미니-포스트모템 모음

(Mini-Postmortem Roundup)

작성자: 게임 디벨로퍼 매거진 담당자 (Game Developer Magazine Staff)

작성일: 2013년 4월 29일

이 글은 가마수트라 자매지인 게임 디벨로퍼 매거진(Game Developer magazine) April 2013[1]에 실렸던 것으로, 다양한 고급 인디 타이틀의 미니-포스트모템을 모았다.

지난해 게임 디벨로퍼(Game Developer)에서 배운 게 한 가지 있다면, 게임 개발 스튜디오는 항상 가능한 게임 플랫폼의 최신 정보에 귀를 열어 두어야 한다는 것이다. 그렇게 하지 않으면 큰 잠재 고객을 얻을 기회를 놓칠 수도 있다. 이러한 이유로 우리는 각기 다른 플랫폼에서 개발된 4 가지 짤막한 포스트모템을 모아 보기로 했다. 무테키의 <드래곤 판타지(*Dragon Fantasy*)> (모바일), 섭셋 게임즈의 <패스터 댄 라이트(*Faster Than Light*)> (PC), KIXEYE 의 <워 커맨더(*War Commander*)> (소셜), 그리고 RSBLSB 의 <다이애드(*Dyad*)> (콘솔)의 순이다. 당신이 싱글 플랫폼 개발자로서 실제로 아쉬움을 겪고 있든, 또는 단순히 작년의 몇몇 히트 게임의 잘된 점과 잘못된 점을 배우고자 하든 간에 이 미니-모템 모음이 도움이 될 것이다.

모바일: 드래곤 판타지*(Dragon Fantasy)*

애덤 리폰, 브라이언 소울러 (Adam Rippon and Bryan Sawler)

---

[1] 참조링크 : http://gdmag.com/issue/2013/April

우리는 2011 년 4 월 1 일, 돌아가신 애덤의 아버지 톰(Tom)께 바치는 작품으로 <드래곤 판타지>를 시작했다. 애덤은 이 게임 개발을 삶의 우울함과 스트레스에 대처하는 돌파구로 삼았다. 한 가지 프로젝트에 너무 집착하는 게 건전하다고 할 수는 없지만, 애덤은 분명히 짧은 시간에 놀라울 정도로 많은 일을 해 냈다. <드래곤 판타지>의 첫 편은 2011 년 8 월 23 일 iOS 에서 출시되었다.

잘된 점

1. 정기적인 컨텐츠 업데이트

게임은 어느 정도 성공이었고, 우리는 지속적으로 판매가 이어지기를 바라며 즉각 새로운 컨텐츠를 더해 나가기 시작했다.

무료 컨텐츠 업데이트라서 금전적인 성공으로 이어지지는 않았지만, 이 과정에서 얻은 호의적인 반응과 평가만 해도 큰 이익이었다고 생각한다. 우리는 인디 개발 커뮤니티에서 많은 사람들과 친해졌고, 그들은 큰 도움이 되었다. 우리는 쇼와 언론을 통해 게임을 마케팅하는 법을 배울 수 있었다. 또한 게임 개발 과정 중 수차례 소니와 접촉했는데, 우리의 컬트-중심 입지에 대한 헌신에 호의적인 반응을 보인 소니에서는 결국 <드래곤 판타지 북 II>를 그들의 펍 펀드(Pub Fund)에 포함하기로 결정하였다. 우리가 한 편만으로 만족하고 끝내 버렸다면 지금 이 글을 쓰지 못했을 것이다!

2. 언론 노출 성공

항상 좋은 리뷰가 올라와 있어야 한다는 점이 특히 중요하다. – 우리는 항상 좋은 리뷰를 받았다. “왜 이 게임은 공짜가 아니냐”고 불평하는 집단은 늘 있지만, 그 가운데서도 iOS 와 안드로이드 모두에서 별 4.5 개를 유지해 왔다. RPGamer 지 기사에서도 좋은 평을 받았고, 편집장 본인이 이 게임의 열렬한 팬이기도 하다. Joystig 지에서도 긍정적인 평을 받았다. 우리의 최고의 성취는 코타쿠 오스트레일리아(Kotaku Australia)와의 인터뷰를 꼽을 수 있다.—애덤은 이를 복사해서 벽에 걸어 두었고, 애덤의 어머니는 할머니에게 한 부를 보내기도 했다. (기사는 그만큼 좋았다.) 코타쿠 U.S.에서 코타쿠 오스트레일리아의 기사를 다시 싣는 것은 분명 흔치 않은 일이겠지만, 이 글은 미국판에도 실렸다. 이 기사가 나간 이후 판매가 늘어난 것은 물론이다. 너무 너무 멋졌다. 언론 노출 성공의 힘은 그만큼 대단한 것이었다.

3. 우수한 기술의 도움

<드래곤 판타지>는 막강한 픽셀을 자랑하는 최고급 엔진으로 보이지는 않을 수도 있다. 그러나 놀라운 사실은, 우리가 항상 자체 엔진과 도구를 사용했다는 점이다. <드래곤 판타지> 작업은 우리의 매우 강력하고도 사용하기 쉬운 UI 시스템의 프로덕션에 가장 적합하도록 이루어졌다.

게임 자체만으로 큰 돈을 벌지는 못했지만, 이 게임을 위해 개발한 기술들을 다른 도급 프로젝트에도 사용하여 상당한 수입을 얻었다. 우리의 MuTech 엔진을 사용하여 대형 클라이언트들의 수많은 프로젝트를 수행했으며, 정치 뉴스 앱에까지 사용해 봤다! 많은 블로그에 리뷰가 올라오는 와중에도 아무도 이것이 네이티브 아이폰 앱이 아니라는 것을 알아차리지 못했다. 우리가 매우 자랑스럽게 생각하는 부분이다. <드래곤 판타지>를 아무 규격 엔진이나 사용해서 제작할 수도 있었지만, 가능한 수단만 있다면 직접 크로스-플랫폼, 애플리케이션-애그노스틱(application-agnostic) 엔진을 개발하여 얻는 이익은 또 다른 의미가 있는 것이다.

잘못된 점

1. 모바일 런칭

우리는 업데이트할 때마다 모바일 8-비트 RPG 에 관심이 있는 유저들에게 더 많이 접근하기가 점점 어려워진다는 사실을 깨달았다. 그들이 존재하지 않는다는 말이 아니다. 그들은 분명 존재하지만, 모바일 게임에 있어 하드코어 취향을 가진 게이머들에게 정보를 전달하기가 매우 어렵다는 점이 문제였다. 또한 모바일 게임

사이트에 업데이트 정보를 올릴 수 있는 횟수에도 한계가 있다. 유료 업데이트를 사용했으면 금전적으로 좀 더 성공을 거둘 수 있었을지 모르지만, 더 큰 기반이 없으면 그 효과도 금방 사라질 것이다.

