KGDC 2002

게임 제작 공정 관리를 위한 자동화 도구 개발

-ODT(Open Design Tool)을 중심으로 소개-

작성자 : 신 원 동

㈜ 현무엔터테인먼트 이사

wdshin@hotmail.com

1. 게임 개발 각 팀별 업무 협력 및 의사소통을 위한 표준화 방법

의사소통 문제는 어느 프로젝트이건 있다. 처음의 기획의도대로 방향성을 잃지 않고 게임을 성공적으로 개발하기 위해서는 그 의도가 희석되지 않도록 의사가 잘 전달할 수 있는 팀원들의 의지, 방법, 회의를 통해서 의견의 충돌발생 후 조정시 회의의 방법도 매우 중요하다. 다음은 로지컬 씽킹(Logical Communication Skill Traning/ 저자 테루야 하나코, 오카다 케이코)이라는 책에서 메시지에 대해서 언급한 부분을 발췌하였다.

1. 메시지 전달

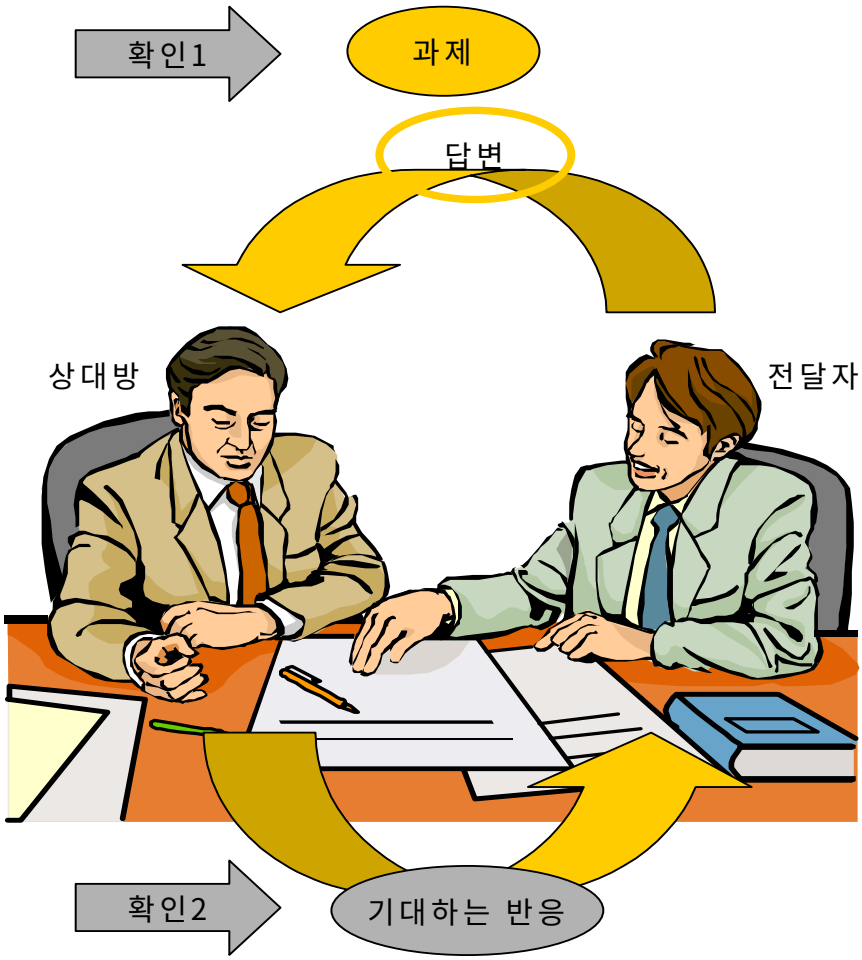
1) 상대방에게 전달해야 하는 메시지란?

'제가 말씀드리고 싶은 것은' 이라는 말로 이야기를 시작하는 사람이 많다. 그러나 '제가 말씀 드리고 싶은 것'이 아니라 '내가 지금 답변해야 하는 과제(테마)에 관해 상대방에게 전달해야 하는 '메시지'가 중요하다.

그러면 메시지란 무엇인가? 메시지란 다음의 세 가지 요건을 만족하고 있는 것이다. 우선 대화(Communication)에서 답변해야 할 과제(테마)가 명쾌히 나와 있어야 한다. 둘째, 과제에 대해 필요한 요소를 충족시킨 답이 있어야 한다. 그리고 세째, 대화를 한 후 상대방으로부터 어떤 반응을 얻고 싶은 것인지, 즉 상대방에게 시사하고자 하는 반응이 명확해야할 것이다.

'과제', '답변', '상대방에게 기대하는 반응'의 세가지 세트(Set)가 메시지(Message)인 것이다. '제가 말씀드리고 싶은 것'은 세가지 세트의 '답변' 부분에 지나지 않는다. 이것은 뒤집어 말하면 자신이 어떤 문서를 입수했을 때, 또는 다른 사람의 이야기를 들었을 경우 과제가 이것이고, 그것에 대한 상대방의 답변은 이것이고, 자신에게 이것을 하기를 바라고 있구나 하는 것이 자신의 머릿속에 명확히 남아 있는지 아닌지가 분명히 되었을 때 비로소 메시지라고 말할 수 있다.

'자신밖에 모르는 병'이나 '얼치기 독심술사 증후군'에 빠지지 않기 위해서는 항상 이러한 메시지의 정의로 돌아가서 1)과제(테마)를 확인하고, 2) 상대방에게 기대하는 반응을 확인한다는 두 가지 확인 작업을 해야 한다. <그림 88> 의사 결정의 결과, 의견, 조언 등의 피드백 참조.



•이해
•의사 결정의 결과,의견,조언 등의 피드백
•행동

<그림 88> 의사 결정의 결과, 의견, 조언 등의 피드백

가. 확인1 : 과제(테마)를 확인한다.

커뮤니케이션(Communication)에서 자신이 상대방에게 답변해야 하는 과제는 무엇인가를 확인하는 것이다. 그것은 10분간의 설명, 한 시간의 거래 및 보고서, 제안서, 기획서를 만드는 일이라 하더라도 동일하다.

“내가 지금 상대방에게 답변해야 하는 과제는 무엇인가?”라고 자문자답해보자. 당신의 생각이 아무리 훌륭한 것일지라도 ‘과제(테마)’가 빗나가 있다면 상대방의 검토 대상조차도 될 수가 없다. 그것이 상사나 본사로부터 부여된 과제, 자신이 설정한 과제라 해도 마찬가지의 경우이다.

비즈니스 현장에서 처음부터 과제를 완전히 잘못 인식하고 있는 경우는 흔치 않을 것이다. 누구라도 처음에는 과제를 올바로 인식한 다음에 검토를 진행한다. 그러나 검토하는 동안에 마음에 걸리는 사안이나 그때까지 보지 못했던 과제가 생기면 그쪽

에 정신이 팔려 어느 사이에 자신의 머릿속에서 과제의 교체가 발생한다. 검토에 열중하면 할수록 이런 일은 자연스럽게 발생한다.

