개발 후기에서 살펴본 실수들

(What Went Wrong? Learning From Past Postmortems)

브랜든 셰필드(Brandon Sheffield)

가마수트라 등록일(2009. 04. 22)

http://www.gamasutra.com/view/feature/4001/what_went_wrong_learning_from_.php

[올해 초에 가마수트라 계열 출판사의 게임 개발자(game Developer)에 게재된 이 기사에서는 EIC 브랜든 셰필드씨와 편집부 직원들은 록 밴드(Rock Band)에서 파이널 판타지 XII(Final Fantasy XII)까지 이르는 게임이 개발되면서 가장 많이 반복되는 '실수'에 관한 문제를 모아봤다.]

비디오 게임의 제작과 더불어 게임을 개발한 팀이 직접 작성한 개발후기는 게임 개발자(Game Developer)잡지에서 십 년이 넘도록 계속 다뤄진 기사였다.

수 많은 저자들이 개발 과정에서 자신들이 잘한 것과 실수한 것을 밝히는 기사들을 살펴보면 일정한 양식이 나타난다는 것을 알 수 있다. 여러 회사에서 똑 같은 실수를 저지르고 있었으며 심지어 어떤 회사에서는 자신들의 실수를 반복하고 있다는 것이 명백히 드러난다.

이러한 점을 염두에 두고 우리는 지난 3 년 동안의 기사에서 모든 "실수들" 을 종합하여 가장 빈번하게 저질러지는 실수(보통 5 번 이상) 들을 조심해야 할 점으로 편집하기로 결정했다.

속담에서 알 수 있듯이 과거에서 교훈을 얻지 못하면 계속 실수를 반복하게 된다. 다음은 현재 가장 많이 일어나는 10 가지 "실수"를 순서에 상관없이 열거하였다. 이는 누구나 저지를 수 있는 실수이다.

1. 너무 늦게 추가된 콘텐츠.

스토리와 기능을 가장 알맞은 시기에 제대로 입수하기는 어렵지만 아슬아슬한 순간에 콘텐츠가 도입되면 콘텐츠의 질도 좋지 않을 뿐만 아니라 해당 콘텐츠에 의존하는 게임의 모든 요소들의 질도 나빠진다.

타이탄 퀘스트(Titan Quest) (Iron Lore, 제프 굿실(Jeff Goodsill)) "제 시간에 기술부에 25 가지에 이르는 상세한 디자인 문서를 넘기기 위해 고생했습니다. 프로젝트가 시작된 지 일 년이 넘어서도 첫 번째 컷의 설계문서를 작업하고 있었기 때문에 기술 팀들은 초기에는 아무것도 모르는 채 작업을 해야 했고, 몇몇 시스템은 다시 작업을 해야 했습니다.

"설계 문서가 기술적으로 구현 가능할 만큼 구체적이지 못하다는 문제가 꾸준히 제기되어 왔습니다."

이런 문제가 발생한 후에 회사에서는 프로듀서, 수석 디자이너, 기술 책임자, 제작사의 크리에이티브 메니저가 설계문서를 최종적으로 만들기 전에 각각의 문서에 서명을 하도록 하는 인가 과정을 고안하였다.

*툼 레이더: 리전드(Tomb Raider: Legend)*를 개발하면서 이와 비슷한 문제를 겪었던 릴리 쿠퍼(Riley Cooper)씨의 말을 들어보자. "게임에서 특정 기능이 제대로 구현되게 하려면 기능을 완벽하게 개발하든지, 아니면 아예 구현하지 않아야 한다는 사실을 알 수 있었습니다."

바이오쇼크(BioShock) (2K Boston, 앨리사 핀리(Alyssa Finley)) "개발 과정 중에 수 많은 스토리 초안이 있었지만 최종안은 거의 다시 작성해야 하는 것으로 밝혀졌습니다"

"시간과 자원을 배정받기 위해 경쟁하게 되는 것은 불행히도 게임에서 설명 기능이 있는 중요한 부분이 최종 수정본이 나올 때까지도 만들어지지 못함을 의미하기 때문에 기존의 게임에 이 기능을 다시 집어넣기 위해 많은 작업을 해야 합니다"

RPG 와 같은 게임에서 스토리는 프로젝트를 이끄는 핵심적인 존재이다. 이상적인 것은 각 시스템이 서로에게 영향을 주며 스토리에 완전히 통합되는 것이지만 완전한 양산체제에 돌입하기 전에 최소한 시나리오는 세부사항이 모두 확정되어야 한다. 대화로 이루어지는 *바이오쇼크(BioShock)* 와 같은 게임에서 구현과 파이프라인의 문제를 초기에 적절하게 다루지 못하면 마지막 순간에 이런 문제가 갑자기 불거져 나올 수 있다. 2K 보스톤 팀에서 일어났던 것처럼 말이다.

