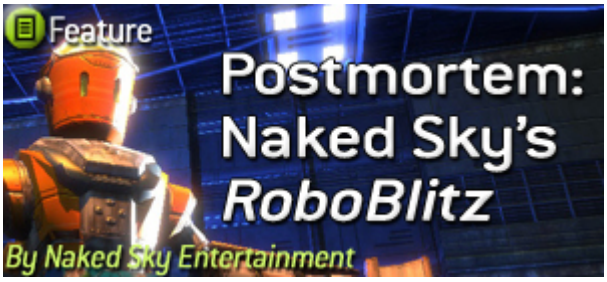
※ 본 아티클은 CMP MEDIA LLC와의 라이선스 계약에 의해 국문으로 제공됩니다

Gamasutra.com

사후 검토: Naked Sky Entertainment의 *RoboBlitz*

Naked Sky Entetainment
2007년 1월 17일
http://www.gamasutra.com/features/20070117/nakedsky_06.shtml

들어가며



사람들은 우리에게 "어떻게 Naked Sky 라는 작은 규모의 독립 게임 개발사가 Xbox 360 Live Arcade 로 그처럼 멋진 게임을 만들어낼 수 있었느냐?"고 물어오곤 한다.

모든 것은 Intel 듀얼 코어 프로세서용 테크놀로지 데모의 개발에서 시작되었다. 8 주간의 개발 기간을 거쳐 2005 IDF 와 GDC 에 선보인 *RoboBlitz Tech Demo(RoboHordes* 라고도 불렸다.)는 Unreal Engine 3 를 기반으로 하고 있었으며, 게임은 오로지 물리적 효과만을 이용하여 진행되도록 구성되어 있었고 단일 레벨로 이루어져 있었다.

그런데 GDC 에서 만난 사람들은 우리에게 "이 게임은 어디서 배급하죠?" "언제 출시되는 거죠?" 등의 질문을 해왔고 그 때서야 우리는 이것을 하나의 완성된 게임으로 개발할 필요가 있음을 깨달았다. 하지만 안타깝게도 원작 게임은 테크 데모를 위해서 개발되었기 때문에 상업용 게임으로 확실히 만들기 위해서는 거의 모든 코드, 아트, 디자인을 제거하고 다시 제작해야만 했다.

RoboBlitz

배급사:
Microsoft (Xbox 360)
Naked Sky Entertainment (PC)

개발사:
Naked Sky Entertainment

출시 플랫폼:
Xbox 360 Live Arcade, PC

개발자의 수:
~12

개발 기간:
11 개월

출시일:
2006 년 11 월 (PC)
2006 년 12 월 (Xbox 360)

게임을 어떻게 배급할 것인지에 관해서 몇 가지 문제점에 봉착했다. 우리는 독립 개발사로 머물러 있는 동안 작품에 대한 지적 재산권을 관리하고자 했기 때문에 기존의 배급 방식을 택할 수가 없었다. 그렇지만 다행스럽게도 우리는 Microsoft 의 Live Arcade 팀과 만날 수 있었고, 그들은 마침 Xbox 360 의 출시를 앞두고 앞으로의 청사진을 짜는 초기 단계에 돌입해 있었다. 작품의 용량을 50MB 이하로 맞출

수만 있다면 Xbox 360 은 *RoboBlitz* 에게 있어 완벽한 플랫폼이 되어줄 것이 틀림없었다.

우리는 자체적인 자금 조달 방식을 택하기로 마음을 먹었기에 개인 투자 금액을 늘리고 사무실을 잡은 후인 2005 년 11 월까지 본격적인 작업에 들어가지 않았다. 게임은 11 개월 후에야 완성되었다. 인디 개발사로써 우리는 개발 과정에 있어 수많은 문제에 봉착했었지만 그것을 헤쳐나가는데 성공했고 *RoboBlitz 를* 자랑스럽게 탄생시켰다!

무엇이 적절했는가

훌륭한 팀

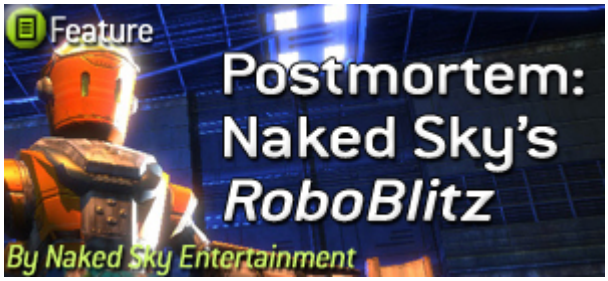
Naked Sky 의 가장 소중한 재산은 바로 사람들이다. 우리는 올바른 사람과 일하는 환경에서는 *무엇이라도* 해낼 수 있을 것이라 믿는다. 이것은 천재 혹은 수많은 경험을 지니고 있는 사람을 말하는 것이 아니라 열정적이고 재능이 있으며, 헌신적이고 성실하며 착한 사람을 말하는 것이다. 간단하게 말하자면 가능성이 큰 사람을 말하는 것이다. Naked Sky 팀의 사람들은 모두가 이러한 자질을 지니고 있으며 그 이상의 것을 지니고 있기도 한다.