예를 들면 “안건 A의 사업화에 착수해야 하는가?”라는 과제에 관해 검토하고 있었다고 하자. 그런데 안건 A의 사업화는 고사하고 사업화의 전제가 되는 기존의 판매망에 중대한 문제가 있다는 것이 밝혀졌고 사태는 급박하다. 그러면 일의 중요성 때문에 어느 순간 당신에게 있어서 과제는 “기존의 판매망이 안고 있는 중대한 과제를 어떻게 해결해야 하는가?”로 바뀌어 버린다.

비록 이러한 문제의식 자체가 옳고, 안건 A보다도 기존의 판매망에 관한 우선적으로 의논해야 하는 것이라 해도 안건 A의 사업화를 의논하기 위해 모인 구성원에게 돌연 “오늘은 일의 중요성을 비추어 기존 판매망의 현황과 과제에 관해 논의합시다” 라고 털어놓는다면 어떻게 될까? 과제가 달라진 것, 왜 그럴 필요가 있는지를 명시하지 않으면 당신이 바라는 논의는 시작될 수 없을 것이다. 과제가 ‘향후의 일’이라고 한다면 모처럼의 제언도 ‘향후의 일’처럼 되어버린다.

문서를 작성하기 전에 또는 다른 사람에게 설명을 시작하기 전에 “오늘의 과제(테마)는 무엇이었지?”, “이제부터 설명하는 것은 OO이라는 과제(테마)에 관한 것”이라고 과제를 확인하는 일을 습관화하길 바란다. 아무리 자신이 “이것이 중요합니다!”라고 강조하더라도 상대방이 그 과제를 ‘지금 검토해야 할 문제’라고 인식하지 않으면 논의하는 것에조차 진출하지 못하는 골이 되고 만다.

최근 유행하는 것의 영업 제안의 어려움은 바로 여기에 있다. 수주 영업의 경우 고객이 무언가 불편을 느끼고 있거나 개선을 하기 위해 상품과 서비스를 주문하기 때문에 누구보다도 고색 자신이 과제를 명확히 인식하고 있다. 그런데 영업 제안은 제안을 하는 측이 제멋대로 “이것이 귀하의 과제입니다.”라고 전하고, 그 과제에 대한 해결책으로서 자사의 상품이나 서비스 등을 제안하는 것이다. 고객 측이 가령 잠재적이라도 제안자처럼 문제를 인식하고 있다면 고마운 일이다. 그러나 문제의식이 전혀 없는 고객의 경우 상품과 서비스를 제안하기 이전에 왜 그것이 필요한지, 어떠한 문제가 있는지를 인식시켜 주는 것, 바꾸어 말하면 상품과 서비스를 필연성을 나타내기 위해 고객이 지금 어떤 과제에 직면해 있는지에 관해 공통 인식을 형성하는 일이 대전제가 된다. 이런 점까지 생각하지 않으면 아무리 서비스와 상품을 권한다 해도 바람직한 답변이 되돌아오기를 기대하기는 어렵다.

‘자신밖에 모르는 병’에 걸려 있는 사람은 “내가 지금 말하고 싶은 것, 말해야 하는 것은 무엇일까?”라고 생각한다. 먼저 이것을 바꾸자. 올바른 접근은 “내가 지금 상대방에게 답변해야 할 과제(테마)는 무엇일까?”라고 자문자답하는 것이다. 그러면 자연히 “그 과제(테마)에 대한 나의 답변은 무엇일까?”라는 질문에 도달하게 되는데 그 전에 한 가지 더 확인해 둘 것이 있다.

나. 확인2 : 상대방에게 기대하는 반응을 확인한다.

회의를 하거나 문서 작성을 통해서 상대방이 어떻게 행동하기를 기대하는지, 어떤 반응을 이끌어내고 싶은지와 같은 기대성과가 없는 커뮤니케이션은 '독백'에 지나지 않는다. 그리고 '독백'과 상대해줄 만큼 한가로운 기업도 사람도 많지 않다.

비즈니스(Business)에서 상대방에게 무언가를 전달한다고 하는 행위 자체가 목적인 경우는 매우 드물다. 상대방을 이해시키거나 상대방의 니즈와 의견을 이끌어 내거나 상대방이 무언가 실행(Action)을 취하도록 하는 등 상대방으로부터 무언가 '반응'을 얻어내는 것이 최종 목적이다. 전달하는 것은 수단이지 목적이 아닌 것이다.

예를 들면 상사와 30분간 미팅을 한다고 해보자. 머리가 자신이 설명할 사항으로 가득 찬 상태에서 미팅에 임하는 사람과 30분 후에 상사가 자리에서 일어날 때 "자네가 말한 A, B, C라는 안건 중에서는 난 B가 좋다고 생각하네. 다음에는 코스트 분석(Cost Analysis)과 관련 부문의 의견을 청취해본다면 어떨까?"라는 상사의 의사표현과 지시를 이끌어 내보자고 생각하면서 미팅에 임한 사람과는 미팅의 성과에 큰 차이가 날 것이다. 또 고객에게 15분간 서비스 설명을 한다고 해보자.

이것 또한 '어쨌든 15분간 설명해보자' 라고 생각한 사람과 15분 후에 고객으로부터 "그러면 우리 회사라면 구체적으로 어떤 서비스를 어떻게 제공해 줄 수 있는가?"라는 질문을 이끌어 낼 수 있어야 성공이라고 생각하며 설명에 임하는 사람과는 설명 내용 자체가 달라진다.

커뮤니케이션(Communication) 뒤에 상대방으로부터 어떤 반응을 이끌어낼 수 있다면 그 커뮤니케이션은 성공이라고 말할 수 있는가? 사전에 이러한 질문의 답을 준비해두는 것은 '자신밖에 모르는 병'을 예방하기 위한 처방전이다.

이렇게 말하면 영업에 종사하고 있는 분 중에는 "영업 활동의 목적은 통상 매출을 올리는 것이기 때문에 고객에게 기대하는 반응 따위는 꼼꼼히 생각할 것이 있나?"하고 반문하는 사람이 있을지 모른다. 그러나 한번의 영업 활동으로 수주에 이르는 경우는 드물기 때문에 미리 계획을 짜둔다.

예를 들면 첫 번째 방문에서는 자신과 자신의 회사, 그리고 상품을 알리는 것을 목적으로 한다. "아, OO주식회사가 이런 상품을 내놓았구나! 꽤 흥미로운 상품이네!"라고 상품에 관심을 갖도록 한다.