첫 번째 교훈: 3 달러짜리 모바일 게임의 고객을 찾는 것은 사람들 말처럼 어려운 일이다. 그 고객들이 일반 모바일 고객보다 하드코어라고 해도 결국 마찬가지이다.

또한 출시에도 문제가 있었다: <드래곤 판타지 1>에서 아직까지도 큰 문제가 되고 있는 것은, 모바일 게임이라는 '낙인'이다. 우리 게임이 저급 모바일 RPG 처럼 보이지는 않지만, 작은 모바일 팀에서 제작했기 때문에 모바일 외의 세계에서 주목을 받는 데 심각한 어려움을 겪었다. 스팀 제출은 몇 달 걸려 결국 거절당했고, 좀 더 인디에 호의적인 다른 스토어에서도 '모바일 포트(mobile port)에는 관심 없다'는 대답을 들어야 했다. (그날 나는 정말로 충격 받았다.) 두 번째 교훈: 코어 게임을 만들고 있다면 "모바일 포트"라고 불리기 전에 다른 플랫폼에서도 출시할 것.

2. 손쉬운 포팅(Porting) 때문에 소홀해진 테스트

우리는 <드래곤 판타지>를 매우 열정적으로 작업했고, 이 게임이 모바일에서 무명으로 남는 것을 원하지 않았기 때문에 바로 다른 플랫폼으로 포팅을 시작했다. 우리가 구축한 기술 덕분에 게임을 iOS 에서 맥(Mac)과 윈도우(Windows)로 빠르게 포팅할 수 있었다. 그러나 게임의 포팅이 너무 손쉬웠기 때문에 이 포트들의 상태에 대해서 과신하는 결과를 초래하고 말았다.

맥 샵(Mac Shop)을 전문으로 한다는 것은 맥 포트의 테스트도 철저하게 해서 안정성을 유지해야 한다는 뜻이다. 반면 PC 쪽에서는 우리의 테스트가 충분하지 못했고, 결과 출시 시점에 불만족스러운 결과물이 되고 말았다. 윈도우 게임에서는 다른 플랫폼에서는 필요 없는 많은 것들을 해야 했다. – 인스톨러(installer), 런타임(runtimes), DirectX 의 적합한 버전이 설치되어 있는지 점검 등등. 뿐만 아니라 윈도우 XP("어디 두어도 상관 없다")에서부터 윈도우 7("미안, 파일을 거기다 두면 안돼!")에 이르기까지 각기 다른 윈도우 버전 때문에 고생한 결과 우리는 세 번째 교훈을 얻게 되었다. 다른 데서 잘 되었다고 해서 테스트를 소홀히 하면 안 된다.

3. 너무 진품(Authentic)이라서 문제

처음 언론에 게임에 대해서 언급하기 시작했을 때, 우리는 모든 방면에서 진품으로만 만들었다는 것에 자랑스러워 했다. 엄격하게 오리지날 NES 에서 구현 가능한 색깔만을 사용했고, 아트 작업조차도 일정 수의 타일당 컬러(colors-per-tile)로만 제한했다. 문제는, 우리는 개발자로서 이를 존중했고 많은 언론에서도 이 점을 칭찬했지만, 대중은 동의하지 않았다는 점이다. 우리는 과거 8 비트 게임의 모습과 사람들이 8 비트 게임이라고 *기억하는* 모습에는 큰 차이가 있다는 것을 깨달았다. 우리의 일부 맵들은 우수한 NES 타이틀과 대비하여 견고하게 제작됐지만, 게임의 외모가 나쁘다는 불평을 계속 들어야 했다.

내부적으로 우리는 이 상황을 “이해를 못 하는” 사람들, 또는 1980 년대 중반 우리에게 영감을 준 게임을 해 본 적이 없는 사람들의 의견으로 치부해 버렸다. 이러한 피드백을 무시하기로 한 우리의 결정은 결과적으로 잠재적인 게이머들이 해 보기도 전에 등을 돌리게 만들었다. 우리는 대규모 PS3/PS Vita ‘재런칭’과 기존 플랫폼 업데이트를 준비하면서 그 때 받았던 피드백을 다시 돌아보고 사람들이 원하는 좋은 방향으로 모든 아트워크를 업데이트했다. 비전을 유지하는 것을 “인디스럽지 않다”고 말할 이유는 없다. 가장 큰 잠재 고객에게 다가갈 수 있는 방법이 있다면 그것이 중요한 것이다!

마지막 워리어 퀘스트

누군가 우리에게 3 년 전에 인디 게임을 개발할 만한 플랫폼이 뭐냐고 물었다면 나는 분명히 iOS 를 타겟으로 하라고 대답했을 것이다. 소규모 개발자들이 퍼블리셔 없이 저렴한 초기 자본으로 게임을 출시할 수 있는 기회가 되었던 이 시장이, 지금은 아이러니컬하게도 너무 경쟁이 치열해져서 여기서 주목받으려면 타이틀을 출시하기까지 큰 돈을 대 줄 수 있는 퍼블리셔를 얻는 것이 가장 좋은 방법이 되고 말았다. 어떤 플랫폼이든지 타겟 소비자를 찾는 것은 어려운 일이다. 그러나 아이튠 앱 스토어와 같은 새 출시작이 넘쳐나는 곳에서라면? 당신이 유명한 개발자이거나 아주 운이 좋지 않은 이상, 여기서 성공하기는 어려울 것이다.

따라서 <드래곤 판타지>의 플랫폼으로 iOS 를 포기한 것은 아니지만, 다음 게임을 기획할 때는 iOS 가 첫 번째 플랫폼은 아닐 가능성이 높다. PC 가 다시 인기를 얻어 돌아오고 있고, 콘솔도 인디 개발자들에게 점점 열리고 있다. (소니는 분명히 그렇다고 단언할 수 있고, 듣기로는 요즘 들어 닌텐도도 함께 일하기가 한결 수월해졌다고 한다.) 게임을 출시할 수 있는 곳은 많다. 당신의 게임이 빛날 수 있는 곳을 골라서, 그 곳으로 가라.

PC: <빛보다 빠르게(*Faster Than Light*)>

저스틴 마, 매튜 데이비스 (Justin Ma and Matthew Davis)

<빛보다 빠르게(FTL)>는 취미로 시작한 프로젝트가 큰 성공으로 이어진 사례이다. 여기서는 본 게임의 개발 과정의 잘된 점과 잘못된 점, 그리고 킥스타터(Kickstarter) 캠페인을 소개한다.