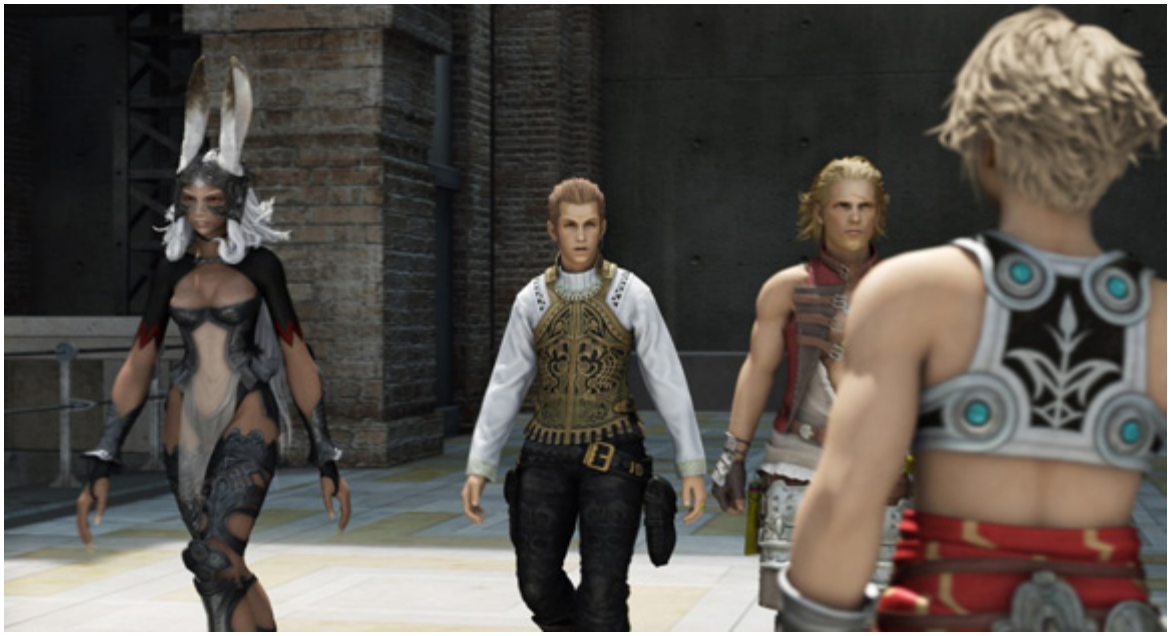
2. 의사소통

마감일이 다가오면 모두가 쫓기게 되어 의사소통을 소홀히 하기 쉽다. 물론 게임의 핵심인 완성도가 우선되어야 하지만 의사소통이 이루어져야만 완성도가 높아진다고 확신할 수 있다.

에이지 오브 부티(Age of Booty) (Certain Affinity, 맥스 호버만(Max Hoberman)) "프로젝트를 진행하면서 게임을 구상한 상태와 실제 구현된 상태가 일치하지 않은 부분이 여러 군데 있었습니다".

"가장 힘들었던 것은 중매나 파티 지원과 같은 복잡한 기능을 프로그래머가 크게 간소화(없애는 경우도 자주 있음)해 버린 지 한참 지난 후에도 이를 세부적으로 설계하고 있는 디자이너들이었습니다. 프로젝트에 대한 관심의 결여, 커뮤니케이션의 부재, 낙관적인 생각으로 인해 디자인 팀은 실제로 구현할 수 있는 것보다 훨씬 앞선 기능들이 구현될 것으로 믿으며 프로젝트가 많이 진행이 된 후에도 이런 현실을 깨닫지 못합니다"

이런 말이 Xbox Live Arcade 의 타이틀 개발후기에서 나왔으며 비교적 소규모 팀에서 지적되었다는 사실이 더 아쉽다. 회사는 동시에 여러 프로젝트를 진행하고 있었으며 이것이 커뮤니케이션의 갭을 더 키우는 결과를 낳았다. 그렇지만 의사소통을 해야 마감시간을 지킬 수 있을 것이다.



스퀘어 에닉스(Square Enix)의 파이널 판타지 *XII* (*Final Fantasy XII*)

파이널 판타지 XII (Final Fantasy XII) (Square Enix, 타쿠 무라타(Taku Murata)) "*파이널 판타지 XII(Final Fantasy XII)*를 개발하면서 성공에 대한 부담이 너무 커서 아주 작은 오해에도 이성을 잃곤 했습니다. 이 때 우리 팀은 다양한 개발 단계에서 자유롭게 게임을 변경할 수 있었지만 자유로 인한 대가로 잘못된 커뮤니케이션, 혼란, 무질서 등이 생겨났습니다. 작업이 어떻게 분배되어야 하는지가 애매한 경우도 자주 문제가 되었습니다"

관리 부담은 일본에서 특히 크게 나타나는 문제이지만 미국에서도 일어나지 않는다고 할 수는 없다. 관리자가 너무 많긴 하지만 진정한 관리는 이루어지지 않은 경우가 종종 있지 않은가.

3. 범위와 규모.

욕심은 양날의 칼과 같다. 한편으로는 창의력을 발휘할 수 있게 하지만 다른 한편으로 과욕으로 인해 프로젝트가 망가질 수도 있다.

개발자가 항상 스케줄을 결정하는 것은 아니지만 스케줄에 관한 문제는 개발자들의 동의를 얻어야 한다. 최선을 다해 스케줄을 지키고 게임의 범위를 합리적인 범위 내로 제한하는 것은 쉽지 않다. 그렇지만 이것이 가장 중요한 부분이다.