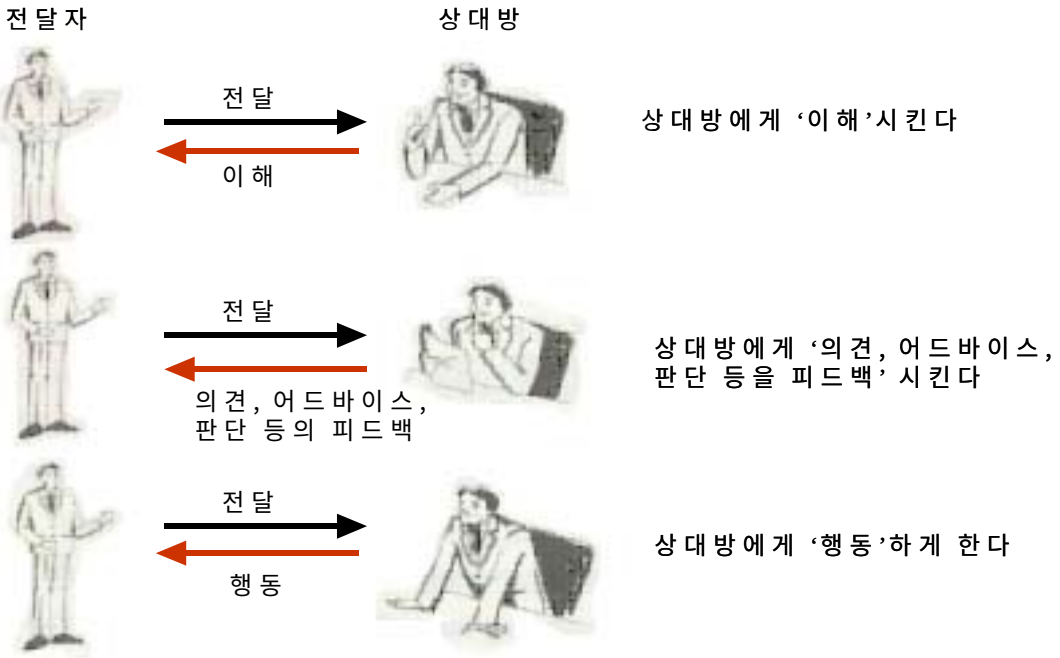
그리고 두 번째 방문할 때는 당사의 신상품이 종전의 다른 회사 상품과 무엇이 다른지, 그 고객에게 이 신상품을 사용하는 것이 어떤 이익을 주는지를 이해시킨다. 신상품의 경쟁 우위를 알리고 고객이 얻게 되는 이익을 이해시키는 것이 목적이다. 고객이 그 상품을 사용하는 이미지를 한층 더 구체적으로 갖도록 할 수 있을 것이다.

그리고 세 번째는 강력하게 동기 부여를 한다. 기간이 한정된 할인계획을 준비하거나 실제로 그 상품을 사용하고 있는 고객의 이야기를 들려줌으로써 "그렇다면 한번 사용해볼까?"하는 욕구를 이끌어낸다.

영업을 하는 사람이라면 누구라도 이런 영업 계획을 짤 것이다. 세 번의 영업 활동

의 최종 목적이 '사게 하는 것'이라도 각각의 커뮤니케이션으로 고객으로부터 이끌어내고 싶은 반응을 사전에 생각해둔다면, 지나치게 많은 정보로 고객을 소화불량 상태로 만들거나 고객이 강매라고 느끼게 하는 사태를 피할 수 있다. 또 잘 풀리지 않았을때 궤도 수정도 쉽다. 바꾸어 말하면 어떤 상대, 어떤 상황에서도 항상 같은 세일즈 토크(Sales Talk)를 주문처럼 외우는 사람에게는 "이 커뮤니케이션에서 고객으로부터 어떠한 반응을 이끌어내고 싶은 것인가?"라는 발상이 없는 경우가 허다하다.

비즈니스에서 상대방에게 기대하는 반응은 다음 세 가지로 귀착 된다. <그림 89> 상대방에게 기대하는 반응을 참조하자.



<그림 89> 상대방에게 기대하는 반응

● 상대방에게 '이해' 시킨다.

- 전달할 내용을 상대방에게 정확하게 이해시킨 뒤에 알고 있게 하고 싶은 경우. 업무 연락, 사무 연락 등 해당

● 상대방에게 '의견, 어드바이스(Advice), 판단 등을 피드백'시킨다.

- 전달할 내용을 상대방에게 정확하게 이해시킨 뒤 상대방이 그 내용에 관해 어떻게 생각하고 있는지, 찬성 또는 반대인지, 무언가 빠뜨린 점은 없는지 등 상대방으로부터 판단, 조언, 감상 등을 듣고 싶은 경우 해당
- 의견 청취나 테스트 마케팅으로 고객의 니즈(Needs)를 이끌어내는 경우 해당
- 사내 회의나 보고도 해당

● 상대방에게 ‘행동’하게 한다.

- 전달할 내용을 상대방에게 정확히 이해시킨 뒤에 상대방에게 실제로 어떤 액션을 취하게 하고 싶은 경우 해당
- 상품과 서비스의 판매 확대를 위해 판매 대리점 등의 제3자에 대해 판매 확장 시책의 설명, 캠페인의 참가 의뢰 등을 실시
- 상대방에게 앙케트에서 답을 얻는 경우 해당

같은 과제라도 자신이 전달하는 내용을 상대방에게 ‘이해’시키기만 하면 좋은 것인지, 아니면 자신의 생각에 대해 상대방의 의견이나 판단, 요구 등의 ‘피드백(feed back)'을 원하는 것인지, 또는 구입이나 판매 확대 등의 ’액션'을 취하게 하고 싶은 것인지 등 상대방에게 기대하는 반응을 확인함으로써 답변으로 전달할 깊이와 내용이 달라진다는 것은 쉽게 상상할 수 있을 것이다.

2) 무엇을 말해야 ‘답변’이 되는가?

자신이 이제부터 하려고 하는 커뮤니케이션에서 상대방에게 전달해야 할 ‘과제(테마)’를 확인하고, 그 대화 결과 상대방에게 어떤 ‘반응’을 기대하는지를 확인하고 나서야 비로소 답변의 내용을 생각하는 단계가 된다.

과제에 대한 답변을 정리할 때 많은 정보와 요소를 앞에 두고 매번 신음부터 하는 사람을 자주 볼 수 있다. 확실히 일하면서 답변해야 할 과제는 다양하고, 답변 내용 자체도 천차만별이다. 실제로 필자가 일하면서 접한 기업과 비즈니스맨은 업종도, 직면하고 있는 과제도 넓은 범위에 걸쳐 있다. 게다가 필자 자신은 그 기업에 속한 사람도 아니고 또 직접 사업 자체에 관여하고 있지도 않다. 그럼에도 불구하고 어째서 타인이 작성한 원고를 읽고, “여기는 뭔가 이상하다”, “이렇게 하면 좋지 않을까?” 하고 지적할 수 있는지에 대해 말해본다면 언제나 다음과 같은 질문을 자문자답하고 있기 때문이다.