잘된 점

1. 독특하고 새로운 디자인

<FTL>은 매우 기본적인 컨셉만으로 시작되었다. 우리는 보통의 스페이스 게임이 우주선 조종 등에만 중점을 둘 뿐 플레이어 자신이 캡틴이 된 느낌을 주지는 못한다는 것을 알고 있었다. 우리는 각각의 승무원들의 행위를 관리하는 것은 물론 높은 수준의 비행 전략 의사 결정도 경험할 수 있는 게임을 원했다. 보다 흥미로운 게임 경험을 창조하기 위해 우리는 독특한 타입의 게임플레이를 고안해 냈다. 일부는 “시뮬레이션/전략”, 일부는 “본인이 원하는 대로 선택하는 어드벤처”인 혼합 RPG 장르이다.

많은 사람들이 기본적인 게임플레이의 힘(벤 프룬티의 놀라운 음악에 힘입어) 때문에 이 게임의 결점을 보지 못하고 간과하기도 한다. 단순한 그래픽, 반복적인 성격, 때때로 지나친 난이도 등. 이 게임들은 우리가 멋진 공상과학 영화나 텔레비전을 볼 때 느꼈던 판타지를 효과적으로 충족시켜 준다.

처음에 우리는 이렇게 경쟁이 심한 게임 분야에서는 시장 가능성이 없다고 생각했기 때문에, 많은 사람들이 우리 게임을 좋아한다는 사실에 놀랐다. 결국 플레이어가 자신이 원하는 대로 게임을 창조하고자 하는 욕망을 실현한 것이 호소력을 가진 것이라고 믿는다. 개발자들은 늘 이 감정에 대해 강조해 왔지만, 특히 우리의 <FTL>의 성공에는 이 욕망의 감정이 큰 역할을 한 것으로 생각된다.

2. 기가 막힌 타이밍(그리고 운)

2011 년 초반 우리는 둘 다 다니던 직장을 그만두고 1 년 동안 소규모 게임 프로토타입 제작에 몰두했다. <FTL> 개발을 시작한 이래 우리는 IGF China 2011 제출 마감에 맞추는 것을 목표로 견고한 마일스톤을 구축했다. 그 때까지 완성도 있는 게임 프로토타입을 얻지 못하면 다른 아이디어를 찾기로 했다.

이후 4 달간 우리는 게임보다는 게임의 매커닉 작업에 몰두했는데, 게임의 방향성을 잃고 좌절하던 중 막판 2 주를 남겨 두고 좋은 생각이 떠올랐다. 게임의 구조와 페이싱은 거의 하룻밤만에 결정했고, 플레이 가능한 우리의 첫 프로토타입을 IGF 에 제출할 수 있었다. 인디 게임 대회의 마감일은 정기적인 개발 마일스톤으로 멋지게 이어졌는데, 이는 우리의 운 좋은 타이밍의 시작에 불과했다.

2012 년 초반 우리는 자금이 바닥나서 크라우드펀딩(crowd funding)을 알아보기 시작했다. 이 때는 킥스타터(Kickstarter)가 게임 프로젝트에 수백만 달러를 모으기 전이었는데, 당시 우리는 이것이 일부 대중에게 접근하여 몇천 달러 정도를 모금할 수 있는 좋은 방법이라 생각했다. 우리는 2012 IGF 에서 호평을 받으며 퍼블리시티를 얻기 시작했는데, 그 덕분에 GDC 도중 OnLive 서비스에 간단한 퍼블릭 데모를 선보일 수 있었다. 따라서 우리는 이미 이 이벤트들과 함께 킥스타터를 준비하고 있었다. – 그러나 가장 운 좋은 순간은 더블 파인(Double Fine)에서 자신들의 킥스타터 캠페인을 런칭하면서 시작되었다. 우리가 준비되기 약 2 주 전이었다. 더블 파인 덕분에 우리의 킥스타터도 더욱 활발히 진행되었다.

그래서 FTL 킥스타터는 “더블 파인 효과”, IGF 에 실린 호평 두 건, OnLive 와 GDC 쇼 플로어에서 가능한 데모 덕분에 단 한 주만에 큰 효과를 볼 수 있었다. 이렇게 완벽하게 연속적으로 이어진 퍼블리시티가 킥스타터의 놀라운 성공의 원동력이 되었음은 분명하다.

3. 킥스타터(Kickstarter): 긍정적인 면

돈은 *모든 것*을 바꿀 수 있다. 킥스타터의 성공은 하룻밤만에 <FTL>을 단순한 취미 프로젝트에서 비즈니스로 바꾸어 놓았다. 우리는 우리 게임 출시를 손꼽아 기다리는 상당수의 팬을 포유한 “스튜디오”로 거듭났다. 이는 FTL 개발 중 가장 큰 성공의 순간이었던 동시에 모든 스트레스와 이슈들의 근원이 되기도 했다. (추후 자세히 설명)

우리의 자금은 개발 1 년만에 바닥나 버렸다. 우리가 상업적인 출시를 목표로 게임을 기획한다면 먹을 것, 렌트, 라이센스, 변호사 등 모든 비용을 준비해야 했다. 따라서 우리는 킥스타터 캠페인을 런칭하면서 $10,000 정도의 모금을 목표로 했다. 이 목표에 간신히 도달하거나 운이 좋으면 $15,000 에서 $18,000 까지도 모금할 수 있을 것으로 기대했다. 그러나 위에서 언급했던 운 좋은 타이밍 덕분에 우리는 $200,000 이 넘는 자금을 모을 수 있었다.

킥스타터는 우리의 개발 프로젝트에 여러가지 긍정적인 효과를 가져왔다: 무엇보다도 우리는 더 이상 렌트비를 걱정할 필요가 없었다. 또한 우리는 FTL 의 가능성을 확장할 수 있었다. 벤은 우리가 최소한으로 한정했던 초반의 뮤직 기획을 풀-사운드트랙으로 확장할 수 있었고, 작가 톰 주버트(Tom Jubert)에게 의뢰하여 <FTL>의 설화 세계를 확장하여 더 많은 우주선, 외계인, 무기들을 선보일 수 있게 되었다.

이 캠페인은 또한 게임 홍보의 기회가 되기도 했다. 우리의 캠페인이 킥스타터의 운동 확대에 기여하기도 했고, 우리 작품이 출시됐을 때 우리는 킥스타터의 성공의 “첫 주자”로 소개되었다. – 이 둘 다 언론의 관심을 끄는 동기가 된 것은 물론이다. 밸브(Valve)지에서까지 관심을 보여 우리는 게임을 배급하고 스팀에서 우리의 베타를 호스트할 수 있게 되었다. 이 프라이빗 베타(private beta)는 아마도 킥스타터의 가장 중요한 부분일 것이다. 우리는 3,000 명에 달하는 베타 테스터들을 모집했는데, 이는 새 인디 개발자로서는 놀라운 규모이다. 또한 이를 통해 FTL 은 더욱 안정적으로 다듬어진 게임으로 거듭날 수 있었다.