스트랭글홀드(Stranglehold) (Midway Chicago, 브라이언 에디(Brian Eddy)) "믿기지 않을 지 모르지만 스트랭글홀드는 원래 GTA 유형의 자유로운 게임으로 개발된 것이었습니다. 그렇지만 개발된 지 약 6 개월이 지난 후에 우리는 관리부서에게 이를 개발하는 것은 너무 오래 걸리며 우리가 작업하는 다른 모든 것을 진행하면서 Midway 의 첫 번째 차세대 게임을 개발하는 것은 너무 위험하다는 것을 설득하였습니다. 그런 후에야 게임을 보다 직선적인 샌드박스 스타일의 게임으로 줄일 수 있었습니다"

"그렇지만 게임에는 여전히 싱글 플레이어 게임, 월드 인터렉션, 4 가지 특수 움직임, 막대한 파괴력, 전투장치가 탑재된 차량, 확장 멀티플레이어 등과 같은 확장된 기능이 있었습니다. 우리는 개발을 진행하면서 공격적으로 각 기능의 범위를 줄였지만 기능 자체는 계속 유지했습니다. 개발한 지 약 1 년 후에 이 모든 기능을 AAA 품질로 완성할 수 없다는 사실이 분명해졌지만 그래도 계속 노력했습니다"

" 2 년이 되자 관리부서에서 지키기 어려운 마감일을 통보했습니다. 이로 인해 운전할 수 있는 차량을 포기하게 되었고, 나머지 기능의 품질을 높이는데 집중하게 되었습니다. 궁극적으로 차량을 포기한 것이 게임을 제 때에 출시하는 데 도움이 됐지만 이런 결정을

1 년 일찍 내렸더라면 더 많은 시간을 다양한 미션과 전체적인 품질을 높이는데 사용할 수 있었을 것입니다”

주주에게 사로잡힌 관리자들과 싸워서 얻을 수 있는 것은 많지 않다. 현실적인 목표를 제시하고 필요할 때는 기능을 포기하기로 결정하며, 나아가 애초에 관리할 수 있는 범위의 기능인지 확인하는 고위 관리직의 책임이 막중함을 알 수 있다.

기타 히어로(Guitar Hero) (Harmonix, 그레그 로피코로(Greg LoPiccolo)와 대니엘 서스맨(Daniel Sussman)) “프리스타일 모드를 지원해서 플레이어가 우리가 정말로 사랑하는 다이브밤 기술, 피드백, 핑거 태핑, 그 외의 청소년들의 과시하고자 하는 기타 연주기법을 지닌 광적인 솔로를 만들 수 있다는 것에 대해 흥분했었습니다.”

“ 이 기능을 개발하기 위해 소중한 시간과 자원을 많이 투자했지만 아쉽게도 이 기능을 포기해야 했습니다. 매우 야심 찬 계획이었지만 음질을 높이고 기능을 게임에 통합할 수 있는 시간이 부족했습니다. 일부에서는 모험을 걸만한 일이라고 생각하는 사람들도 있었지만 확신이 없었습니다”

시간과 규모의 적절한 관리는 일부 실수를 예방할 수 있지만 모든 경우에 해당되지는 않는다. 기능이 제대로 구현되지 않을 때 이를 포기하는 것과 시간이 문제일 때 이를 개선하는 것은 문제가 틀리다. 시간 문제로 인해 부분적으로 개발된 기능을 포기하는 것은 아무도 원하지 않는다.

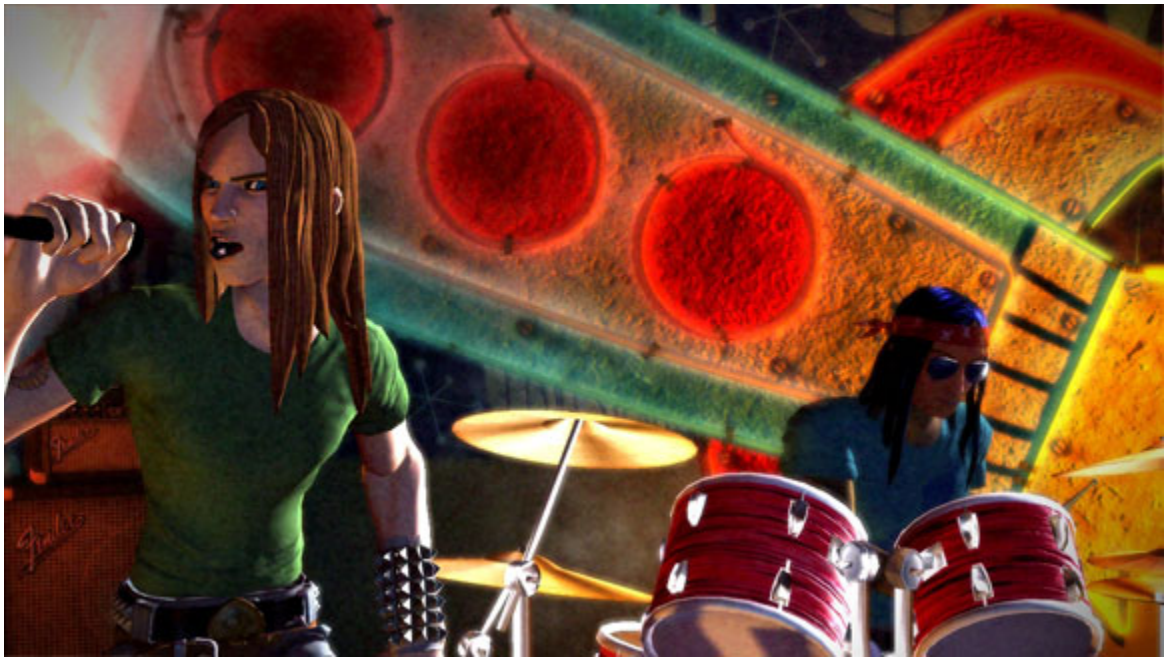
4. 고용.