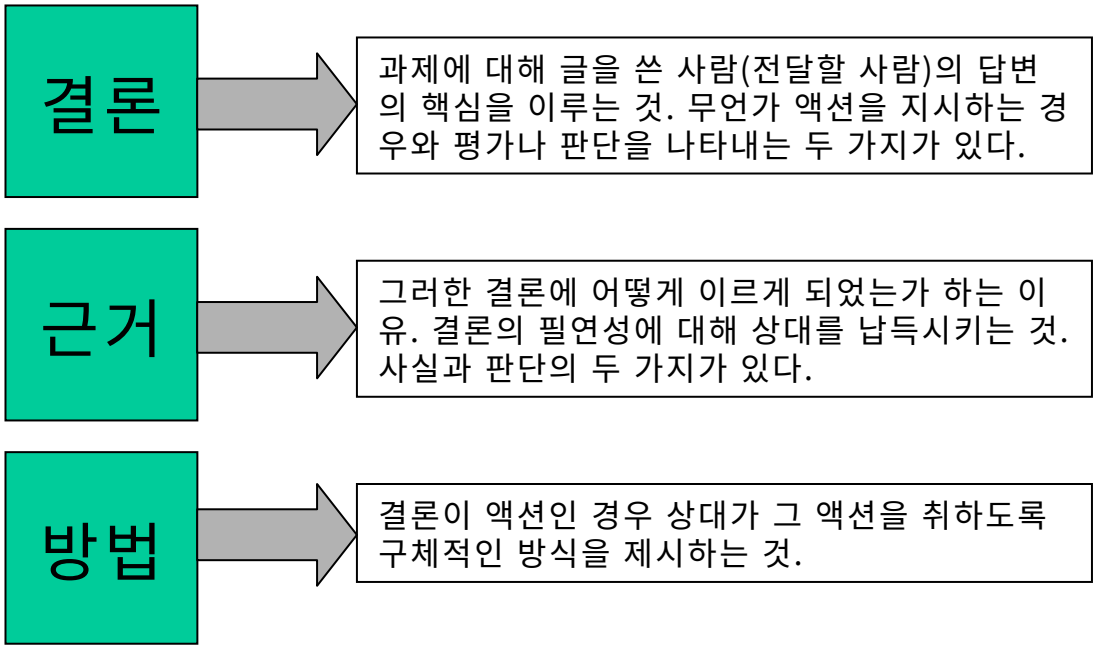
가. 과제에 대하여 전달자가 어떤 행동을 취해야 한다고 말하고 있는가? 긍정인가 부정인가? 또는 어떤 의견을 가지고 있는지가 확실히 머리에 남아 있는가?

나. 그 결론에 도달한 근거에 설득력이 있는가?

다. 결론이 액션인 경우 구체적인 실행 방식이 표시되어 있는가? 자신이 그 액션에 대하여 부하에게 지시하는 경우를 상정할 때 지시의 내용이 구체적으로 떠오르는가?

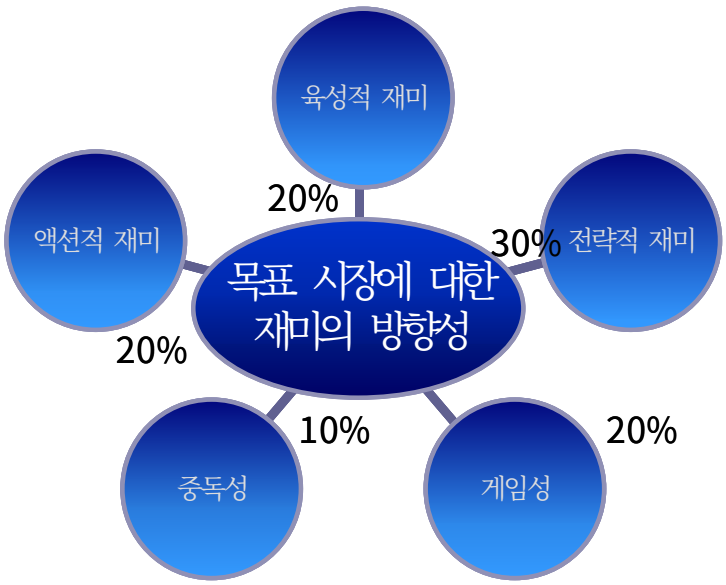
이런 질문에 긍정이라고 답할 수 있는가 없는가가 과제에 대한 답변의 요소가 있는가 없는가 사항이 체크 포인트(Check Point)이다.

비즈니스에서 과제의 답변으로 준비해야 할 요소는 과제에 따라 다르지 않다. 답변의 요소는 단지 3개뿐이다. 그것은 “제 답변을 한마디로 말씀드리자면, 요컨대...”로 설명하는 답변의 핵심 부분인 ‘결론’, 나아가 그 결론이 액션인 경우에는 그것을 어떻게 실행하는지를 설명하는 ‘방법’이다. <그림 90> 답변의 요소를 참조하자.

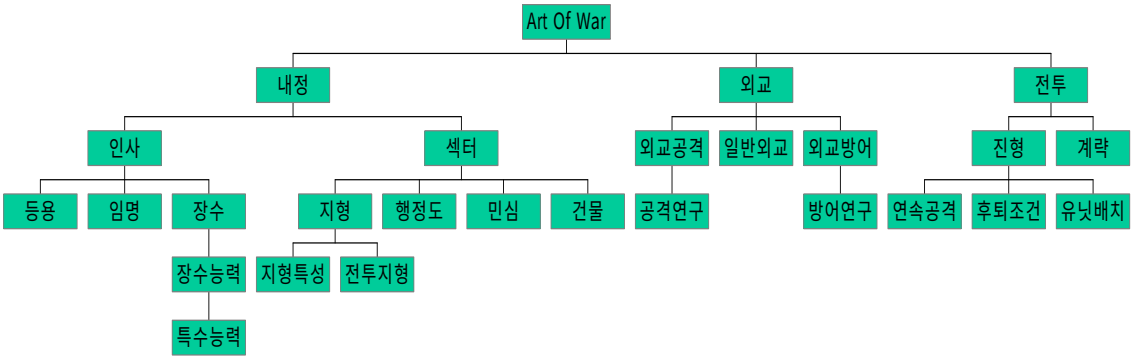


<그림 90> 답변의 요소

2. Open Design Tool에서 게임 기획 요소 밸런싱



<그림 105> ODT 게임기획 요소

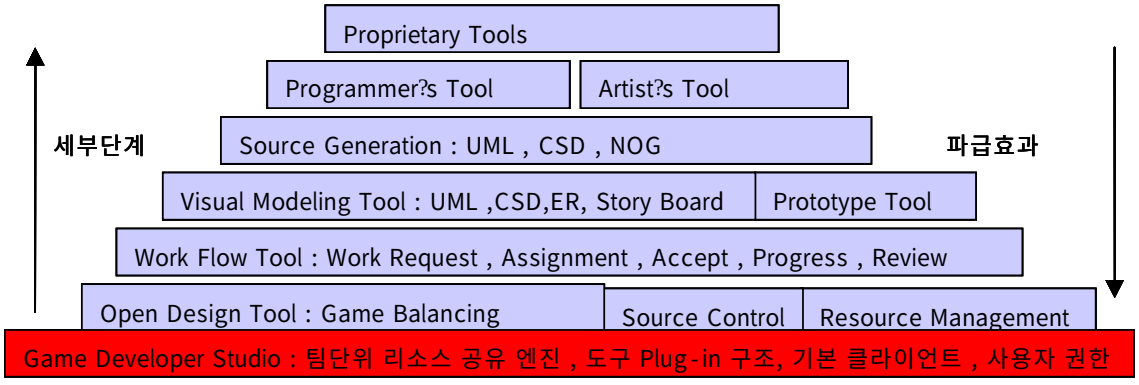


<그림 106> ODT 게임기획 요소 예제

ODT에서 게임 기획 요소를 트리구조(Tree)으로 나타내고 그 상세 기획내용은 문서(Document) 형태로 최근의 내용이 게시판 형태로 저장된다. 최초의 게임 기획의 방향성이 어떤 재미를 추구한다.

