이 있다. 디자이너가 사용할 만한 기능이나 변수들이 많을 수록, 잘못을 저지를 확률도 더 커진다.

디자이너 코드가 열악하여 이를 수정하는 것만큼 프로그래머가 하기 싫어하는 일도 또 없을 것이다. 더욱이 디자이너는 프로그래밍을 하도록 교육을 받지 않기 때문에, 프로그래머가 하도록 훈련 받을 일을 하느라 애쓴다면 시간만 낭비하는 것이 될 것이다.

이러한 견해도 일리가 있지만, 이 문제를 디자이너의 생산성의 견지에서도 바라볼 필요가 있다. 디자이너가 중단 없이 레벨 작업을 할 수 있다면, 좀 더 집중하는 가운데 더 생산적으로 진행할 수 있을 것이다.

디자이너가 새로운 기능을 원할 때마다 매번 프로그래머에게 달려가 애걸해야 한다면, 리듬이 깨지고 말 것이다. 단일 레벨에 집중할 수 있기 보다는 프로그래머만이 해결할 수 있는 문제에 봉착할 때까지 천천히 모든 레벨에 조금씩 관여하는 편이 낫다.

필자가 레벨 디자이너일 때 생산성에 있어 겪은 차이에 대해 개인적인 경험에 비추어 이야기해 보고자 한다. 레벨 작업에서 별다른 진전을 볼 수 없던 때에, 스낵 자판기에 더 자주 다녀오고 동료들과 잡담도 더 자주하고, 시계만 쳐다보곤 하였다. 중단 없이 작업을 할 때에는 퇴근 시간이 될 때까지 자리를 떠나지 않았다.

디자이너가 할 수 있는 것과 할 수 없는 것 사이의 균형을 깨어야 한다고 필자는 생각한다. 지저분한 디자이너 코드는 디자인 및 프로그래밍 팀이 제작하고

시행하는 엄격한 코드 표준으로 완화할 수 있다.

단순히 모든 디자이너가 중국제 상품 가게에 진열된 소처럼 모두 똑같다고 가정할 것이 아니라, 모두에게 최고의 시스템을 만들어 주기 위해 프로그래밍과 디자인 간의 원활한 의사소통이 필요하다.

8. 재미 요소 없이 생산에 돌입하는 일

Mario 64 을 제작할 때, 제작자들은 단순한 정원을 마리오가 돌아다니는 듯한 느낌을 연출하는 데 수 개월을 소비하였다. 제작자들은 플레이어가 3D 게임의 개념에 익숙해 질 동안 계속 붙잡아 둘 만큼 이것이 재미있을지 확인하고 싶어했다.

Mario 64 의 경우처럼, 많은 뛰어난 게임들은 기본 요소들로 돌아갈 때 그 재미 요소가 살아있다. *Metal Gear Solid 2* 는 영화 예술과 형식화된 눈에 띄는 장면이 없는데도 여전히 재미있다. 이는 *Metal Gear Solid 2: Substance* 의 VR 미션으로도 입증되었다.

마우스 버튼 하나만으로 전투를 벌이는 *Diablo* 의 경우도 이 게임에 더 깊이 있는 RPG 요소들을 부여하기 전까지 새로운 플레이어들의 관심을 끌기에 충분한 재미 요소가 있었다.

안타깝게도, 많은 프로젝트는 뚜렷한 재미 요소도 없이 전면적인 생산에 돌입하곤 한다.

재미있는 프로토타입 또는 검증용 샘플코드가 있다면, 프로젝트에는 큰 이익이 될 것이다. 길을 인도하는 등불처럼 작용할 수 있다. 프로젝트가 생산 단계까지 수개월이 지속되고 팀이 더 이상 재미있을 것이 없는 게임에 익숙해져 버린다면, 디자이너는 그 프로토타입을 시동하여 처음 플레이 할 때 느꼈던 감정을 되살려 볼 수 있다.

반면에, 어떤 재미 요소도 없이 생산에 들어갈 경우 사기가 땅에 떨어질 수 있다. 재미 없는 검증용 샘플코드만 가지고 생산에 돌입한, 하지만 모든 것이 균형을 잡으며 실행되기만 하면 실제 게임은 재미있을 것이라는 기대감과 함께 생산에 돌입한 프로젝트에 참여한 적이 있었다. 수 개월 동안 지지부진 한 가운데, 전혀 재미있지도 않은 아이디어들이 검증용 샘플코드 위에 쌓여만 가고 있었다.