게임의 개발에서 인력은 가장 비용이 많이 드는 요소이며 재능이 있는 인력을 발견하는 것 역시 가장 어려운 일이다. 과거의 개발후기에서는 고용을 주저하는 것뿐만 아니라 새로 고용한 사람을 기존의 문화와 방법론에 통합하는 어려움에 대해 공통적으로 언급하고 있다.

록 밴드(Rock Band) (Harmonix, 롭 케이(Rob Kay)) “게임 타이틀을 시간에 맞춰 완성하는 유일한 방법은 팀의 규모를 키우는 것입니다. 대부분의 직원을 게임에 투입하면서 회사의 규모를 새로운 수준으로 확장했습니다. 사무실이 터져나갈 듯이 가득 찼죠. 3 개 층에 직원들이 흩어져 있었으며 새로운 직원을 위한 공간도 없었습니다. 게임을 완성해야 했기 때문에 이를 견뎌냈고 그 후 더 큰 공간을 위해 회사 전체가 이사를 했습니다”

“그럼에도 불구하고 직원을 공격적으로 충분히 고용하지는 않았습니다. 소규모의 꽉 짜인 게임을 수 년 동안 만들면서 효율성에 대한 생각이 깊숙이 베어 있었기 때문에 ‘적은 것이

더 좋다'는 생각을 떨쳐버릴 수 없었죠. 관리부에서 직원을 더 뽑으라고 요구할 것이 두려웠고 이 때문에 우리 모두 더 힘들고 긴 시간을 보내게 된 것 같습니다"



EA / Harmonix 사의 *록 밴드(Rock Band)*

록 밴드 팀은 후에 더 고생하게 되었고 새로운 사람을 고용하여 기존 팀에 통합시키기에는 너무 늦었다는 바보 같은 생각을 하게 되었다. 그렇지만 결국은 베타 버전을 개발하면서 새로 직원을 고용하게 되었다. 이 때가 새로운 직원을 훈련시키기에 좋은 시기는 아니었지만 말이다.

크랙다운(Crackdown) (Realtime Worlds, 필 윌슨(Phil Wilson)) "또 다른 충격적인 현실은 고용해야 할 프로그래머의 수가 기하급수적으로 늘고 있다는 것입니다. 실질적인 개발 환경에서 선택할 수 있는 것은 다음 세 가지입니다. 1) 예산을 늘린다. 2) 욕심을 낮춘다. 3) 전기선을 뽑아버린다."

"불행히도 주주들은 실용적인 것과 거리가 멀고, 명백하게 정의된 '프로젝트 주요기둥'에 도움이 되지 않는 기능을 과감히 버리기 보다 범위를 줄이려는 논의를 시작하면서 또 다른 아이디어를 내놓곤 합니다"

프로젝트를 다시 평가할 때 원래의 목표대로 팀을 유지하는 것은 어렵다. 더 많은 직원을 고용하고 게임의 범위를 넓히는 것은 목적과 맞지 않는 것이다. 그렇지만 반대로 톰 레이더

리전드는 초기에 너무 많은 사공이 있어 문제가 되었다. 전반 작업에 팀의 규모가 너무 크게 되면 타이틀이 지향해야 할 방향에 45 명이 모두 동의하기가 힘들어진다.

5. 헛도는 프로젝트, 지도력 부재

이 두 가지 문제들은 결합되어 나타나는데 이들이 매우 유사한 결과를 낳기 대문이다. 직원들 층은 너무 얇고 너무 많은 책임을 짊어지고 있다. 이는 직원들이 종종 게임의 진척도와 게임의 느낌을 큰 안목에서 바라보지 못하는 결과를 낳는다.

에이지 오브 부티(Age of Booty) (Certain Affinity, 맥스 호버만(Max Hoberman)) "강력한 출발에도 불구하고 우리는 심각한 실수를 저질렀습니다. 이는 너무 많은 프로젝트에 초점이 너무 분산되었기 때문에 일어난 것으로 직원들을 어떤 게임에 관여하게 했다가 다른 게임에 관여하게 한 후 다시 돌아오게 하는 경우도 종종 있었습니다"