우리 팀의 절반 가량은 대학에서 게임 미술 및 프로그래밍을 전공하고 최근 졸업한 사람들로 이루어져 있다. 이것은 재정 부문에 도움을 주었는데, 작은 규모의 인디 개발사로는 업계의 베테랑에 대한 보수를 지급할 수가 없기 때문이다. 물론 훌륭한 리더 없이는 팀을 올바르게 이끌어 나갈 수 없지만, 우리에게는 운 좋게도 모두를 하나의 방향으로 이끌어줄 경험 있고 박식한 리더들이 있었다. 이들은 신입사원들을 훈련시키며 새로운 툴과 기술을 시연하거나 다양한 문제 해결 방법을 설명하는 일에 직접 참여하였고, 이러한 훈련을 통해 사원 모두가 문제를 스스로 해결할 수 있고 위급하거나 갑작스러운 상황에 대한 팀의 대처에 일조할 수 있는 능력을 키울 수 있었다.

우리 팀은 단순한 회사가 아닌 가족과 같은 매우 각별한 관계였는데, 우리는 이러한 관계를 *패미글리아(famiglia)*라고 불렀다. 우리는 언제나 서로를 믿고 신뢰했으며, 일과 놀이를 통해 우애를 다졌다. 탁구와 *스트리트 파이터 2* 를 매일 같이 즐기고 주요 마일스톤의 완수를 자축하기 위하여 야유회를 가기도 했다. 끊임없이 "성장하는 우리 아이들"을 위해 몸에 좋은 코스트코 스낵을 잔뜩 쌓아 놓기도 했다.

팀원 모두가 함께하며 우리는 즐거운 시간을 가졌고 문제들을 해결해나갔다. 우리 팀의 대우는 좋았고 모두가 함께 행복하게 일을 했기 때문에 마침내 마법과 같은 일이 이루어진 것이다.

혁신이 가지는 위험

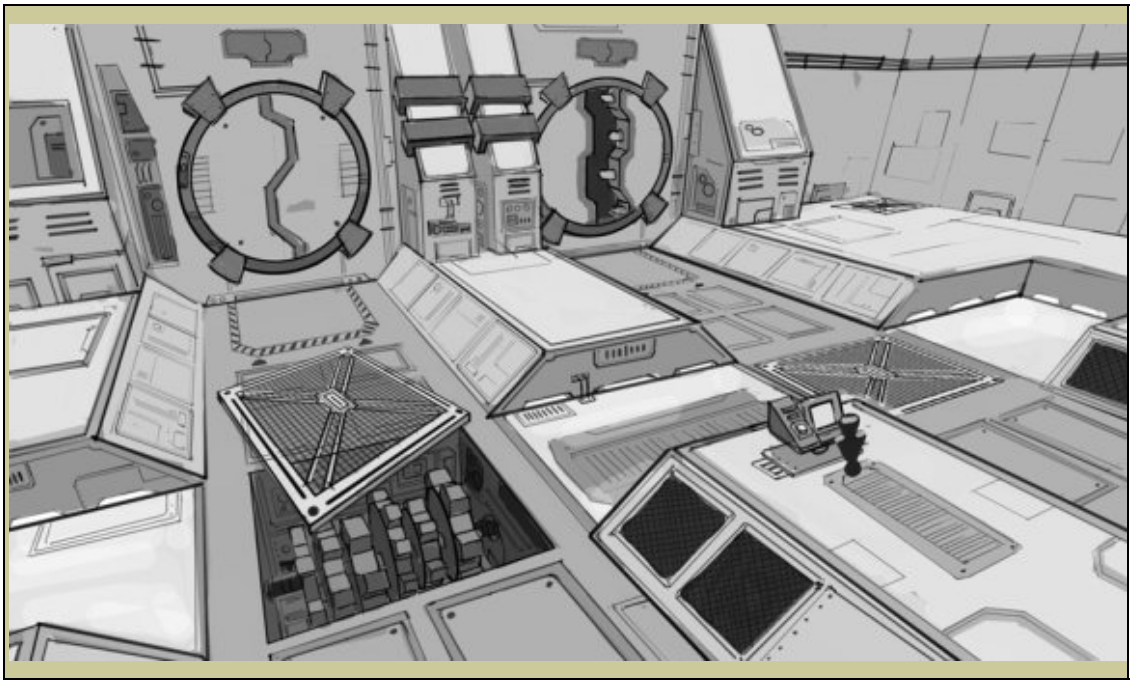


우리는 *Roboblitz* 를 만들며 수많은 기술적 도박을 했는데, 이러한 도박은 운 좋게도 모두 먹혀 들었다. 초기에는 업계의 친구 몇몇은 완전히 물리적 효과에 기반한 캐릭터가 등장하는 게임을 신생 팀이 제작하는 것은 불가능하다고 말했었다. 그렇지만 수많은 힘든 작업을 거쳐 우리는 그것을 해내는데 성공했고, 이것은 게임 내에서 움직일 수 있는 물체라면 그 어떤 것이라도 두 개를 연결할 수

있는 P2P 광선(P2P Beam)과 같은 혁신적인 무기를 만들어 낼 수 있도록 해주었다.

모든 캐릭터가 물리적 효과에 기반하여 움직이는 덕분에 P2P 광선을 이용해 나쁜 적 3 대를 하나로 붙여버리고 놈들이 거대한 지렁이처럼 꿈틀거리며 움직이는 것을 바라보며 적을 하나하나 처리하는 등의 행동이 가능했으며, 놈들을 거대한 분쇄기에 던져버리거나 다른 위험한 장소에 날려버리는 것도 가능했다. P2P 광선을 비롯한 이 게임의 여러 독창적인 무기와 장치들은 물리적 애니메이션 시스템이 없이는 가능하지 않았을 것이다.