디자인 팀 외부에 있는 대다수 사람들은 우리가 만들고 있는 것이 무슨 게임인지 명확하게 알지 못했고, 팀 내부 사람들은 통일된 비전을 제시하지 못했다. 이렇다 할 만한 재미요소가 없었기 때문에 게임 연출가가 선택한 이 과정이 과연 옳은 것이었는지 확신이 서지 않았다.

영세 자금으로 제작한 많은 독립 프로젝트들이 어떻게 수백만 달러를 투자한 프로젝트보다 더 재미있을 수 있는지 흥미로운 일이 아닐 수 없다. 이로써 반드시 비싸야만 재미가 있는 것은 아님을, 재미있을 만한 것도 없이 시작하면 어떻게 되는지를 잘 보여준다.

9. 디자인 관련 문서를 최신 상태로 유지하지 않음

"그것들은 읽을 필요가 없어. 최신 것이 아니거든"이라는 말을 얼마나 많이 들어보았는가? 한번도 들어본 적이 없다면, 아마도 게임 업계에 있지 않거나 매우 운이 좋은 것이다. 개발팀에서는 날짜가 지난 디자인 문서를 보는 것이 매우 흔한 일이다.

불가피한 일처럼 보일지 모른다. 기능이 문서 상에 설계된 대로 정확히 일치하게 출시되는 일은 거의 없다. 오히려 여러 차례 수정이 있게 된다. 수정이 있을 때마다 문서를 그에 맞춰 변경한다면 상당히 번거로운 일거리가 될 것이다. 시간 소모성 일로 보일지도 모른다. 그러나 그렇게 하지 않을 때 초래되는 결과는 매우 심각하다.

문서를 최신 상태로 유지하지 않으면, 사람들은 곧 그에 대한 신뢰를 잃게 될 것이다. 소스 컨트롤에 있는 것이 무엇이든 유효하지 않을 것이라고 생각할 것이다. 궁금한 것이 있을 때에도 귀찮게 문서를 참조해 보지도 않는다.

오히려 디자인 팀원에게 직접 물어보는 편을 택할 것이다. 일단 디자이너가 자신의 문서가 읽혀지고 있지 않다는 것을 알게 되면, 관리하는 데에는 더더욱 에너지를 소모하려고 하지 않을 것이다. 결국 팀은 "디자인 문서는 쓸모가 없다"는 결론에 이르게 된다.

말로써 아이디어를 전달하는 데에는 본질적으로 한계가 있다. 디자이너가 프로그래머에게 제시한 설명이 원래 합의된 것과 정확하게 일치하지 않을 수 있다.

프로그래머는 디자이너의 설명에 대해 기억하는 것에 기초하여 그 기능을 수행하게 될 것이며, 그러면 디자이너 팀장이 논의한 것과 다른 기능이 수행된 것을 보고는 논쟁이 일어날 것이다.

또 다른 문제는 정보가 단편화되어, 한 두 사람 정도만 각 구성요소들이 어떻게 작용하는지를 알게 되는 것이다. 필자는 게임에 포함될 레벨의 목록이 필요한 상황에서 프로젝트 생산은 이미 4 개월이 진행된 적도 있었다.

이러한 정보를 구하는 유일한 방법은 디자인 팀장에게 이메일을 보내면 그가 내게 자신의 데스크톱에 있는 스프레드시트를 보내주는 것이다. 그나마 소스 컨트롤에 있는 두 레벨 짜리 문서는 이미 오래된 정보였다.

디자인 관련 문서를 최신으로 유지하기란 쉬운 일이 아니지만, 기술적으로는 가능하다. 문서가 마지막으로 수정되고 2 주가 지나면 자동으로 문서에 "날짜가 지났음" 표시가 되는 시스템을 사용하는 회사와 일한 적이 있다. 날짜가 지난 문서는 디자이너가 최신 정보라고 표시하기 전까지는 열리지 않았다. 전반적으로 성공적인 시스템이었다.

10. 외부 플레이테스트를 과정의 일부로 여기지 않음

점점 더 많은 회사들이 정기 플레이테스트를 디자인 과정의 일부화하는 가치를 인식하기 시작했다. 예를 들어, Valve는 *Half-Life 2* 와 *Portal*을 제작할 때 엄격한 플레이테스트 과정을 사용한 것으로 잘 알려져 있다. Ubisoft 역시 플레이테스트에 상당한 우선순위를 두고 있는 것으로 알려져 있다.