하지만 아이디어 단계에서 브레인스토밍을 통한 많은 아이디어가 제시되고 게임에 최대한 반영을 하다보면 처음 기획의도와는 상관없는 즉, 아무것도 아닌 종합선물세트 게임이 되기 쉽다.

ODT의 게임기획 요소 밸런싱은 이러한 것을 사전에 방지.예측하기 위해서 각 트리(Tree)의 노드(Node)에 무게(Weight)값을 주어 아이디어를 추가할 때 무게(Weight)값을 자체 평가하여 균형을 맞추도록 한다. 그리고 의사결정을 함에 있어서 중요도에 따라 쉽게 분간할 수 있게 되므로 회의진행시 의견 조율이 쉽게 일어나도록 한다.



<그림 110> Game Developer Studio 구성

<그림 110>에서 ODT를 상세히 다루고 요약한 것이다. 참고하도록 하자.

기 능	설 명
게시판	- 팀원의 아이디어, 전달 및 제공, 토론
게임 밸런싱 트리	- 게임의 기획요소를 최초의 방향성을 잃지 않도록 밸런싱 트리를 사용 및 유지
협업 기획	- 모든 팀원들이 게임성에 대한 검증에 참여 - 좋은 아이디어 제시 - 문서를 공유하여 편집 및 통합 - 최종 기획안에 의사결정과정 기획자 노트 포함
버전 콘트롤	- 기획 문서의 히스토리 기능, 데이터의 히스토리 기능
게임 기획 Rating	- 마케터, 기획자, 프로그래머, 그래픽 아티스트의 관점에서 그 기획요소의 다른 평가 - 하나의 게임 기획요소에 대해서 여러 가지 관점에서 점수 평가로 기획에 있어서 여러 가지 관점에서 조망
프로젝트 관리	- 기획자의 과점에서의 공정 다이어그램을 사용하여 프로젝트를 관리 - 주로 기획요소를 생성 및 점검하는 역할
공정 패턴 템플릿	- 게임의 장르나 플랫폼에 따라 다른 게임 개발 공정 - 미리 제시된 템플릿을 사용하여 프로젝트를 관리
게임 역공학 기능	- 기존의 게임의 기획 요소들을 역공학하여 트리로 나타냄 - 실패.성공한 게임의 주 원인을 파악
게임 개발 지식 베이스	- 정보들이 많이 쌓여서 데이터베이스의 역할을 하게 되면 검색기능이 필수적 - 추후 프로젝트에 경험지식을 적용
GDL (Game Development Language)	- GDL이라는 새로운 표준 게임개발 언어를 개발하여 팀원간의 의사소통을 원활
업무 진행 관리	- 팀원이 자신의 과제를 확인하고 검수를 받고 팀전체의 업무가 어떤 것이 있는지 파악
다른 도구와 연동	- 다른 도구와 연동되어 통합 도구로 사용

<표 13> 게임 공정상의 진행 설명

3. 게임개발공정관리에 대한 종합 결론

1. 프로젝트 매니지먼트의 관점에서의 대책

프로젝트 계획은 A 지점에서 B 지점으로 도착하는 방법을 알려주는 도로지도와 같다. 항상 계획은 불확실성을 갖고 있어서 완벽하지 않다. 복잡도가 높다고 해서 불확실성이 높은 것이 아니라 단계적인 발견에 의해 의존되는 경우의 불확실성이 높은 것이다.

게임개발 프로젝트의 경우 새로운 기술이나 개념을 도입할 시기가 되었을때 이러한 불확실성을 인지하고 계획을 세우느냐 그렇지 않느냐에 따라 그 계획이 성공적으로 수행이 될지 안될지를 파악할 수 있다.

계획에는 항상 예측과 다른 돌발 상황이 있기 때문에 현실의 진행과 반드시 일치하기는 어렵다. 그러므로, 통제(Control)를 통해서 일정관리를 해야 하고 이러한 통제에 있어서도 허용범위를 프로젝트의 특성에 따라 설정하여 허용범위와 지나치게 차이(Gap)가 나는 일정에 대해서는 특별하게 관리한다. 또한 전체 일정을 바로 잡는데 힘써야 한다.

계획과 통제에 대해 의구심을 가진 사람들은 이 계획과 통제 자체에 신경을 가지게 됨으로써 생기는 비효율을 지적한다. 이것은 프로젝트의 성격마다 다른 수준의 계획과 통제 수준이 요구되므로, 게임 프로젝트에 대해서도 적당한 수준을 찾아내야 할 것이다. 대개의 경우 프로젝트의 복잡성, 규모, 불확실성 수준, 조직의 요구사항, 프로젝트 도구의 편리성에 의존하여 계획과 통제의 수준이 결정된다.

게임 프로젝트의 경우에는 신규 장르를 개척할 경우에 불확실성이 매우 높고, 신규 소재의 경우도 그렇다. 게임은 독창성이 그 생명이라고 할 만큼 차별성을 가져야 상품가치가 있으므로 대개의 경우 불확실성이 높다고 말 할 수 있다.

그럼에도 불구하고 독창성을 확보하기 위해서 방종에 가까운 프로젝트 관리를 하는 사례가 많이 있고, 그로 인해 예산 초과 및 다급하게 마무리를 지어 결국은 졸작으로 끝나게 된다. 게임개발 프로젝트에도 이제는 프로젝트 관리가 필요하다는 인식이 필요하고 정규 교육으로 그러한 마인드가 함양된 인재들을 육성하여야 할 것이다.

2. 소프트웨어 공학적 관점에서의 대책

게임은 하나의 소프트웨어(Software)이다. 소프트웨어가 가지는 본질적 특성때문에 프로젝트 관리의 어려움이 존재한다. 즉, 복잡도, 적합성, 변경성, 불가시성이 그 이유이다. 소프트웨어는 거대해질수록 그 복잡도가 증가한다. 객체지향 패러다임으로 모듈 또는 객체간의 복잡도를 감소시켰는데 이것이 모든 해결책은 아니다. 본질적인 과제의 복잡도를 소프트웨어가 해결하기 위하는데 있어서도 여전히 복잡도가 남아 있는 것이다.