우리는 또한 최초로 Unreal Engine 3 를 사용해 출시된 게임으로써 언론의 많은 주목을 받았는데 이것은 또 다른 심각한 기술적 문제를 가져왔다. 게임 개발과 UE3 개발 과정이 같은 시간대에 속해있었다는 것은 결코 사소한 문제가 아니었다. 엔진은 훌륭했지만 우리가 1 년 전 *RoboBlitz* 의 개발을 시작할 무렵 엔진은 전체 기능을 발휘하지 못 하고 있었던 것이다. 우리는 필요한 기능이 있을 때면 그 기능을 Epic 이 내놓는 것에 의존하여 필사적으로 개발을 진행하였다. 가끔 우리는 개발을 기다리기 보다는 특정 기능을 직접 개발해 넣기도 했는데 마지막에는 이것 역시 위험을 감수할 가치가 있었음이 밝혀졌다. 마케팅 예산은 전혀 배정하지 않았지만 최초의 UE3 사용 게임이라는 점은 좋은 광고 수단이 되었다.



마침내 수많은 트릭과 기술을 이용하여 19 레벨의 차세대 게임을 50MB 이하로 유지하며 출시할 수 있었다. 시험조차 거치지 않은 *ProFX* 를 게임에 넣는 것부터 시작해서 모든 에니메이션이 키프레임 없이 이어지도록 하는 것(이것들은 매우 압축되어 있었다), 패키지 압축 루틴을 향상시키는 것, 음악의 반복 구간을 제대로 맞추는 것, 아티스트들의 훌륭한 쿵푸 실력까지 많은 것들이 게임을 50MB 이하로 유지하는데 커다란 노력을 필요로 하였다.

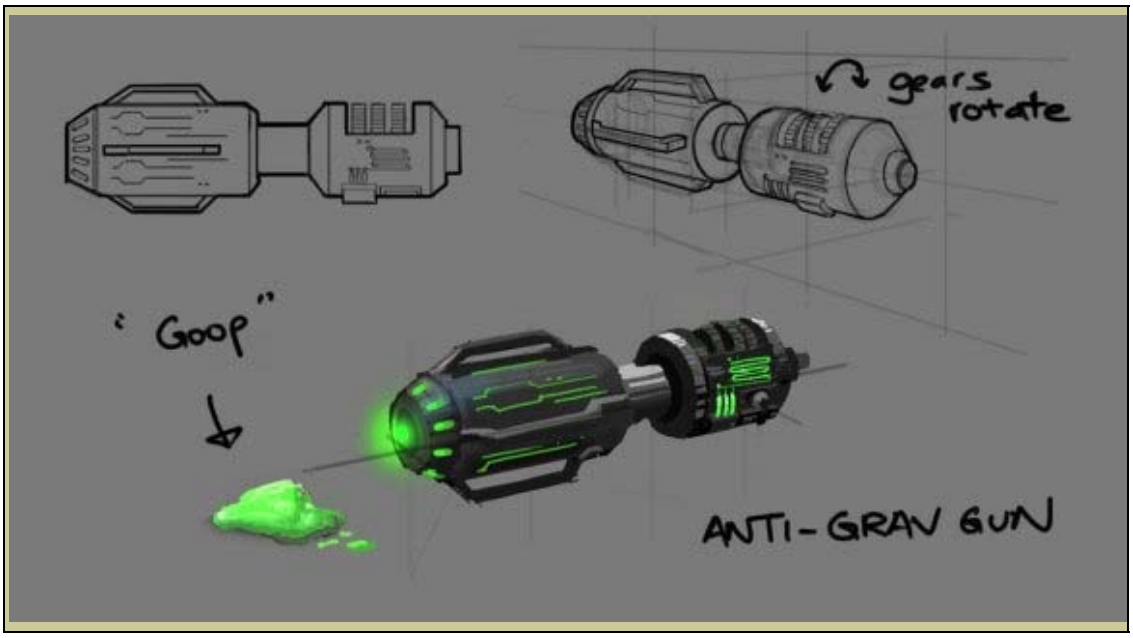
개발 시작 당시부터 이러한 일들을 계획한 것은 커다란 도움이 되었지만 우리가 정말 해낼 수 있을까 하는 의문이 드는 순간들이 분명히 존재했다. 마침내 최종 패키지를 완성하고 그것을 50MB 이하로 내놓는데 성공한 마지막 날에야 크게 안심할 수 있었다. 게임을 개발하며 느낀 걱정의 대부분은 Xbox Live Arcade 로 게임을 내놓는 것에 관련되었었는데, 특히 파일들의 크기를 줄이는데 들이는 노력들이 보상될 수 있을까 하는 것이었다.

훌륭한 기술의 라이선스

우리가 이 프로젝트와 관련하여 이루어낸 가장 현명했던 결정 두 가지는 바로 Epic Games 의 Unreal Engine 3 를 라이선스 받는 것과 Allegorithmic's ProFX 절차적 텍스쳐 생성 툴을 사용하는 것이었다.