잘못된 점

1. 킥스타터: 부정적인 면

$200,000 을 모금하고 10,000 명에 가까운 팬을 보유하게 되면 대중의 기대도 변하게 마련이다. 많은 사람들이 자금 사정이 좋아진 만큼 원래 계획보다 게임 규모도 더 커지고 품질도 좋아지기를 기대했지만, 우리는 킥스타터가 끝난 시점에서 불과 다섯 달만에 출시를 계획하고 있었다. 프로젝트를 확장하기 위해 온갖 수단을 동원해야 했고 (추가 인력 투입, 더 좋은 기술의 라이센싱 등), 시간도 더 필요했기 때문에 일의 양과 주어진 시간 사이에서 균형을 찾기가 어려웠다. 이 부분에서는 그런 대로 성공했다고 생각한다. 게임은 크게 확장됐고, 예정보다 불과 2 주일 후에 출시할 수 있었다. 하지만 그 과정은 분명히 쉽지 않았다.

팬들이 늘어나고 홍보 효과가 커지면 홍보 관련 일도 그만큼 늘어난다. 우리는 PR 담당자나 마케팅 매니저가 따로 없었기 때문에 늘 가득찬 이메일에 일일이 답하는 것도 우리의 일이었다. 우리의 고마운 새 팬들을 위한 커뮤니티 공간 포럼을 기획하면서 전에 겪어보지 못한 많은 기술적인 문제가 생기기도 했다. 되돌아보면 PR 담당자를 고용하여 웹사이트를 담당하도록 하고 우리는 개발에만 전념하는 편이 더 효율적이었을 것 같다.

외주 계약을 하고 커뮤니티를 구축하면서 우리는 본격적인 회사의 모습을 갖추게 되었고, 변호사를 찾거나 디스트리뷰터와의 계약을 검토하는 등 점점 일이 많아졌다. 매일 매일이 전에 해 본 적 없는 일들을 배워 나가는 과정의 연속이었다. 우리는 전혀 준비가 되어 있지 않았기 때문에, 친한 친구나 가족들의 조언에 의존해야 했다. 게임 개발을 시작하면서도 비즈니스에 대해서 배울 것이 이렇게 많다는 것을 전에는 미처 몰랐던 것이다.

2. 너무 많은 행사/제한된 개발 시간

FTL 은 작은 실험 프로젝트로 시작되었기 때문에 원래의 비전은 매우 한정적었다. 10 여 가지의 기본적인 이벤트와 각 타입별 플레이버 텍스트(flavor text)만 계획하고 있었다. 본질적으로 전통적인 보드 게임에서의 이벤트 카드 세트와 비슷한 수준의 제작을 계획했던 것이다. 좀 더 다양성을 더해야 한다면 텍스트를 조금 더 넣으면 된다는 것이 우리의 (순진한) 계획이었다. 여기까지는 어려울 게 없는 일이다.

그러나 킥스타터 이후 우리는 게임 세계를 확장하여 다양한 외계인 종족과 더불어 더 많은 로케이션과 이벤트를 추가하기로 결정했다. 이 때까지 우리는 10,000 단어 정도의 이벤트만 갖고 있었다. 프로젝트가 끝날 무렵 FTL 은 20,000 단어를 갖게 되었다. 이것만 해도 방대한 텍스트이지만, 우리는 곧 이것도 부족하다는 것을 깨달았다. 섹터별로 이벤트를 나누게 되면 20,000 단어도 모자란다.

FTL 리뷰어들과 플레이어들이 제기한 몇 가지 공통된 이슈는, 이벤트가 반복된다는 점이었다. 작가는 출시 6 달 전부터 작업에 들어갔지만, 우리가 원하는 수준의 다양성을 창조하기에는 부족했다. 텍스트를 써서 게임에 넣는 것은 어렵지 않지만, 독특한 애니메이션과 아트 부분은 수백 회의 리플레이에 걸쳐 신선함을 줄 정도로 볼륨을 유지하는 것이 매우 어려운 일이다. 한 번 하고 나면 임팩트를 잃어버리는 수많은 독특한 이벤트들을 만드는 것보다는, 메인 이벤트를 좀 더 설득력 있게 만드는 데 시간을 투자했더라면 더 좋은 결과가 되었을 것이다.

3. 멀티-OS(Multi-OS) 런칭

크로스-플랫폼 개발은 보통 좋은 계획이다. 우리는 다른 OS 를 지원하여 유저 베이스를 가능한 한 많은 관심 있는 플레이어들까지 확장하고자 했다. 그러나 첫번째 게임 프로젝트를 3 개 플랫폼에서 동시에 출시하기로 계획한 것은 좋은 계획이 아니었다.

우리는 준비되었다고 생각했다. – FTL 은 첫날부터 크로스-플랫폼 게임으로 가기로 결정했기 때문에 모든 라이브러리와 코드도 (대부분) 쉽게 다른 시스템으로 옮겨질 수 있도록 했고, 18 개월의 개발 기간 중 5 개월이 지난 시점에는 이미 FTL 의 리눅스 개발 버전까지 마련되어 있었다. 우리는 어리석게도 윈도우의 빌드가 준비 완성되면 OSX 와 리눅스 빌드는 쉽게 마무리할 수 있을 것이라고 믿었었다. <FTL>의 '마무리'에 들어가는 작업을 과소평가했던 것이다.

막판 크로스-플랫폼 작업은 고사하고, 윈도우 버전을 끝내는 데만도 하루 12~15 시간 작업이 꼬박 들어갔다. 성공했다고 생각했지만 출시 4 일 전에 리눅스 빌드가 수많은 다른 리눅스 플레이버에서 돌아가지 않는다는 것을 발견했다. 이 문제를 해결하기 위해 수많은 밤샘작업을 했고 리눅스 포팅 전문가들에게 셀 수 없을 만큼 전화를 했다. 우리는 결국 문제 없이 런칭했지만, 그 과정은 매우 힘들었다.

출시 직후의 지원도 바로바로 해야 하는 상황에서 추가로 OS 를 지원하는 것도 큰 문제였다. 우리는 시스템 관련 문제를 해결하기 위해 한달 만에 두 가지 패치를 출시했고, 각각은 출시 전에 4 가지 시스템 (윈도우, OSX, 32 비트와 64 비트 리눅스)을 빌드하고 테스트하는 데 필요했다. 우리가 개발자로서 경험이 부족했기 때문에 파이프라인의 빌드는 최적으로 이루어지지 않았고, 단 두 사람이 4 가지 빌드의 독특한 현상들을 테스트하는 작업을 신속하게 하는 것은 불가능한 일이었다!

단순히 윈도우 출시 후에 리눅스/OSX 출시를 2~4 주 정도만 미루었어도 작업이 훨씬 용이했을 것이다. 이제는 게임 제작과 파이프라인 빌드에 대한 경험이 생겼기 때문에 앞으로는 이 정도로 힘들지는 않을 것 같다. 그러나 아무것도 모르던 시절에 처음부터 좌충우돌 했던 것은 좋은 선택이 아니었다.