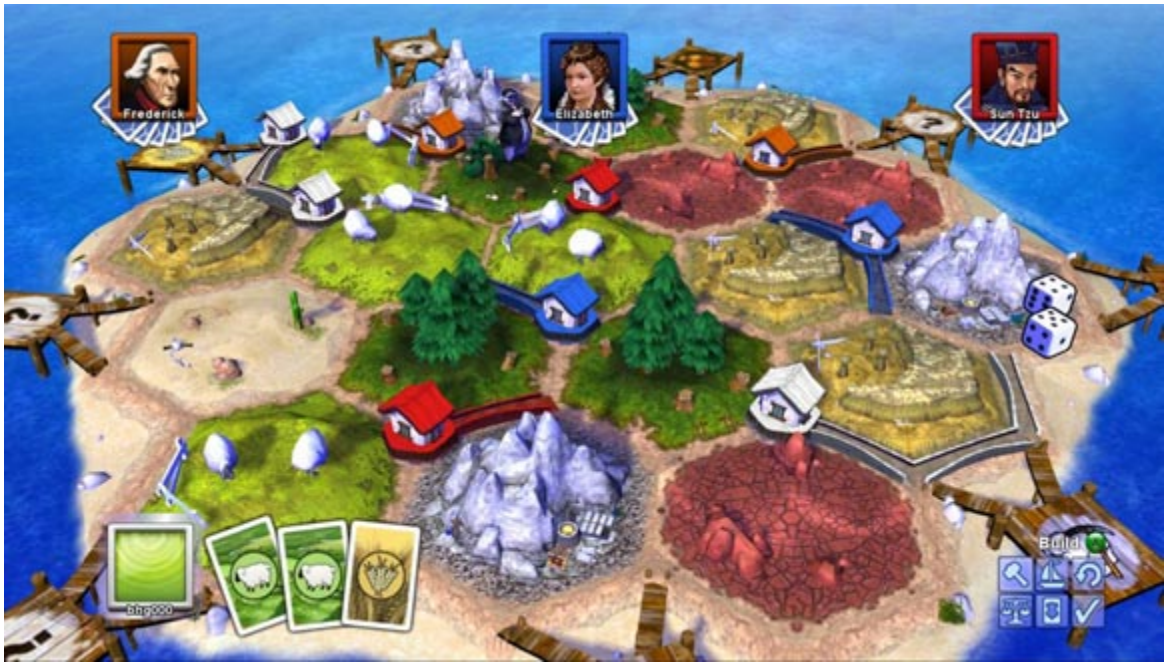
" 예를 들어 프로그래밍 측면에서 살펴 보면, 우리는 거래를 협상하고 최종적으로 계약이 체결될 때까지의 수 개월 동안 두 명의 프로그래머를 프로젝트에서 제외시켰습니다. 결국 이들을 대신 당시 또 다른 큰 프로젝트였던 레프트 4 데드(Left 4 Dead)를 처리하도록 두 명을 고용한 후에 이들을 원래 프로젝트에 다시 투입할 수 있었습니다.

" 그런 후 세 번째 프로그래머를 고용하여 두 직원을 지원하게 하고 동시에 일부 프로그래밍 작업은 하청을 주었습니다. 그런 후 또 정규직원을 고용하면서 계약직 프로그래머와의 작업을 종결했습니다. 계약직이 수행하던 작업은 레프트 4 데드의 프로그래머 중 비는 시간에 계약직이 하던 작업을 완수하겠다고 자원한 직원에게 넘겼습니다."

이런 현상은 계속되었다. Certain Affinity 사는 세 번째 프로젝트를 시작했고 자원한 직원을 다시 복귀시켰으며 결국 원래의 두 프로그래머를 다시 복귀시켰다. 소규모 프로젝트에서도 이런 프로그래머의 혼란스러운 상황이 게임 개발을 산만하게 만들 수 있다. Certain Affinity 사가 막 커지는 회사로써 들어오는 프로젝트를 거절할 수 없었을 것으로 이해는 되지만 이런 혼란이 없었다면 에이지 오브 부티의 품질이 더 높아졌을 것이다.

카탄(Catan) (Big Huge Games, 브라이언 레이놀드(Brian Reynolds)) "우리의 가장 큰 실수는 정규직의 수석 프로그래머나 수석 아티스트를 프로젝트에 배정하는데 실패한 것입니다"

"견고한 관리 구조를 갖지 못한다는 것은 틈새를 통해 일이 잘못될 가능성이 있다는 것을 의미합니다. 프로그래밍 팀이나 아트 그룹에서 목표를 설정할 사람이 없었죠. 일별, 주별, 또는 월 단위로 무엇을 해야 하는지 확신시켜주는 사람이 없었습니다. 직원들은 때때로 헤매면서 다음에 무엇을 해야 할지 확신을 갖지 못한 채 별로 중요하지 않은 작업에 너무 많은 시간을 쏟고 실제로 게임에 핵심적인 요소를 무시하곤 했습니다."



마이크로소프트 /Big Huge Games 사의 카탄(*Catan)*

개발자들이 관리직에게 불평이 많은 만큼 구조와 범위를 정해주는 지도자의 역할이 중요하다. 관리되지 않는다면 아티스트(한 예일 뿐이다)는 마지막까지 빈둥거릴 것이다.

6. 기술적 문서의 결여

이 문제는 상상했던 것보다 더 많이 일어났다. 그렇지만 적절한 코드 리뷰 작업을 통해 자신의 툴의 규모를 결정하는 것이 초기 단계에서 게임을 구축하기 위해 대단히 중요하다는 것은 당연한 이야기이다.