UE3 는 말 그대로 아름다웠다. 차세대 게임을 지원할 능력을 지니고 있을 뿐만 아니라 그 어떤 방식으로도 수정하거나 확장할 수 있도록 디자인되어 있었다. 우리 프로그래머들은 전부 그래픽 부분에 있어선 전문가가 아니었기 때문에 UE3 를 라이선스 하는 것은 그래픽적으로 우리에게 한 발 앞서 경기를 시작하는 것과 같은 효과를 가져다 줬다.

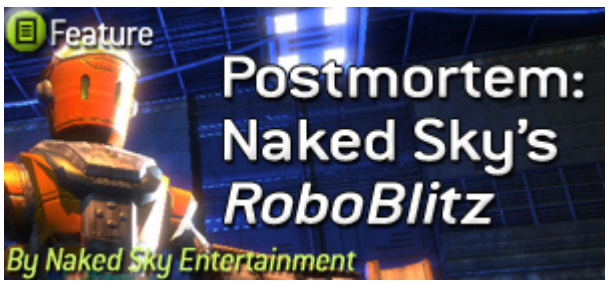
그렇지만 엔진을 훌륭하게 만드는 것은 단지 엔진이 지닌 기능만이 아니라 그 뒤의 팀이었다. Epic 의 기술자들은 모두 친절하고 협조적이었고, 우리와 함께 일하는 것을 정말로 즐거워했다. UE3 를 이용하여 완전히 물리적 효과에 기반을 둔 게임을 개발하는 것은 엔진에 커다란 수정을 요구했는데, 우리가 건드리고 있는 부분에서 세부 사항에 대해 질문을 할 것이 생기면 언제라도 Epic 의 누군가는 당장이라도 우리를 도울 수 있도록 준비가 되어 있었다.



Allegorithmic's ProFX 절차적 텍스쳐 생성 툴은 UE3 에 비하면 크게 알려지지 않았지만 *RoboBlitz* 를 Xbox Live Arcade 로 내놓는데 있어 도움이 되었었다. 앞서 말했듯 Live Arcade 로 내놓는 게임은 50MB 의 용량 제한이 존재하는데, 절차적 텍스쳐 생성을 거치지 않고는 *RoboBlitz* 의 모든 텍스쳐를 이 제한에 맞춰 넣을 방법이 없었기 때문이다.

ProFX 를 사용하여 우리는 각 레벨의 텍스쳐 중 80 퍼센트를 불러오기 시간 중에 생성해낼 수 있게 되었고, 이 덕분에 레벨을 장식할 텍스쳐를 더 많이 사용할 수 있게 되었다. 이러한 기술은 3D 로 이루어진 차세대 수준의 19 개 레벨을 포함한 게임을 50MB 이하로 줄이는데 아주 큰 도움이 되었다. Epic 처럼 Allegorithmic 팀 역시 매우 협조적이고 박식했기 때문에 그들과 함께 일하는 것은 정말 멋졌다.

자체적 금액 조달



자신의 자산을 투자하여 게임을 만드는 것은 소심한 사람에게는 권하고 싶지 않은 일이다. 특히 그것이 이제껏 상업용 게임을 출시해본 적이 없는 팀이 내놓은 아이디어와 독창적 지적 재산을 기반으로 아직

개발이 끝나지 않은 최신의 기술을 이용하여 제작되고 있는 차세대 게임이라면 더더욱 권하고 싶지 않은 일이다. 그렇다면 우리는 왜 이러한 일을 했을까?

답은 간단하다. 오직 우리만이 원래의 구상에 어긋나지 않게 *RoboBlitz* 를 제작할 수 있기 때문이었다. 이 프로젝트의 독창적 구상을 완료하기 위해서 우리는 자체적으로 금액을 조달할 필요가 있었다. 또한, 이러한 위험 부담이 존재하는데 우리에게 접촉할 배급사가 존재할 리가 없기 때문이기도 하였다.

자체적으로 금액을 조달하는 것에 있어서 장점은 게임을 원하는 만큼 얼마든지 자유롭게 창조적이고 독특하게 만들어낼 수 있다는 것이다. 도대체 어떤 배급사가 거대한 금속 공 위에서 굴러다니는 오렌지색 로봇이 등장하는 게임에 투자를 하겠는가? 사람이 직접 제작하곤 했던 애니메이션들을 모두 물리적 효과에 기반한 움직임으로 변경하는 것이 과연 받아들여질 것인가? 이러한 것도 문제가 없다고 생각한다면 이건 어떨지 생각해보자. 만약 우리가 1 년 전 Unreal Engine 3 를 이용하여 19 개의 레벨을 포함한 게임을 50MB 이하로 제작한다고 말했다면 누가 그 말을 믿었을까?

완성되지 않은 최신 기술을 이용하여 개발을 진행하는 것은 언제나 개발 지연으로 이어졌고, 이 기술을 새로운 빌드에 적응하기 위하여 디자인/아트/코드를 갑작스럽게 변경하는 것은 더 긴 개발 지연으로 이어졌다. 우리가 배급사의 투자에 의존했었더라면 *RoboBlitz* 는 오래 전에 취소되었을 것이다. 그렇지만 이 게임은 우리가 힘들어하며 낳은 아기였고, 마침내 49.32MB 의 용량으로 건강하게 태어났다!