복잡도는 테스트, 유지보수에도 어려움을 겪게 만든다. 적합성이란 기존의 시스템과의

연동을 위해서 불필요한 복잡도를 갖게 되는 경우를 말한다. 하드웨어가 아닌 소프트웨어라는 용어의 차이점에서 오는 변경요구에 대한 압력 때문에 처음부터 새로 만드는 것이 빠른 경우에도 변경요구를 받아들여야 하는 경우가 생긴다. 그러므로 유용한 소프트웨어가 되려면 그러한 변경요구를 받아들여서 진화할 수 있는 구조로 만들어야 한다.

불가시성이란 소프트웨어를 눈으로 보여주기 힘들다는 점이다. 소스코드로 만들어진 도큐먼트는 그 소프트웨어를 파악하는데 거의 도움이 되지 않는다. 좀 더 추상화된 고급 레벨의 언어, 다이어그램, 개념도를 통해서 소프트웨어를 이해하는데 도움이 된다. 게임의 경우에는 동적인 모델이 많아서 특히 불가시성이 높은 분야라고 볼 수 있다. 게임을 위한 특별한 의사소통도구가 필요하다.

SW-CMM은 소프트웨어 개발 회사의 개발수준을 측정하는 도구이다. CMM Level 1은 최하위수준으로써 대부분 몇몇 사람에만 의존적인 개발 공정을 가진 회사를 말한다.
Level 3에서는 CASE Tool의 도움을 받고, 공정이 문서화되고 관리적, 기술적인 측면이 분명히 정의되어 소프트웨어 품질을 높이는데 노력을 들인다. 게임개발회사를 위한 GSW-CMM을 새롭게 정의하여 게임개발사들의 개발공정 레벨의 측정 및 컨설팅으로 개발사들의 효율을 증진시킬 수 있을 것이라 판단된다.

일반적인 소프트웨어는 7단계의 생명주기를 갖는데 요구사항 단계, 명세 단계, 설계 단계, 구현 단계, 통합 단계, 유지보수 단계, 폐기 단계로 구성된다. 각 단계를 확실히 구분하고 각 단계에서의 목적을 이루고 다음 단계로 넘어가는 것이 중요하다. 단계마다그에 맞는 테스팅 방법이 다르다. 게임개발에서도 단계를 잘 나누고, 명확한 단계에서 목적 설정 및 달성 여부의 검증 테스트 방법을 개발하여야 할 것이다.

3. 게임 개발 관점에서의 대책

컴퓨터 게임이란 컴퓨터를 통해 즐거움을 얻는 것이다. '컴퓨터 게임을 한다'라는 의미는 '플레이어가 조작하는 캐릭터와 컴퓨터가 조작하는 캐릭터(Character)의 상호작용'의 연속적인 행동을 말한다.

컴퓨터 게임은 컴퓨터의 출현과 더불어 시작되었다. 퍼스널 컴퓨터(Personal Computer)가 생기기 이전에는 아케이드(Arcade) 장에서의 게임기가 있었고, 메인프레임급의 컴퓨터에서도 컴퓨터게임은 있었다. PC(Personal Computer)시대에 들어오면서 많은 종류의 게임이 생겨나게 되었는데, 크게 5가지로 분류된다. 액션게임(Action Game), 어드벤처게임(Adventure Game), 보드.퍼즐게임(Board.Puzzle Game), 롤플레잉게임(RPG Game), 시뮬레이션.전략게임(Simulation.Strategy Game)이다.

게임을 할 수 있는 플랫폼(Platform)도 다양해져서 아케이드, 가정용 게임기, PC, 모바일 게임기, 휴대폰 등 여러 가지가 있다. 그 플랫폼에 따라서 게임의 특성도 다르게 디자인된다.

컴퓨터 게임의 제작은 일반 소프트웨어와는 달리 기술적인 요소가 상대적으로 덜 중요시되고, 컨텐츠적인 성향이 짙으므로, 요구사항도 잘 정의되지 않고, 자주 바뀐다. 그러한 점 때문에 사업성보다는 도박성이 짙은 면이 있으므로, 대규모 프로젝트보다는 소규모 프로젝트가 대부분이다.

게임개발에도 이제는 장르별로 정해진 프로세스가 생겨났는데, 공통점은 있다. 프로젝트가 시작되면, 게임의 핵심요소 설정, 게임의 내용 및 흐름 결정, 세부사항 결정, 구현, 테스트의 과정을 반드시 거친다. 일반 소프트웨어와는 달리 그래픽이나 음악적인 요소가 중요시되므로 개임의 역할이나 담당부서의 역할도 일반 소프트웨어와는 다르다고 볼 수 있다.

앞으로 게임 제작 기술은 여러 가지 미래 기술들의 종합체인 첨단 기술의 총아로서 자리잡게 될 것이다. 그러므로, 게임 제작은 종합 예술과 종합 기술로서 개척할 부분이 많이 남아 있다.

4. 사례조사에 대한 대책

앞서 언급한 게임개발 프로젝트의 어려움 때문에 발생하는 사례들을 게임잡지의 사후분석을 통해서 조사하였다. 많은 게임개발회사들이 게임성 높은 게임을 개발하려고 노력하고 있으나 여러 가지 여건으로 인해 어려움을 겪고 있다. 이러한 문제점을 여러 가지 면에서 돌출되었는데 첫째, 게임개발이 학문적으로 연구되지 않았고, 둘째, 전산 비전공자들이 게임을 개발하는데 있다.

프로젝트 관리 비전공자들이 겪었던 이러한 경험지식을 철저히 분석하고 습득하여 게임개발 회사들의 시행착오를 줄이고 성공적으로 게임을 개발하여야 한다.

5. 설문조사 결과에 대한 대책

사례조사를 통하여 설문을 넓게 8개의 문항으로 나누었다. 200여개 업체의 국내 게임개발자들에게 현재 게임개발의 전반적인 과정을 조사하였다.

설문조사에서 경력별, 직위별 의도하는 것이 다르고, 이들이 조화롭게 모두의 욕구를 충족시키기 위해서는 게임성 있는 게임을 추구하되 현실적인 일정으로 무리 없이 진행하여 출시일에 맞추는 것이라는 것이 지배적이다.

게임성 있는 게임을 만들려면 양질의 자료수집 및 종합, 팀원들의 게임 기획에 참여하면서도 애초에 기획한 게임성을 개발하면서도 방향성을 잃지 않고 유지하는 것이다. 또한 일정을 잘 맞추기 위해서 정기적으로 게임개발의 진행상황을 점검하고 공유하여 새로운 기획내용에 맞게 개발해야 한다. 특히 예산 책정과 일정 계획 조율 측면에서 개발업체의

실패원인으로 작용하는 것으로 설문조사에서 확연히 보여주었다. 필요한 게임개발 도구로는 향후 확장 가능한 통합 개발 도구 위에서 팀원들이 협업할 수 있는 환경이 준비되고 최신의 기획내용을 공유할 수 있는 시스템 즉, 형상관리를 필요로 하는 것으로 나타났다.