록 밴드(Rock Band) (Harmonix, 롭 케이(Rob Kay)) "*록 밴드*를 만들기 위해 프로그래밍 부서를 확장하는 것은 재능은 있지만 디자이너와 프로그래머 기능을 모두 갖추지 못한 프로그래머를 새로 고용하는 것을 의미했으며 이들이 디자이너와 프로그래머의 자질은 모두 갖췄다고 해도 그렇게 작업을 해도 괜찮은지 알지 못했습니다. 구현될 기능을 적절하게

테스트하거나 설계에서 예외 조건을 테스트하는 시간을 갖지 않고 새로운 시스템을 구현하는 데 곧장 뛰어드는 일이 너무 잦았습니다. 이런 현상이 몇몇 영역, 특히 온라인 중매 분야에서 걸림돌이 되었는데 결국 여러 번 다시 설계해야만 했습니다"

"의심의 여지가 없습니다. 복잡한 새로운 시스템을 미리 정해진 기술적인 계획 없이 바로 개발하는 것은 완전한 실수입니다"

Harmonix 사는 이를 새로운 시스템을 구현하기 전에 코드 리뷰 작업과 기술적 설계 문서 작성을 하도록 하여 해결하였다. 몇몇 경우에서는 이를 분명히 하고 있지만 개발후기에서 이 문제가 발생하는 빈도를 보면 이 문제가 모든 사람들의 우선순위의 순위권에 들어 있지는 않다는 것을 알 수 있다.

인디고 프로페시(Indigo Prophecy) (Quantic Dream, 데이비드 케이지(David Cage)) "우리가 저지르는 또 다른 고전적인 실수는 만들려고 하는 게임 전용의 툴을 만들기 보다 미래에 쓰일 것을 염두에 두고 포괄적인 툴을 개발하려고 노력하는 것입니다. .

"포괄적인 툴은 아주 다양한 경우를 관리할 수 있지만 어느 것 하나도 매우 효율적으로 다루지는 못합니다. 미래의 제작현장에서 툴을 재사용한다는 것은 단기간에 시간과 비용이 소모되는 허망한 희망사항이며 장기적으로 이익이 될 것이라는 아무 보장도 없습니다."

Quantic Dream 은 겨우 중간규모의 프로젝트와 이에 맞춘 툴을 실현했지만 어느 정도의 작업 손실이 있었다. 프로젝트의 특정한 요구에 관해 적절한 기술 문서를 갖는 것은 확실히 도움이 된다..

7. 아웃소싱.

게임 개발에서 아웃소싱은 점점 더 많이 일어나고 있다. 아웃소싱은 비용을 절약할 수 있지만 제대로 수행되기가 힘들며 전송용량이 커야 한다.

샘 앤 맥스(Sam & Max) (Telltale Games, 텔테일 팀(Telltale Team)) "일단 어느 계약자와 일할 것인지 결정된 후에도 우리가 원하는 것을 정확하게 설명하기 위해 이들 업체를 여러 번 방문해야 했습니다. *샘 앤 맥스*의 스케줄이 매우 빡빡했기 때문에 계약업체의 능력 부족이나 의사소통의 어긋남으로 인한 실수나 이의 수정방법에 대해 설명할 시간이 많지는 않았습니다. 예를 들어 계약 스튜디오가 A 팀을 보내고 계약이 체결되자 우리가 합의했던 수준보다 떨어지는 B 팀을 보내왔을 때 정말 힘들었습니다"

"계약회사가 국내에 있지 않을 때(지구 반대편의 회사와 계약하기도 했습니다) 의사 전달이 더 힘들었습니다. 우리가 한밤 중일 때 계약회사가 일하는 시간일 경우에는 보통 10 분이면 되는 대화를 이틀에 걸쳐서 하게 됩니다"

마감을 지키는 것은 계약사에게 대단히 중요한 것이지만 A/B 팀과 같은 문제를 예측하는 것은 힘들다. 이런 문제는 *스텁스 더 좀비(Stubbs the Zombie)* 팀에서도 지적된 사항이다. 이들의 이야기를 들어보자.

스텁스 더 좀비(Stubbs The Zombie) (Wideload Games, 알렉산더 세로피언(Alexander Seropian)) "우리는 계약사가 작업을 제출하는 데 얼마나 걸릴지에 대해 과소평가했습니다. 작업이 시간이 걸릴 것이라고 알고는 있었지만 이런 예상에도 불구하고 아트 감독과 아트 제작의 결합은 우리가 갖고 있는 시간보다 더 많은 일이 필요했습니다"

"프로듀서가 부족했고 우리의 아티스트들은 자신들의 콘텐츠를 제작하도록 되어 있어서 제시간에 맞추어 제출된 결과를 검토할 여유가 없었습니다. 제작 단계에서 계약사가 보내준 작품을 더 자세하게 살펴보아야 한다는 것을 깨달았지만 너무 늦었던 거죠. 계약사의 피드백에 우리의 내부적인 힘을 쏟는 것이 우리 아트 감독 팀의 우선순위에서 더 높은 순위를 차지했어야 했습니다."

아스파이어(Aspyr)/Wideload 사의 스텁스 더 좀비(*Stubbs the Zombie*)

여기서 우리는 수석 관리 팀의 부담을 살펴볼 수 있다. 계약사를 관리하는 전담 팀이 없다면 수많은 작업이 사용되지 않게 된다. Widelod 사에서는 팀 일부에서 이런 역할을 전담하게 되면서 후속 제작에서는 작업이 조금 더 부드러워졌다고 한다.

8. 끝마무리.

마무리 단계에서의 시간 부족은 세부사항이 정교하지 못하거나 플레이어의 기대를 충족하지 못하는 게임이 탄생됨을 의미한다.