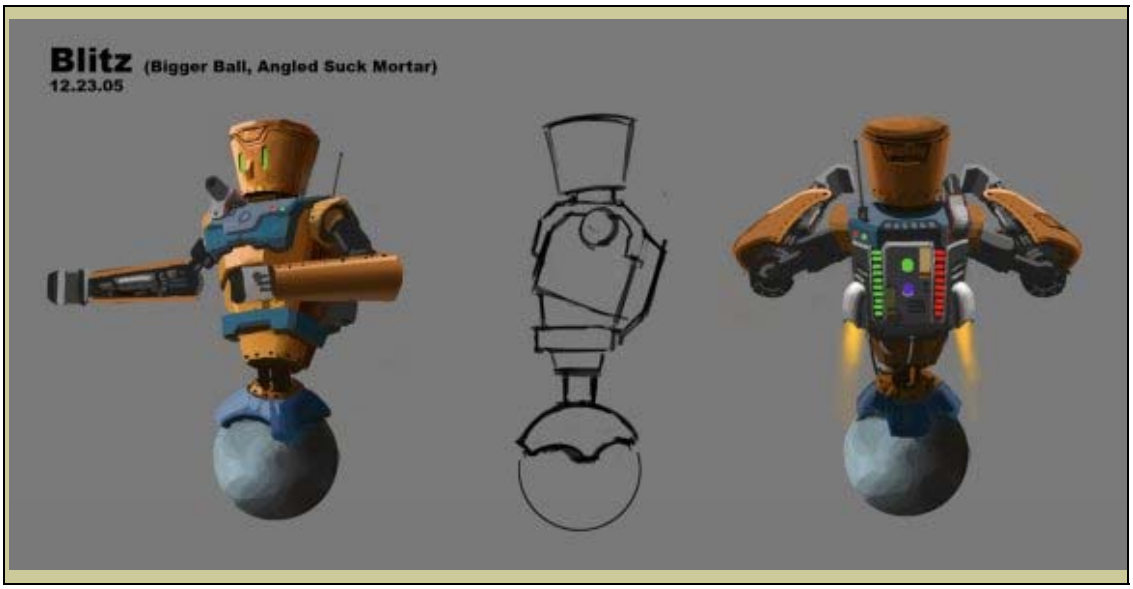
마지막으로 언급하자면, 우리가 자체적으로 금액을 조달했기에 배급사와 수익 배분을 할 때 우리가 큰 몫을 지닐 수 있기도 했다.

커뮤니티 지원

출시하기 대여섯 달 전, 우리는 *RoboBlitz* 의 웹사이트를 열어 유저들로 하여금 게임에 관하여 토론하고 질문을 할 수 있는 장을 만들었다. 이것은 사람들의 관심을 끌어 모을 수 있었는데, 게임의 출시 후 기술적 문제를 빠르게 포착할 수 있는 기회였을 뿐만 아니라 극도로 제한된 우리의 자본으로 유저 기반을 쌓을 수 있는 기회이기도

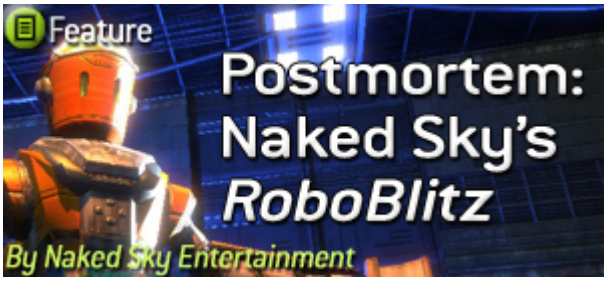
하였다. 출시가 이루어지고서 우리는 그러한 유저 기반을 유지하는데 심혈을 기울였고, 기술 팀의 3 할을 기술 지원 E 메일과 포럼 관리 담당으로 배정하기도 하였다.

RoboBlitz 의 에디터를 내놓고 에디터 위키를 사이트에 설치하는 것 역시 튼튼한 커뮤니티를 만드는 단계의 일부였으며 이 역시 수월하게 진행되어 갔다. 현재 우리의 위키는 여전히 UE3 기반의 에디터에 관한 상세한 정보를 제공하는 최고의 공개 소스이며, 이는 *Roboblitz* 에 많은 관심을 끌어내고 판매로 이어지기도 하였다. 우리는 모드 제작 커뮤니티에도 참여하며 사람들로 하여금 물리적 효과 기반의 게임 플레이의 한계를 이끌어 낼 수 있도록 장려하였다.



RoboBlitz 의 온라인 포럼을 제공하는 것은 그만한 가치가 있었다. 출시 후 우리는 특정 그래픽 칩을 사용하는 사람들이 게임을 즐기기 위해서 필요한 최신 드라이버를 구하는데 문제를 겪고 있다는 것을 알게 되었지만 회원 중 한 명이 재빨리 이러한 문제를 해결하기 위한 수단을 강구해냈다. 좀 더 헌신적인 유저들은 포럼을 관리하는 일에 자원하기도 하였다. 이처럼 모든 유저들이 서로를 도와가며 *RoboBlitz* 의 커뮤니티가 성공할 수 있었다.