6. CSD와 ODT의 특징

여러 가지 대안에서 가장 차별성이 있는 대안은 CSD(Command State Diagram)와 ODT(Open Design Tool)이다. 아래 <표 14> 에서 CSD 및 ODT 특징을 나타낸 것이다. 표 왼쪽에는 기능설명으로 나타냈으며, 각 기능별 CSD 및 ODT의 특징으로 조사하였다. ODT의 경우 기획 중심의 도구이고 CSD의 경우 설계 및 구현 중심의 도구이다. 자세한 사항은 도표를 참조하자.

조사에서도 나타났듯이 기획단계에서 효과를 높이는 도구가 가장 전체 게임개발에 있어서 효용성이 있는 도구이다. ODT가 그 역할을 할 것이다.

기 능	CSD	ODT
개발단계커버	- 설계+구현	- 기획(초기+상세)
프로젝트관리 기능	- 있음	- 있음
의사소통기능	- 있음	- 있음
게임기획 공헌도	- 보통	- 매우 높음
UI 측면 공헌도	- 매우 높음	- 보통
차별성	- 복잡한 게임 및 UI와 로직 프로그래밍의 병렬진행에 알맞음	- 게임 기획요소 분배 트리
워크 플로우	- 명령 상태 추적	- 기획자 중심
작업 공정율 파악 방식	- 명령 구현, 상태 구현, UI 구현	- 기획 요소 구현
복잡도 관리 방식	- 명령 파라미터 복잡도 관리	- 기획 요소간의 밸런싱을 통한 일관성 유지를 위한 복잡도 관리
다른 도구와의 결합성	- 유저인터페이스 에디터 + 명령형식 에디터 + 프로젝트 관리	- 프로젝트 관리 + 다른 모든 개발 도구

<표 14> CSD 및 ODT 특징

7. 향후 해외 시장 진출을 위한 전략적 대안 도출

NXN-Software의 Alien Brain의 경우 Sony Online Entertainment, Electronic Arts, Infograms 등의 업체에 제공하고 있는 DPM(Digital Product Management) 시스템 제공업체이다. 리소스 매니지먼트 도구(Resource Management Tool)에서 경쟁자가 없을 정도로 독보적이고 소스 콘트롤 도구의 경우 MS-Visual Studio를 포함, 게임개발업체에서 많이 쓰이고 있다. 특히, MS- Visual Studio에서 Source Safe를 활용한다.

게임개발의 경우 개발환경이 IT업계의 개발환경과 매우 다르기 때문에 내장된 많은 기능들이 필요 없다. 게임개발 업체의 경우 보통은 자체 제작한 게임엔진 위에서 개발하고 있고, 엔진에 맞는 개발환경을 구성함으로써 보다 양질의 게임을 제공할 수 있다.

C/C++ 컴파일러는 기술 선점하기가 어려우므로 라이센스(License)하거나 공짜 컴파일러를 사용한다. 게임개발에 적당한 개발환경 구성 및 자체제작 엔진에 맞도록 커스텀 마이징(Customizable)할 수 있다면, 많은 게임개발 업체들에게 도움을 줄 수 있을거라 확신한다.

또한 OOAD 도구(Object Oriented Analytic Development Tool)를 개발하는 회사는 많지만 게임개발 공정에 맞는 도구는 현재 전세계적으로 찾아보기 힘든 실정이다. 국내에서 조차 주먹구구식의 게임개발 공정이 이루어짐으로 시간, 인력, 기타 제반비용의 손실을 가져온다. 그리고, 또한 게임엔진에 맞도록 소스코드를 생성하도록 제작되어 있는 경우가 없다. 3장과 6장에서 살펴보았듯이 CSD(Command State Diagram) + UI Editor(User Interface Editor) + N/W Object Generator(Network Object Generator)를 통합한 제품은 찾아보기 힘들다. ODT(Open Design Tool)의 경우도 마찬가지이다.

ODT의 경우 중요한 기획단계에서부터 적용 및 게임개발 과정의 검색하는데까지 쓰이므로 활용가치가 가장 높다고 볼 수 있다. 그리고 게임개발업체의 제작환경과 중립적인 상황의 특성을 가짐으로 모든 게임개발업체도 손쉽게 사용하리라 여겨진다. 따라서 게임 디자인 도구로써 활용된다면 국내.외적으로 게임산업의 발전을 가져올 수 있다.

8. 종합

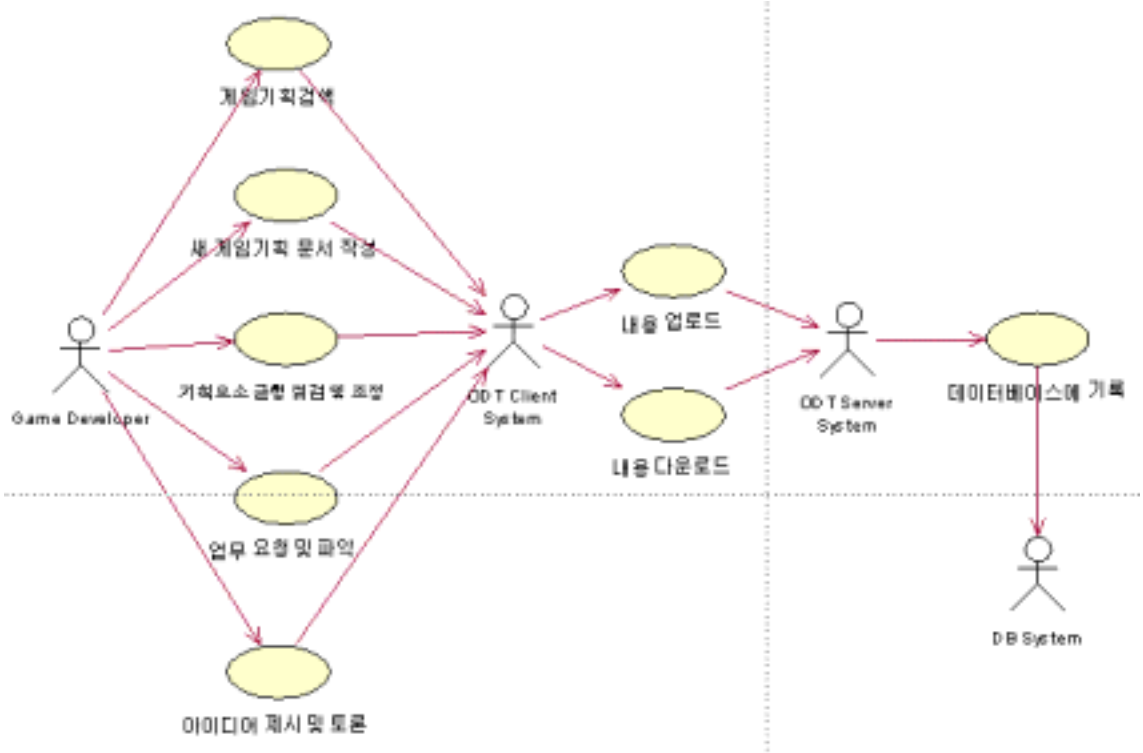
게임개발의 본질을 기초조사 및 사례 분석을 통하여 진행하였다. 즉, 게임개발업체에 종사자의 설문조사, 전화, 인터넷 검색, 유명서적, 경험적 사례를 통하여 게임개발 공정관리 자동화 도구 개발의 필요성을 제시하였고 이러한 결과로 성공적인 게임개발을 위해서는 프로젝트 관리 및 게임개발 도구가 필요하다는 것이 증명되었다. 또한 이러한 도구의 개발함에 있어서 미리 선행되어야 할일, 개발시 필요한 것, 문제점, 보강사항 등이 이번 연구보고서에서 나타내고 있다. 특히 게임개발도구 중 가장 중요한 것으로써 ODT를 개발하는 것이 시급하다. 그러나 ODT의 문서 공유기능이나 협업기능은 다른 도구와 비교할 때