안타깝게도 이러한 관행은 필요한 만큼 널리 확산되어 있지는 못하다. 많은 개발자들은 그 중요성을 잘 모르고 있으며, 심지어는 유해하다고 보는 이들도 있다. 반면 경험 많은 게임 디자이너는 플레이테스트를 툴 목록에서 가장 유용한 것으로 간주한다.

대상 플레이어가 전형적인 "하드코어" 인구 통계, 이를테면 가볍게 즐기는 게이머나 어린이 등에 포함되지 않을 경우, 플레이테스트는 불가피하다. 일반 플레이어의 눈으로 게임을 보려고 한다면 디자이너는 누적된 게임플레이 지식의 15 년 이상의 담을 쌓아버려야 하는데, 다시 말하면 불가능한 일이다.

가능한 한 가장 단도직입적인 인터페이스를 만들었다고 스스로 자부할지라도, 40 세 된 어머니에게 처음으로 컨트롤러를 건네주는 순간에 얼마나 많은 세부점을 당연시하고 있는지 알면 놀랄 것이다.

아동용 게임을 만들고 있다면, 훨씬 더 많이 조정하지 않으면 안 된다. 여섯 살 된 아이라면 가장 기본적인 개념을 이해하기에도 벅찰 것이기 때문이다. 그들의 뇌는 우리가 자연스럽게 이해하는 것들에 대해서 이해하기에는 아직 충분히 발달하지 않은 것이다.

하드코어 독자를 위해 디자인하고 있을지라도, 플레이테스트는 여전히 매우 중요하다. 생산 과정을 진행해 가면서 모든 게임 디자이너는 자신이 개발하고 있는 게임의 대가가 되어 있을 것이기 때문이다.

새로운 플레이어에게 도전이 되는 문제들도 디자이너에게는 그저 지루할 뿐이다. 자신에게는 그저 약간의 기교를 가지고 있으면 다룰 수 있을만한 셋업을 만들려고 할 때에도 실제 게임에 포함시키기에는 너무 어려운 것이다.

팀이 베타버전이 있을 때까지도 외부 플레이테스트를 시작하지 않은 경우, 결국 나중에 가서야 성급히 다듬게 되기가 쉽다. 종종 그 결과로 난이도 변경이 제멋대로가 되는 것이다.

반면 플레이테스트가 디자인 과정의 일부일 경우, 디자인 팀은 게임의 절정에 도달할 때까지 점차적으로 상승하도록 난이도를 정교하게 조정할 수가 있다.

개발 단계에서는 플레이테스트를 하지 않는 편을 선호하는 디자이너들과 작업한 적이 있었는데, 그들은 개발 단계에서 플레이테스트에 너무 의존하게 되면 특정 집단을 대상으로 만든 영화와 비슷해 질 것이라고 주장하였다. 예술적 시각을 흐리게 만든다는 것이다. 이러한 주장은 단순히 게임의 재미요소만을 고려하고 그 전반적인 이해가 부족한 것이라 할 수 있다.

Billy 가 그 게임을 재미있게 생각하는지 여부에 디자이너가 관심이 없다면, 그에게 묻지 말라. 하지만 그가 첫 번째 레벨도 통과하지 못해 쩔쩔맨다면, 그 이유가 듀얼 아날로그 스틱 조준 시스템을 사용할 줄도 몰라 일어난 일이라면, 문제가 발생한 것이다. 그런 문제는 빨리 알아낼수록 좋다.

결론

독자가 게임 디자이너로 일하고 있다면, 아마도 이러한 디자인 과정의 함정 중 적어도 몇 가지는 경험해 보았을 것이다. 필자는 돈도 있고 재능도 있지만 프로젝트가 엉성한 디자인 과정 때문에 실패하는 상황을 보아 왔다.

무엇보다 중요한 것은 본 기사에서 언급한 여러 가지 핵심 요소들, 이를테면 동료 평가, 문서를 최신 상태로 유지하는 일 등이 자연스럽게 업무의 일부가 되는 것이다. 그러려면 적어도 한동안은 규율과 강경한 시행이 필요하다.

게임 개발이 양질의 디자인 과정을 알고 실행하는 데 필요한 경험이 있는 사람들에 의해 관리된다면, 정말 놀랄 만큼 멋진 콘텐트를 만들어 낼 가능성이 있다.