타이탄 퀘스트(Titan Quest) (Iron Lore, 제프 굿실(Jeff Goodsill)) "우리는 게임의 균형을 맞추고 끝마무리하는데 6 개월이 적당하다고 말하곤 했습니다. 세 가지 난이도 수준 별로 각각 30 시간이 넘게 게임을 하면서 계획된 시간을 다 쓰고도 시간이 더 필요했습니다. 수천 개의 장치와 수십만 개의 변형 장치는 말할 필요도 없거니와 모든 기술과 전문지식을 조합하여 기술 시스템의 균형을 잡는 것만으로도 대규모 작업이었습니다."

"마지막에는 3 개월밖에 시간이 없었습니다. 다행히도 게임의 모양새를 갖출 수 있게 됐지만 시간이 모자라서 구현하지 못했던 기능들도 수없이 많이 있었습니다"

팀은 타협을 해야 했고 게임의 전반부에만 초점을 맞추고 보다 난이도가 있는 수준의 마무리는 프로젝트의 거의 마지막에 가서야 할 수 있었다. 플레이어가 계속 게임을 즐기게 만드는 방법이 아닌 것은 확실하다.

건(Gun) (Neversoft, 스콧 피스(Scott Pease), 채드 핀들리(Chad Findley)) "너무 짧은 게임보다 나쁜 것이 무엇이 있을까요? 너무 짧고 너무 쉬운 게임일 것입니다. 빡빡한 스케줄과 해당 장르에 대한 미숙함으로 인해 게임 난이도에 대해 아주 단순하게 접근했으며 게임 앞에 '쉬움, 보통, 어려움, 또는 미친 듯이 어려움'이라는 선택사항을 붙여놓았습니다. 그런 후 뒤로 물러 앉아 플레이어가 자신의 입맛에 맞게 적절한 난이도를 고르도록 하였습니다. 무슨 헛소리를 했던 것일까요."

"비디오 게임 시장에서는 플레이어에게 게임을 하기도 전에 난이도를 정하라고 하는 것은 아주 아마추어적인 생각입니다. 리뷰에서 나타난 반응 중 중요한 것은 잘못된 난이도에서 게임을 하는 사람들이 너무 많았다는 것이었습니다. 이건 저희의 잘못이었죠"

Neversoft 사의 팀이 난이도 테스트에 초점을 맞췄지만 게임을 하기 전에 난이도를 선택하는 것은 많은 플레이어들에게 부담이 된다. 이는 시간이 많이 있으면 해결되는 또 다른 문제이다. 그렇지만 어떻게 시간을 벌 수 있는지가 더 큰 문제이다.

9. 빈약한 툴 구현.

게임을 위해 툴을 구축하는 것은 끔찍하게 힘들고 이 툴의 사용법이 분명하지 않거나 분수에 맞지 않게 클 때 문제가 더 커지게 된다. 이는 기술 문서의 결여와 비슷한 문제로, 그만큼 많이 일어나기도 한다.

레지스탕스: 인류몰락의 날(Resistance: Fall of Man) (Insomniac, 마커스 스미스(Marcus Smith)) "자체적으로 만든 툴과 기술의 상반되는 측면은 툴이 신속하게 바뀔 수 있다는 점과 모든 변화에 대해 직원들을 적절하게 교육시키는 것이 불가능하다는 것입니다. 어떤 기능을 구축하면서 이를 만드는데 필요한 툴을 동시에 만드는 것은 모래 위에 집을 짓는 것과 같습니다.

"아티스트가 툴을 열고 나서 새로운 인터페이스 버튼을 발견해도 이것이 무엇을 위한 것이었는지 어떻게 사용하는 것인지 알지 못했습니다. 구축된 툴의 변화로 인해 많은 자원이 다시 구축되고, 다시 조명되어야 하며 다시 애니메이션으로 만들어져야 했습니다."

"혼란이 생긴다는 것 외에도 소수의 프로그래머만이 이 문제를 해결할 수 있는 지식을 갖고 있기 때문에 이들은 여기저기서 요청되는 도움요청을 해결하기가 버거웠습니다. 초인적인 노력과 키보드 위에서 보낸 오랜 시간이 없었다면 출시 일자를 지킬 수 없었을 것입니다"

Insomniac 사는 이런 문제로 인해 어마어마한 노력을 들여 겨우 안정된 빌드를 유지할 수 있었다. 툴을 접근할 수 있게 만드는 것이 절대적으로 필요하다.