소셜: <워 커맨더 (*War Commander*)>

데이빗 스콧 (David Scott)

나는 항상 리얼-타임 전략(RTS) 게임의 열렬한 팬이었다. RTS 에 대한 나의 열정은 처음에는 <듄 2(*Dune II*)>로 시작해서 <토탈 어나힐레이션(*Total Annihilation*)>으로 옮겨갔고, <워크래프트 3(*Warcraft III*)>와 <커멘드 앤 컨커:제너럴(*Command & Conquer: Generals)*>에서는 정점에 달했다. 나는 이 게임들을 죽을 때까지 할 기세로 몰두했고, 게임을 통해 좋은 친구들도 만났다. – KIXEYE 공동 창설자인 폴 프리스(Paul Preece)도 이 친구들 중 한 명이다.

폴과 나는 지난 2007 년 <워 크래프트>를 플레이하면서 mods 에 영감을 받아서, 몇몇 플래시 형태의 타워 디펜스(Tower Defense) 게임을 개발하기에 이르렀다. 이 게임의 성공으로 우리는 둘 다 직장을 그만두고 풀-타임 게임 개발자가 되었다. 3 년이 빠르게 흘러 우리는 페이스북을 위한 데스크탑 TD 버전(*Desktop Defender*)을 작업하고 있었고, 나는 여유 시간에 개발할 만한 브라우저 베이스의 RTS 게임을 찾고 있었다.

내가 찾아낸 게임은 간단한 HTML 형식이었다. 빌딩을 배치시키는 슬롯이 있고, (빌딩의 위치는 상관 없다) 즉각 공격이 시작되었다. 플레이어가 볼 수 있는 것은 잃은 군대와 약탈한 자원을 보여 주는 배틀 리포트뿐이다. RTS 에서 처음부터 끝까지 에픽 배틀을 즐기는 사람이라면 이 게임에는 크게 실망할 것이다.

우리는 타워 디펜스와 RTS 를 한 게임으로 접목시키고 싶었다. 빌딩과 방어지의 위치가 중요한 게임, 더 중요한 것은 플레이어가 리얼 타임으로 공격 상황을 보고 참여할 수 있는 게임 말이다.

2010 년 초에 나는 <백야드 몬스터(*Backyard Monsters*)> 작업을 시작했다. 이는 핵심 RTS 매커닉 주변에 친근하고 장난스러운 아트 디렉션(예를 들면 백야드에는 탱크나 총기 대신 몬스터 빌딩 베이스가 있음)이 있는 MMORTS 게임이다. <백야드 몬스터>는 출시후 대대적인 호평을 받았고, 플레이어 베이스와 함께 우리의 자신감 또한 커졌다. 우리는 그래픽을 보다 현실적이고 엣지있게 업데이트하기로 했다.

<백야드 몬스터>의 성공에 힘입어, 곧 <워 커맨더> 제작에 들어갔다. MMORTS 게임의 모든 것을 보여 주는 게임을 만들고자 했다. 탱크, 총기, 비행기, 헬리콥터, 자살 폭탄 테러단, 모든 것을! 12 달 전만 해도 시작할 엄두도 안 났던 이 게임이 지금은 페이스북의 최고로 인기 있는 MMORT 게임이 되었고, 평균 플레이 시간은 주당 8.6 시간을 기록하고 있다.

그러나 “탱크와 총기가 많은 <백야드 몬스터>를 완전히 리메이크하는 것”과 단 4 명의 팀이 6 달만에 라이브 게임을 만들어 내는 것은 전혀 다른 얘기다. 몇 가지 부분은 생략하고 런칭 후에 추가하기로 할 수밖에 없었다. 첫 세달 만에 1 백만 건이나 설치가 될 줄은 예상하지 못했고, 플레이어들이 게임을 하면서 이렇게 빠르게 향상될 지도 예상하지 못했다. 미처 준비가 안 되어 있던 우리는 출시 후 첫 몇 달간 재빨리 새 컨텐츠를 추가하는 일에만 몰두해야 했다.

잘된 점

1. 소수정예 팀 구성

우리는 <워 커맨더>를 3 명의 인원으로 6 달만에 해 냈다. 매우 효율적이었다고 할 수 있다! 1 년에 걸쳐 인원은 약 2 배로 늘었지만, 현재도 다른 게임 팀들에 비하면 소수 멤버를 유지하고 있다. 모든 게임은 한두 명이 디자인을 시작해서 메커닉을 점검하기 위한 프로토타입을 만들고 나서야 서서히 기술 담당 멤버들이 조인하여 출시 준비를 한다. 팀이 작으면 커뮤니케이션상의 부하도 최소한으로 낮출 수 있고, 결과적으로 초반에 매우 빠르게 진행할 수 있다.

KIXEYE 방식의 또 다른 장점은 완전히 스튜디오 모델을 따른다는 점이다. 각 팀이 완전히 독립되어 있고, 마치 각자가 작은 회사인 것처럼 자신들의 간부 프로듀서를 CEO 처럼 따르며 작업한다. 한 팀에서 쓰는 방식이 다른 팀에서는 적용되지 않을 수도 있기 때문에, 우리는 팀간에 특정 방법론을 강요하지 않는다. 물론 배운 점들을 공유할 때는 있지만, 팀간 상대방과 점심식사를 같이하며 수다를 떤다든가, 지난 결과를 정리하고 추후 작업을 논하기 위한 주간 프리젠테이션을 하는 등의 유기적인 형태로 유지하고 있다.

2. 트랙 만들기!

‘워 커맨더’가 완성된 지 2 년이 되었지만 현재도 매주 새로운 컨텐츠와 요소들이 업데이트 되고 있다. 이제 게임 개발은 5 단계 트랙(track)으로 관리하고 있다. 첫 트랙은 개발에 한달쯤 걸리는 큰 요소들이고(월드 맵이나 맞춤 가능한 유닛 등), 두번째 트랙은 보다 작은 요소들(부서별로 1~2 주 정도 걸리는 여타 작업들), 그리고 3,4,5 트랙은 인프라스트럭쳐를 개선하고 버그를 수정하고 오퍼레이션의 효율성을 증진하는 과정이다.

5 가지 트랙 모두 서로 독립적으로 움직이기 때문에 한 트랙이 늦어져도 다른 트랙의 진행에 영향을 주지 않는다. 우리는 개발자들이 신선한 감각을 유지하게 하기 위해 트랙간에 이동하면서 작업하도록 하고 있다.--때때로 모든 팀원들이 플레이어에게 당면한 문제를 함께 해결해 나가는 것은 좋은 일이다.