무엇이 잘못되었는가

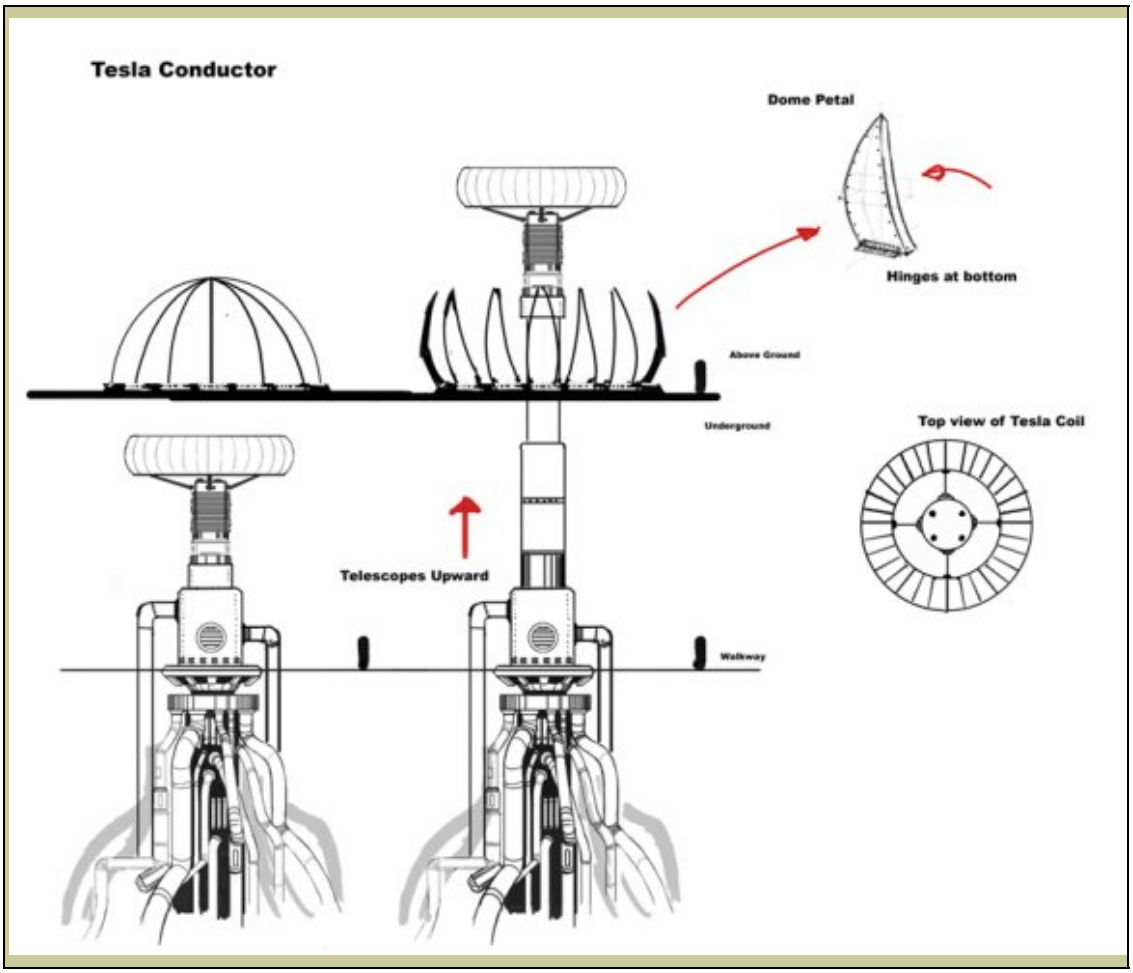


사전 제작의 부족

RoboBlitz 의 개발을 시작하며 우리는 사전 제작 단계를 거의 거치지 않고 곧바로 레벨 제작 단계로 뛰어들었다. 우리는 로봇이 어떤 기능을 지니고 있으며, 우리가 만들기를 원하는 것이 퍼즐과 액션으로 구성된 게임 플레이라는 것도 알고 있었다.

그래서 우리는 좋은 아이디어가 떠오르면 무엇이든 간에 그걸 넣고, 잘 작동하는 것들은 그대로 사용하는 방식으로 제작을 진행하였다. 이러한 방법은 게임 디자인에 있어 굉장히 빠르고 역동적인 변화를 가져왔는데, 사전 제작 단계에서 취했다면 아무런 문제가 없을 방식이었다. 그렇지만 우리는 6 개월에서 11 개월짜리 프로젝트에서 새로운 무기를 추가하는 등의 개발을 진행하고 있었기에 일은 잘 풀려나가지 않았다.

사전 제작의 부족이 가져온 가장 큰 문제는 레벨을 제작하기 전에 주요 게임 플레이 체계와 캐릭터 행동 양식을 확실하게 결정하지 않았다는 것이었다. 게임의 많은 부분이 제작된 후에 새로운 기능을 추가하거나 캐릭터의 행동 양식을 수정하는 것은 엄청나게 부담스러운 일로 다가왔다. 이 말인 즉, 우리가 전체 게임을 다시 돌아보며 새로운 기능이 기존의 게임플레이에 방해가 되지 않도록 레벨들을 고쳐야 했다는 것이다. *RoboBlitz* 와 같이 자유로운 이동과 게임플레이를 선보이는 게임에 있어서 그러한 작업은 결코 간단한 것이 아니었다.



우리는 플레이 체계, 조작, 카메라 각도, 레벨 특징 등에 크게 변화를 줘야 했지만 컨텐츠 완료 마일스톤까지 남은 시간은 몇 주되지 않았다. 이러한 새로운 변화들은 현지화된 텍스트들을 다시 번역하고, 음향 효과를 개선하거나 아트를 손보게 하였고 이를 테스트해봐야 하는 문제도 가져왔다. 결국 게임의 일부 지역과 요소는 우리가 원래 원했던 것보다 부족한 형태로 테스트를 거치게 되었다.