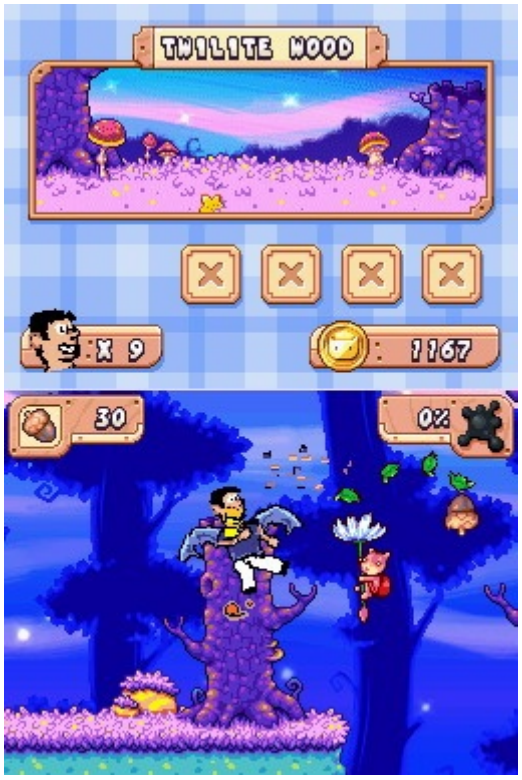
언차티드: 엘도라드의 모험(Uncharted: Drake's Fortune) (Naughty Dog, 리처드 르막셩(Richard Lemarchand)) "툴을 접근하는 방식에서 너무 많은 것을 잘못했다는 것을 깨달았습니다. 너무 지능적이 되려고 노력한 결과 각각의 종류의 툴에서 겪었던 모든 문제를 해결할 수 있는 복잡한 방식을 만들려고 했습니다. 더 나쁜 것은 고고한 열망에 사로잡혀 사람들이 실제로 수행하고 해당 레벨을 게임할 수 있게 하는 파이프라인 구축작업을 아주 늦게까지 하지 않았다는 것입니다"

"우리가 알게 된 것은 툴의 목표를 높게 잡는 것이 좋을지라도 선택을 해야 하며 팀에서 일어나는 모든 툴에 관한 문제를 해결하려고 노력해서는 안 된다는 것입니다. 어떤 경우에서는 간단한 차선책이 더 좋을 때도 있으며 이로 인해 게임 자체에 대한 작업에 더 많은 시간을 투자할 수 있습니다."

이런 문제는 기술을 공유하려는 일념에서 비롯된 것이기 때문에 목적 자체는 훌륭하다. 그렇지만 교훈은 똑같다. 바로 프로젝트의 목표가 완벽하게 달성될 때까지 개발범위를 제한하라는 것이다.

10. 너무 많은 작업량

이제 피할 수 없는 문제인 작업량에 관해 언급하겠다. 모든 사람이 이것이 문제라는 것을 알고 있지만 계속해서 이렇게 작업을 하고 있다. 너무 많은 작업량은 종종 위의 목록에 있는 다른 대부분의 문제의 부작용으로 인한 것이며, 이로 인해 역으로 원래의 문제를 악화시키는 악순환을 야기한다. 어떤 스튜디오는 이 문제를 해결하기 위해 고유한 방법을 사용하기도 하지만 다른 스튜디오에서는 욕심과 돈, 기타 여러 요인으로 인해 똑 같은 실수를 저지르고 있다. 이는 결코 좋은 것이 아니다.



드론투라이프(Drawn to Life) **(5th Cell, 죠셉 트린갈리(Joseph Tringali))** "항상 야근하는 것은 즐겁지 않고 게임을 개발할 때 꼭 야근을 해야 하는 것은 아니지 않을까요. 프로젝트 개발 후반부 내내 주중과 토요일까지 늦게까지 일해야 했습니다"

"우리 스튜디오에서는 플랫폼에 대한 경험의 부족, 욕심이 앞서는 게임, 너무 빡빡한 스케줄 등의 모든 문제들이 폭발하면서 작업량이 늘게 되었습니다. 이런 문제가 소홀했던 전반 작업과 결합되면서 게임에서 요구되는 기능을 구현하기 위해 수많은 단축키를 눌러가면서 첫날부터 스케줄을 따라 잡기 위해 애썼습니다"

첫날부터 스케줄을 따라잡아야 했다는 것이 여기서 핵심적인 구절이다. 적절한 스케줄과 제대로 된 프로젝트 관리로 이를 해결할 수 있다. 물론 행동으로 옮기기가 힘들지만 말이다.

스텁스 더 좀비(Stubbs the Zombie) **(Wideload Games, 알렉산더 세로피언(Alexander Seropian))** "(계약업체가 있었지만) 우리는 3 개월 내내 작업했는데, 이전의 경험에 비하면 그렇게 나쁘진 않았지만 좋지는 않았죠."

“주요 구성요소 중 일부를 후반 작업으로 넘겨버리고도 스케줄이 4 개월이나 밀려있었기 때문에 계약사보다 뒤떨어질 수 밖에 없었고 후반 작업을 더 많이 해야 하는 결과를 낳았습니다”

이 예를 통해 계약회사가 꼭 막대한 작업량을 없애주는 것은 아니라는 것을 기억하길 바란다. 제대로 된 후반 작업은 매우 유용하지만 다른 작업단계가 늦어졌을 때 이를 보완하는 시간으로 이용되어서는 안 된다.

잘못된 점의 공유

개발과정에서는 늘 똑 같은 문제가 불거져 나올 것이다. 직원들은 오래된 작업 스타일을 버리지 못하고 모범적인 방법을 따르지 못한다

Scrum 과 같은 방법론에 대해서도 평가를 해야 되지만, 프로젝트를 간결하게 관리하는 것은 치명적인 문제를 일으키기 전에 문제점을 확인하는데 확실히 도움이 된다. 이런 문제점들이 계속 공유되었으면 하고, 여러분이 우리 잡지사에 편지를 보내 이들을 Game Developer 잡지에서도 공유할 수 있길 바란다.