3. 수직 작업

월드 맵을 제외하고는(뒤쪽에서 다시 설명), 우리는 게임을 제 시간에 출시하면서도 좋은 품질을 유지하기 위해 올바른 의사결정을 했다. 에어 유닛을 더하기 위해 건물 몇 개는 희생해야 했고, 무기를 다양하게 하기 위해서 보병들의 수는 줄였고, 플레이어가 명령을 내릴 때 유닛에서 말하도록 하기 위해 음악도 줄였다. 모두 올바른 결정이었다.

수직적인 작업이 성공의 열쇠였다. 이 방법으로 하면 작업자가 한 지역에만 정체되지 않고 게임의 모든 부분을 조금씩 개발하게 된다. 나는 인디 게임 개발자로서 게임의 작고 사소한 부분을 파고들며 반복하다가 중심을 잃어버리기 쉽다는 것을 잘 알고 있다. 이 글에서 꼭 새겨두어야 할 부분이다. 지금 바로 당신의 게임 개발을 수직 작업으로 바꾸고, 어떤 한 요소도 게임 처음부터 끝까지 해 보면서 개발하도록 하자.

잘못된 점

1. 메타게임으로서의 부족함

게임 출시 전에 우리가 소홀히 했던 부분 중 하나가 플레이 공간인 월드 맵이다. <워 커맨더> 초반에는 타겟 리스트가 대신 제시되었다 (AI 와 플레이어 컨트롤 베이스). 플레이어는 리스트의 누구라도 스카우트하거나 공격할 수 있고, 죽이고 나면 리스트에서 제거되면서 더 높은 레벨의 타겟으로 대치된다. 이 시스템은 잘 돌아갔지만, 그 자체가 게임이라고 할 수 있는 리얼 월드 맵에 비하면 초라한 대안품이었다. 우리는 런칭 8 개월 후인 2012 년 4 월이 돼서야 맵을 추가할 수 있었다. 이를 출시하자마자 참여(engagement), 점유(retention), 금전적 교환(monetization) 등 활동이 25%나 급증하였다. – 그리고 우리는 월드 맵 메타-게임 레이어가 얼마나 중요한지를 까달았다. (또한 좀 더 빨리 출시하지 못한 우리가 얼마나 바보같았는지도 말이다.)

2. 광범위하지 못하고, 신속하지 못했던 점

우리는 너무 오랫동안 최근 게임 플레이어들의 니즈를 만족시키기 위해 게임에 빌딩과 유닛을 더하는 데만 의존해 왔다. 그 대신 이들이 플레이할 수 있는 샌드박스를 창조할 수 있는 요소들을 더했으면 더 좋았을 것이다. 우리는 플레이어들에게 빌드할 것들을 던져 주는 대신에 기존의 컨텐츠를 새로운 방식으로 접근하며 각각의 플레이어나 그들의 군대 베이스가 독특한 게임을 만들어 가도록

해야 했다. 이는 “내 게임에는 모든 것이 최상 레벨로 구축되어 있다”고 말하는 대신 “최상의 레벨로 모든 것이 갖추어져 있지만, 이것이 가장 좋은 조합은 아닌지도 모른다. 실험이 필요하다!”는 것을 깨닫는 과정이다. 이러한 태도로 접근하면 베이스 디자인과 공격 전략에 있어 공격쪽과 수비쪽 모두에 다양성을 창조할 수 있을 것이다.

3. 시간 예상 오류

우리는 견고한 RTS 게임을 갖고 있었다. 베이스, 군대, 지배할 세계 모두 갖춰져 있었지만, 한 가지 없는 게 있었다. 리얼-타임 배틀이 그것이다. 많은 기술적인 문제 때문에 우리는 온라인상에서 두 플레이어간에 공격할 수 있게 할 수가 없었다. 우리는 2012 년 10 월에야 라이브 배틀을 출시하면서 이 문제를 해결했고, 이제는 동시에 온라인에 있는 플레이어들끼리 같은 배틀에 참가하고 리얼타임으로 상호작용할 수 있게 되었다. 이는 당장 금전적으로 큰 수익을 가져올 만한 변화는 아니었지만 어쨌든 우리는 추진했다. 그렇게 하는 것이 더 멋졌고, 실시간 PyP 배틀이 되지 않는 게임을 우리 이름을 걸고 RTS 게임으로 내놓고 싶지 않았기 때문이다.

우리는 새로운 요소를 계속 개발 출시하면서 동시에 멀티플레이어를 라이브 게임으로 개장하는 것이 쉽지 않을 것은 알고 있었다. 그러나 이 과정이 실제로 얼마나 시간이 걸릴지는 전혀 몰랐다. 클라이언트와 서버 모두에서 작동할 수 있게 하기 위해서 게임에서 방대한 양을 완전히 다른 언어로 다시 작성해야 했다 (기존 팀 멤버들 아무도 익숙하지 않은 언어였다). 길고 힘든 6 개월 정도 기간에 걸쳐 우리는 개발하고, 테스트하고, 출시했다. 출시 전에 이 작업을 했더라면 몇 달간의 작업을 단축할 수도 있었을 것이고, 바로 프로덕션에 돌입하지 않고 잠시 앉아서 업무별로 작업을 추정해 봤더라면 시간 계획을 더 잘 세울 수도 있었을 것이다. 이 모든 것에 한 가지 긍정적인 면이 있다면, 라이브 배틀 출시 이후 팀원들에게 새로운 프로세스가 생겨 불필요한 작업을 최소화하고 매우 정확하게 마감에 맞출 수 있게 되었다는 것이다.

콘솔: <다이애드(*Dyad*)>

숀 맥그라스 (Shawn McGrath)

<다이애드(*Dyad*)>의 개발은 길고 어려운 과정이었다.