지나친 야망

우리의 개발 팀에는 훌륭한 사람들이 많았지만 상업 타이틀을 내놓기 위한 개발 과정에 임해본 사람은 몇 명 되지 않았고, 이것은 후에 우리에게 영향을 끼치게 되었다.

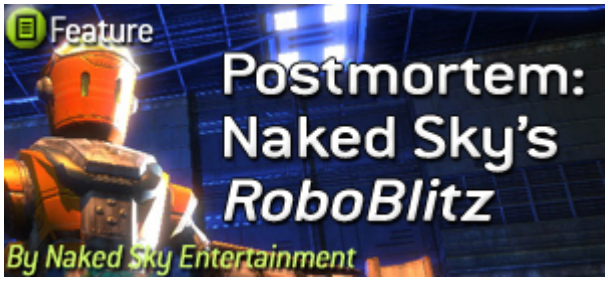
가장 초기 단계에서 우리는 아직 입증되지 않은 기술을 게임에 이용한다는 것이 우리에게 가져다 줄 영향을 정확히 예측하지 못하였다. 우리는 참신한 아이디어가 가득했기 때문에 이것들을

게임에 넣기로 결정했다. 그러나 우리에게 주어진 예산과 마감일 하에서 이러한 요소를 넣는다는 것은 제정신을 가진 경험 많은 스튜디오라면 받아들이길 꺼렸을 법한 것이었다.

이 정도 스케일의 게임에 필요한 시간과 예산을 과소평가하는 바람에 우리는 시작부터 일손이 부족하였다. 한 명의 사람이 여러 업무를 넘나들며 작업을 진행했으며 가끔 이것은 부정적인 결과로 이어지기도 하였다. 예를 들어 우리의 리드 프로그래머는 동시에 시스템 관리자이기도 하였기에 네트워크 혹은 버그 트랙커, 버전 컨트롤 소프트웨어 등에 문제가 발생하면 그는 문제를 해결하기 위해 프로그래밍을 멈춰야만 했다.

우리의 야심만만한 디자인과 부족한 일손은 팀 전체를 개발기간 내내 피로에 쌓여 살게 만들었다. 마감에 맞추기 위해서는 어쩔 수 없이 긴 시간을 일해야 했으며, 이것은 우리의 건강과 삶의 질에 경종을 울렸다. 그러나 다행스럽게도 작업을 함께한 모두는 젊고 의욕적이었으며, 이 프로젝트를 좋아했었기에 장기적으로 나쁜 결과가 나타나진 않았다. 그렇지만 우리가 앞으로는 작업 방식을 달리할 것임은 분명한 일이다.

새로운 기술이 지니는 문제점을 과소평가



완전히 물리적 효과에 기반을 둔 게임을 만드는 것은 어려운 일이다. 완전히 물리적 효과에 기반을 둔 *재미있는* 게임을 만드는 일은 OMGWTFBBQ 라는 말이 나오게 어렵다! 우리가 직면한 문제는 단지 사실성을 추구하는 것만이 재미있는 것이 아니라는 것이었다. “사실적”인 물리적 효과와 “재미있는” 물리적 효과 사이의 균형을 지속적으로 조절해줄 필요가 있었던 것이다.

Blitz 의 움직이는 방식이 좋은 예이다. Blitz 가 현재와 같이 부드럽게 움직이게 하기 위해서 프로그래밍을 하고 여러 수치를 조절하는데 몇 주가 걸렸다. 여전히 마리오 같은 움직임을 보이지는 않지만 게임의 모든 부분이 물리적 효과에 기반을 두고 있는 만큼 그건 어쩔 수가 없었다.

Blitz 의 집게 팔 역시 또 다른 문제점이었다. 우리는 유저 친화적이고 재미있는 조작을 위해 코드를 14 번이나 다시 작성했다. 발견법적인 A.I.를 Blitz 의 잡기 루틴에 넣는 것이 열쇠였다. Blitz 가 물건을 더 잘 잡을 수 있도록 몸을 자동으로 돌리는가 하면 물건이 너무 낮은 위치에 있다면 물건을 집기에 알맞은 각도만큼 몸을 숙이는 것이다. 플레이어가 모든 일을 직접 해줘야 한다면 게임의 재미가 떨어질 것이었기에 우리는 이러한 기능을 게임에 넣었던 것이다. 우리는 A.I.를 이용하여 물리적 효과 기반 애니메이션을 조작하게 하는 것이 절차적 애니메이션이 적용된 게임의 미래라고 생각한다. 그렇지만 우리는 이러한 것을 개발 후반에 들어서기까지 깨닫지 못했었고 많은 시간이 지나서야 깨달았다.

무엇보다도 아직 개발 중인 엔진을 이용하여 물리 시뮬레이션을 작성한다는 것은 엄청난 과제였다. 우리는 AGEIA 의 PhysX 툴 2.3 버전을 이용하여 개발을 시작하였는데 이 툴의 기능 중 다수가 우리가 원하는 방식으로 작동하지 않는다는 것을 깨달았다. 왜냐하면 우리 외에는 그 누구도 이 기능들을 게임에 사용하지 않았기 때문이었다.

AGEIA 는 우리의 요구에 대해 답해왔으며 이러한 기능들이 우리의 필요에 맞게 작동하도록 많은 노력을 기울였다. 그렇지만 새로운 엔진 빌드가 나올 때마다 우리는 지난 설정 값들을 전부 재조정해야 하였으며, 베타 버전에 다시 생겨난 버그들을 보며 작업을 해야 했다. 이러한 예측하지 못한 반복적인 작업들은 일정을 유지하는데 문제를 가져왔다.

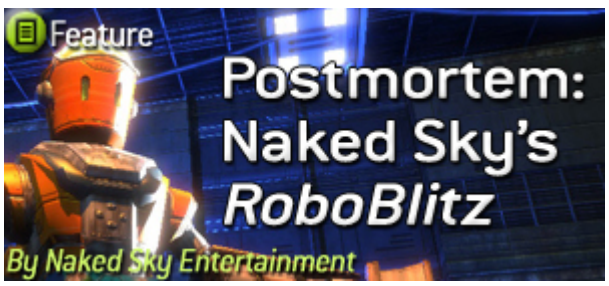
자금 부족

자금 유출, 자금 유출, 자금 부족! 이러한 일은 그 어떤 게임 개발자나 사업을 하는 사람에게 있어 좋지 않은 상황이다. 그렇지만 불운하게도 우리에게 이러한 일이 벌어지고 말았다. 우리의 게임은 위에서 언급된 기술적 문제와 모든 개발 과정으로 인하여 개발 기간이 배로 늘어나게 되었었다. 우리는 질 낮은 게임을 원하지 않았기에 만족할 수 있을 때까지 개발을 진행했었다.

이것은, 당연한 말이겠지만, 우리의 은행 잔고를 거덜나게 만들었고 재정에 압박을 가져왔다. 다행스럽게도 우리는 가족과 친구들로부터 어느 정도의 돈을 빌릴 수 있었다. 그렇지만 여러분은 결코 옛 애인에게 전화를 걸어 2 만 달러를 빌려달라고 하는 일을 겪고 싶지 않을 것이다. 우리처럼 철면피가 아니라면 말이다.

이러한 상황을 피하거나 최소화하기 위해서는 필요하다고 *생각되는 것*보다 더 많은 자본을 준비하라는 것이다. 이 말을 명심하라. 돈은 반드시 필요하고, 더 많이 필요하다. 우리는 *RoboBlitz* 에 예비로 사용할 자금(적어도 우리는 그런 의도로 준비했던 돈)까지 포함하여 수십만 달러를 투자하였지만 게임을 완성하기까지는 더 많은 돈이 필요했다.

부족한 내부 QA 진행



우리는 테스트를 전문 테스트 업체에 외주를 맡겼기 때문에 내부 QA 관리자를 고용하는데 있어서 큰 신경을 쓰지 않았었다. 결국 이것은 우리의 가장 큰 장애 중 하나가 되고 말았다. 외주 업체는 훌륭하게 일을 해냈지만 내부 QA 진행 없이는 자체적 테스트를 거쳐 외주

업체에 제안을 하는 것이 불가능했고, 만약 내부 QA 가 진행되었더라면 가능했을 수 있는 만큼 효과적이고 직접적인 테스트는 진행되지 못하였다.

프로젝트를 반쯤 끝내고서 우리는 내부 테스트 인턴을 고용하였고 QA 과정을 돕기 위한 아트 팀을 모집하기도 하였다. 그러나 곧 비슷한 문제에 부딪히고 말았다. 명확한 규칙이 없이는 그저 무작정 테스트를 진행하는 것에 지나지 않았기 때문이다. 이 때문에 우리는 PC 출시를 1 주일 앞두고서 문제를 찾고, 간단하지 않은 버그들을 수정하는데 시간을 소비했다. 이것은 만약 우리가 내부 QA 진행을 했더라면 결코 일어나지 않을 일이었다.

또한 우리는 엄격하고 자동화된 테스트 형식에 맞추어 개발을 진행해야 했었다. 자동화 테스트 없이는 모든 버그 수정이 또 다른 숨겨진 버그로 발전할 가능성이 높아지며, 문제가 나타나기 전까지는 그것을 인식하지도 못 하고 수정에는 많은 시간을 소비하게 되는 것이다. 이 교훈은 아무리 외주 업체가 훌륭하다고 하더라도 우리가 최소한 한 명의 숙련된 QA 인원 정도는 회사에 두고서 QA 를 전문적으로 담당하게 하여야 한다는 것이었다.

결론

열정, 재능, 훌륭한 파트너, 그리고 인내는 우리로 하여금 마침내 *RoboBlitz* 의 꿈을 이룰 수 있게 해주었다. 과거를 돌아보자면 우리는 이 프로젝트를 진행함에 있어 상당히 미숙했고 원래 생각했던 것에 비해 좀 더 많은 것을 배울 수 있었다. 그렇지만 이러한 것은 자신이 무엇을 모르는 지도 모르는 미숙함으로 인한 필연적인 결과였다.

이러한 좌절들에도 불구하고 우리가 잘못을 수정하려는 노력과 작업에 대한 헌신은 강렬했고, 마침내 팀 모두가 자랑스러워 하는 게임을 다함께 출시할 수 있게 되었다. 모든 사람이 우리가 이 게임을 만들며 즐거워했던 것만큼 *RoboBlitz* 를 즐겼으면 좋겠으며, 다음에 내놓을 물리적 효과에 기반을 둔 열광적인 멀티플레이 게임을 기대하시라!