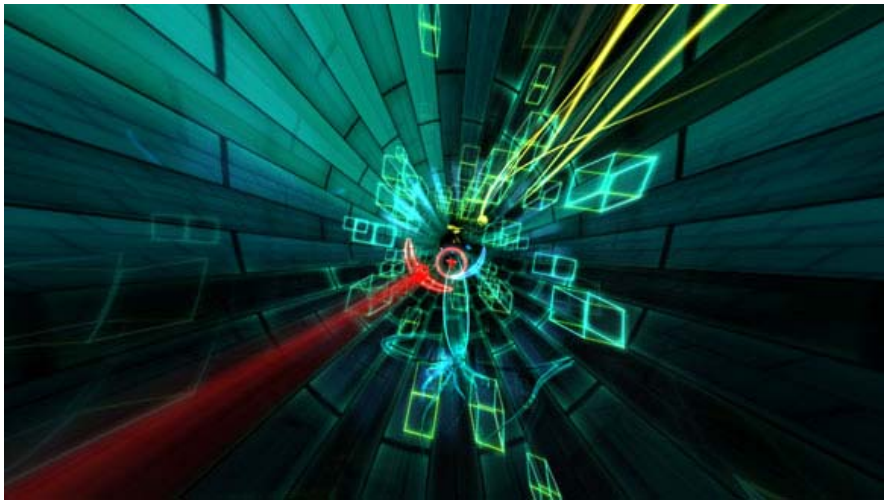
켄타 조(Kenta Cho) 게임을 많이 했던 것이 시발점이었다. <*rRootage*>와<*Parsec 47*>를 좋아했지만, <*Torus Trooper*>는 나하고는 좀 안 맞았다. 그래서 페코 코스키넨 (Pekko Koskinen)과 나는 <*Torus Trooper*>와 여러 다른 레이싱 게임을 분석해 보았고, 결과 대부분의 레이싱 게임에서 공통된 디자인 이슈를 발견했다. 우리는 이러한 이슈를 해결한 새 게임을 만들기로 했다. 게임이 빨라지면서 반사작용과 근육 메모리에만 의존하는 게임이 아닌, 전술적으로 생각하게 만드는 게임을 만들고 싶었다. 그러나 필요 이상으로 조종을 복잡하게 하고 싶지는 않았다.

우리는 이 문제를 해결하고 1 년만에 <다이애드>를 만들어 낼 수 있을 것이라 생각했다. 우리 생각은 틀렸다. 페코는 결국 프로젝트를 떠났고, 나는 혼자 계속 도저히 풀릴 것 같지 않은 문제들과 씨름하며 3 년을 더 보냈다. 3 년 내내 나는 하루에 10-16 시간씩, 일주일에 6-7 일씩 일했다. 그러나 결과는 그만큼 가치가 있었다.

잘된 점

1. 내가 너무 까다로워서

<다이애드>를 만들기 전에 나는 미야모토(Miyamoto)에 대한 이야기를 들었다. 그는 몇 달에 걸쳐 한 작업도 마음에 안 들면 폐기해 버리기로 유명하다. 이것이 쉬울 것이라 생각했지만, 실제로는 결코 쉽지 않은 일이었다. 나는 <다이애드>에 바친 한 작업을 90~95% 완성 단계에서 접어 버렸다. 9 달에 걸쳐 집라인(zip line)을 대체할 “게이트(gates)”라는 메카닉 작업을 했다. 게이트는 게임플레이에 있어 다른 모든 게임을 합친 것보다도 다양성과 재미를 더해 주었지만, 이 메카닉은 도저히 일반인이 이해하기 어려운 것이었다. 심지어 내 아내에게도 1:1 로 직접 몇 시간에 걸쳐 가르쳐 주려 했지만 (아내는 <다이애드> 게임을 매우 잘 한다) 끝내 이해를 못 했다. 그래서 나는 이 계획을 철회했다. 나는 그간 200 개가 넘는 다른 레벨을 창조했지만, 가장 좋은 것 하나만 남겨두고 모두 폐기해 버렸다. 결국 게임이 생각보다 좀 길어졌다. 하지만 이제 더 이상 삭제할 부분은 없다.



2. 혼자서 (거의) 모든 것을 다 하기

2010 년에 나는 이 게임을 몇몇 퍼블리셔들에게 보여줬는데, 이 중 세 군데서 크게 관심을 보였다. 우리는 펀드와 인력 증강에 대해 논의했다. 그러나 나는 이 모든 것을 취소하고, 나 자신의 예금을 털어 가능한 돈을 안 쓰고 생활하며 혼자서 작업하기로 했다. 그리하여 나는 내게 필요한 시간 동안 내가 바라는 방향으로 게임을 만들 수 있었다.

프로그래밍, 게임 디자인, 특히 그래픽은 PS3 버전 부분과 개발 첫 1 년간 파트타임 디자이너의 도움을 받았던 것 외에는 거의 모두 혼자서 했다. 나 자신의 엔진을 프로그래밍하면서 로드 타임을 1 초 이하로 줄일 수 있었고, 로우-레벨(low-level) 그래픽 효과와 광범위하고 다양한 새로운 그래픽 테크닉을 시도해 볼 수도 있었다. 포토샵용 모니터 하나, 코드용 하나, 게임용 하나, 각각 모니터를 따로 두고 그래픽과 코드를 바꿀 때마다 그 변화를 즉각 볼 수 있게 했다. 이렇게 하지 않았더라면 현재 <다이애드>의 비주얼 수준을 달성할 수 없었을 것이다. 그리고 라이브 코드 업데이트를 하지 않았다면 이렇게 많은 게임 디자인 아이디어를 테스트하는 것은 불가능했을 것이다.

프로모션과 광고도 내가 직접 진행했다. 프로모션 첫 작품으로 나는 플레이어의 인-게임 액션에 맞게 기울어지고 돌아가는 거대한 모터 달린 의자(“THE MACHINE”)를 만들었다. 나는 이 의자를 분해해서 내 쉐보레 임팔라에 싣고 2011 PAX East 로 달려갔다. 많은 관객에게 THE MACHINE 으로 <다이애드> 게임을 첫선을 보이기 위해 제이슨 데그룻(Jason DeGroot)과 나는 꼬박 이틀에 걸쳐 다시 재조립했다. PAX East 에서 나는 언론의 호의적인 관심을 많이 받았고, 이 때

마케팅도 스스로 하기로 결심했다. 시간이 많이 걸린다 해도 꼭 내가 직접 하고 싶었다.

3. 데이빗 캐나가(David Kanaga)와의 음악 협업

데이빗 캐나가는 <다이애드>의 반응성(reactive) 음악 시스템에서 가능한 모든 것을 이루었다. 우리는 첫 트랙에만 4 달이 걸려 작업했다. 그는 오클랜드에 있었고, 나는 토론토에 있었다. 그는 레벨별로 코드 변화와 인터렉션 이벤트 사운드가 완성된 음악을 나에게 보내 주었고, 이를 어떻게 게임에 넣으면 좋을지, 각기 다른 게임 요소별로 어떻게 믹스에 변화를 주면 좋을지 설명도 해 주었다. 나는 나 자신이 그 시스템에서 게임을 플레이하는 모습을 비디오에 담아 그에게 보내 주곤 했다. 약 4 달간 이렇게 주거니 받거니 하다가 마침내 데이빗이 토론토로 와서 첫 15 여 개 레벨을 함께 작업하게 되었다. 그는 이번에는 게임 플레이가 가능한 컴퓨터를 가지고 다시 오클랜드로 돌아갔다가 마지막 12 레벨의 스케치를 들고 마무리 작업을 위해 다시 토론토로 왔다.

대부분의 사람들은 내가 음악은 마지막에 추가된 것이라고 말하면 믿을 수 없다는 반응이다. 게임은 정말로 음악 없이 제작되었고, 데이빗이 각 레벨별로 게임 규칙에 따라 독특한 음악을 창조해 냈던 것이다. 그는 총 27 개의 독특한 멜로디를 만들어 냈다!

잘못된 점

1. 스피드 때문에 복잡해진 게임

<다이애드>는 내가 아는 게임 중에서도 매우 빠르고 복잡한 게임이다. 플레이어들은 본인의 아바타의 로케이션, 면역(immune), 콤보(combo), 극성(polarity)의 상태는 물론 각각의 적들의 타입, 상태, 로케이션도 계속 추적해야 한다. 수백만에 이르는 디자인 요소들의 도움 없이는 거의 불가능한 일이다. 이 요소 중 하나라도 불완전하면 게임은 완전히 무너진다. 이 요소 몇 가지를 소개한다.

물체가 빨리 움직일 때는 2D 스크린으로 깊이감을 표현하기가 특히 어렵다. 터널(tunnel) 안에 깊이를 나타낼 수 있는 두 가지 주요 요소가 있다. 적들은

튜브에 희미한 하이라이트를 남기고, 대부분 적들은 또한 그들의 앞쪽에도 끌린 자국을 남긴다.

플레이어의 “우주 오징어(space squid)” 아바타는 완전히 기능적으로 움직이도록 비주얼적으로 디자인되어 있다. 밝은 메인 칼라와 중심의 흑백 서클은 일부러 히트박스(hitbox)를 모호하게 나타내도록 했다. 그 물리적인 크기는 끊임없이 줄었다가 늘었다가 한다. 플레이어는 그레이징(grazing) 할 때 다른 적들과 달리 이 그레이즈 서클 중심에 있는 적들과 대비하면 10 배나 작아진다. 또한 랜싱(lancing)할 때 플레이어는 100 배나 커진다. 또한 플레이어는 극성에 따라 두 가지 다른 사이즈로 변할 수 있다.

이 모든 정보는 숨겨져 있다. 플레이어의 센터는 포지션의 막연한 지표일 뿐이다. 플레이어의 뒤에 남는 자국은 플레이어가 어디에 있는지, 어디로 가고 있는지를 쉽게 볼 수 있게 하기 위해 측면 움직임을 과장해서 나타낸다. <다이애드>는 플레이어가 자신의 아바타를 보기 힘들 정도로 너무 빠른 게임이기 때문에, 나는 플레이어의 포지션과 주변의 모션 변화만 인지하기 쉽도록 트레일(자국)을 디자인하기로 했다.

튜브 디자인도 정보 프로세싱을 극대화하기 위한 또 하나의 중요한 부분이었다. 튜브는 게임 내 모든 목표물의 참조 포인트가 되며, 예쁘면서도 부드러운 공간으로 느껴져야 했다. 대부분 레벨에서는 거리와 적들의 패턴, 또 플레이어가 준비한 것들을 알아보기 쉽게 하기 위해 그리드(grid) 패턴을 사용한다.

<다이애드>에서는 대부분의 레벨에 더블 튜브가 있어서 초고속 게임에 맞게 측면 방향 스피드를 더 잘 감지할 수 있다. – 플레이어 대 적들의 속도에 조심스럽게 집중해 보면 게임이 실제로 느껴지는 것에 비해 전혀 빠르지 않다는 것을 알게 될 것이다. 바깥쪽 튜브는 플레이어가 있는 곳의 바닥에서 튜브 안쪽과 바깥쪽이 라인업(line-up)이 되도록 오프셋(offset)된다. 이는 비주얼 노이즈를 증가시키고 플레이어의 정면으로 시선을 집중시킨다. 현재 상황을 쉽게 볼 수 있도록 플레이어의 전면 지역을 “화이트-아웃(white out)”시키기 위해서 블렌딩이 사용되었다.

이 외에도 정보 프로세싱을 가능한 한 효율적으로 하기 위한 많은 비주얼 디자인 테크닉이 사용될 수 있다. 나는 가능한 한 제한 없이 자유롭게 비주얼 디자인 스페이스를 활용하고 싶었다.

2. 배우기 힘든 메카닉

<다이애드>의 메카닉이 짜증날 정도로 복잡하다는 사실을 얼마 후에야 깨달았다. 나는 이 게임을 처음으로 스콧 필그림(Scott Pilgrim) 런칭 이벤트에서 선보이면서 모두가 곧 배워서 플레이할 수 있기를 기대했다. 못했다. 그래서 이번에는 적당한 튜토리얼을 붙여서 온타리오 아트 앤드 디자인 대학(Ontario College of Art and Design) 행사에서 다시 보여줬다. 아직도 플레이 불가능했다. <다이애드>는 1 년에 걸친 계속된 플레이테스트 후에야 누구나 도움 없이 플레이할 수 있는 게임이 되었다. 이후에도 적절한 러닝 커브(learning curve)가 될 때까지는 1 년이 더 걸렸다.

게임 전체가 다 튜토리얼이다. 메카닉을 작은 조각으로 나누어 각각 설명하며 가르쳐야 했다. 메카닉 하나만으로는 지루하기 때문에 언제 어디서나 플레이어가 활용할 수 있는 다양한 메카닉의 서브세트(subsets)와 함께 여러가지 독특한 목표와 모드를 고안해서 게임의 신선함과 흥미를 유지하도록 해야 했다. 이렇게 해서 게임은 백만배나 더 좋아졌지만, 내 생각에는 간단한 개념인데 사람들이 전혀 이해를 못 한다는 사실은 매우 속상한 일이었다. 2011 년 초반 내내 게임을 설명할 수 없다는 사실 때문에 나는 신경과민 상태였고, 게임의 구조 전체를 개조하고 나서야 안정을 찾게 되었다.

3. <다이애드>가 무엇인지 커뮤니케이션하기

<다이애드>가 무엇인지 설명할 수 없다는 사실 때문에 매출에도 타격이 되었다. 나는 <다이애드>가 다른 매체의 기준을 따르지 않는, 게임 경험으로서 가능한 한 “순수한” 것이기를 바랐다. <다이애드>가 두뇌에 작동하는 과정, 실제로 플레이하지 않고는 플레이가 불가능한 것들 말이다. 나는 베르토프(Vertov)의 영화 ‘카메라를 든 사나이(The Man with the Movie Camera)’에서 많은 영감을 받았다. 이 영화는 독특한 영화로서의 특성만을 보여준 사례이다. 나는 <다이애드>에서 이 영화와 비슷한 시도를 했기 때문에 게임에 대해서 논하는 것이 힘들어졌다. (따라서 대부분 사람들의 흥미를 끌지 못했다). 이 게임의 특성을 포기하지 않고도 게임에 대해 말할 수 있는 방법을 생각해 냈더라면 좋았을 것이다.

©2013, UBM LLC. Initially published in *Gamasutra.com*, a UBM LLC publication. Translated with permission.