새롭게 개발할 경우 경쟁력을 갖추기 어려우므로 ODT의 가장 차별적이고 필요한 기능인 게임 요소 밸런싱 트리기능을 처음 시작하고 개발하는 것이 중요하다. 그리고, CSD를 사용하여 GUI를 중심으로 한 프로젝트 관리 기능을 개발하면 충분히 경쟁력 있는 제품을 제작할 수 있다고 할 수 있다.

결론적으로 프로젝트 진행에 있어서 앞단계에서 잘 해내는 것이 중요하고, 마케팅 및 기획 방향성 설정, 기획부분이 그 부분이라고 할 수 있다. 그 부분에 있어서 도움이 되는 도구를 개발하는 것이 게임개발 전체에 영향을 미치므로 가장 효용성이 높다고 볼 수 있다. 기획의 내용을 팀원의 참여를 유도하여 통합하는 시스템을 잘 받쳐주기 위해서는 그만큼 융통성 있는 게임 개발 방법이 필요하고 CSD와 같은 설계 및 개발에 직접적으로 도움이 되는 4세대 도구들이 필요하다.

4. ODT(Open Design Tool)

1. ODT 상세 요구사항 분석



<그림 112> ODT Usecase Diagram

위 <그림 112>은 ODT Usecase Diagram을 나타낸 것이다. 업무의 효율화를 위한 진행 방식을 설명하고 그림에서 보여주고 있다.

1) 요구사항 기준(基準)

가. 문서화(文書化)

입과 귀를 통해서만이 아니라 공통된 도구를 추가로 이용해서 의사소통하고 작업을 진행하는 것은 생각을 기록하게 하는 강제 수단이 된다.

나. 표준화(標準化)

표준화는 그 언어와 도구의 양식(Format)을 사용하는 회사 내 팀원들 뿐만 아니라 다른 회사와 단체사이의 의사소통을 가능하게 한다는 점에서 의의가 있다고 볼 수 있다. 또한, 표준화된 포맷을 통해 접근하는 다른 목적의 도구와 연계해서 시너지 효과를 기대할 수 있게 해 준다.

다. 도구를 사용한다는 것

도구에서 사용하는 단위, 용어가 다른 개발 공정의 다른 영역들도 정형(定刑)화 시킨다. 또한 회의시간 외에도 일을 공동으로 진행할 수 있으므로 공통 시간을 갖지 않아도 게임의 내용이 수정되고 발전한다.

라. 게임 개발 도구로써 가치

게임이 가진 독특한 문화 산업적 특징을 인식해서 기획과 관련된 보조 수단으로써 기능을 수행할 수 있어야 한다.

마. 공정 도구로서의 가치

게임 제작을 하나의 산업으로 보고 지속적인 프로세스 경험을 자연스럽게 누적시킨다. 그리고 그것을 지속적으로 활용하고 공유할 수 있어야 한다.

바. 진화된 기획 문서

2) 역할로 정의한 ODT 요구사항

가. 협업 기획을 가능하게 하는 도구로서 ODT(Open Design Tool)

“디자인을 날짜에 맞추기” 라는 번역서(Game Architecture and Design 194 Page 참조)를 살펴보자.

과거의 일인 기획, 개발 시스템에서 발전해서 현재의 게임 개발 모델은 다양한 경험을 통해 각자의 분야에 장점으로 특화된 여러명의 기획자들이 함께 공동 기획하는 형태가 많아졌다. 또한 의사 교환이 효율적인 20인 미만의 소규모 회사에서는 기획을

담당한 사람 외에도 개발자나 그래픽 디자이너가 다같이 게임 기획의 진행에 도움을 줄 수 있다는 장점이 있어서 협업 기획의 요구가 많다.

사례 1 - 잘못된 공동 기획 과정

H사에는 보조 기획자가 둘 있다. 보조기획자 A는 꼼꼼히 자신의 생각을 잘 기록하는 편이어서 회의 시간에 그동안 정리했던 자신의 기획 내용을 메인 기획자에게 전달하곤 했다. 반면 보조기획자 B는 그 때 생각하는 것만 전달하는 스타일이었다. 프로젝트를 진행하는 과정에서 A는 자신이 지적한 내용이 수정이 되지 않고 여전히 문제로 남아 있다는 사실을 발견하고 메인 기획자에게 요청해서 다시 회의를 소집했다. 회의를 통해 메인 기획자는 A의 아이디어를 무시한 것이 없다는 것을 알아냈다. 문제는 메인 기획자가 어떠한 판단을 통해 아이디어를 거부하고 현재 상태를 유지하게 되었는지를 전부 기억해내지 못한다는 것이었다. 따라서 A와 메인 기획자는 다시 예전에 회의했던 내용을 반복하고 왜 그런 결정을 내렸는지 추적하는 일을 해야 했다. 보조기획자 B는 자신조차 어떤 의견을 냈었는지 기억을 못하기 때문에 아무 할 말이 없었다.

그러나 위의 사례처럼 지금까지는 기획에 대한 아이디어(Idea)를 제시했을때 최종 기획에 반영되기까지는 많은 장애가 있다. 오히려 능력 있는 일인 기획 체제에 비해 진정한 협업이 쉽지 않은 것이 현실이다.

게임 기획에 대한 협업은 회의를 통해 이루어지는 경우가 대다수이지만 아이디어의 제공자가 회의에 참여하지 않으면 그 아이디어는 기획에 반영되지 못한다. 또한 회의를 통해서 결정된 어떤 기획의 선택이 나중에 다시 문제가 되었을 때 기획의 분기가 어떻게 발생하게 되었는지 잘 알 길이 없다.

위 사항에서 ODT는 기획에 참여할 수 있는 모든 사람들에 대해 열려있는 발전형 기획서라고 정의할 수 있다. 기획서가 열려 있기 때문에, 과거 기획자 혼자서 의견을 모아 발전시키는 기획서와는 달리 진입이 쉽고, 모두에게 기획중인 게임의 항목에 대해 커멘트를 달고 의견을 제시할 수 있게 해 준다. 따라서 새로운 아이디어에 대한 자유롭고 자발적인 참여를 유도할 수 있는 동시에 협업 과정이 기록되므로 기획의 발전 상황을 남겨 놓을 수 있다.

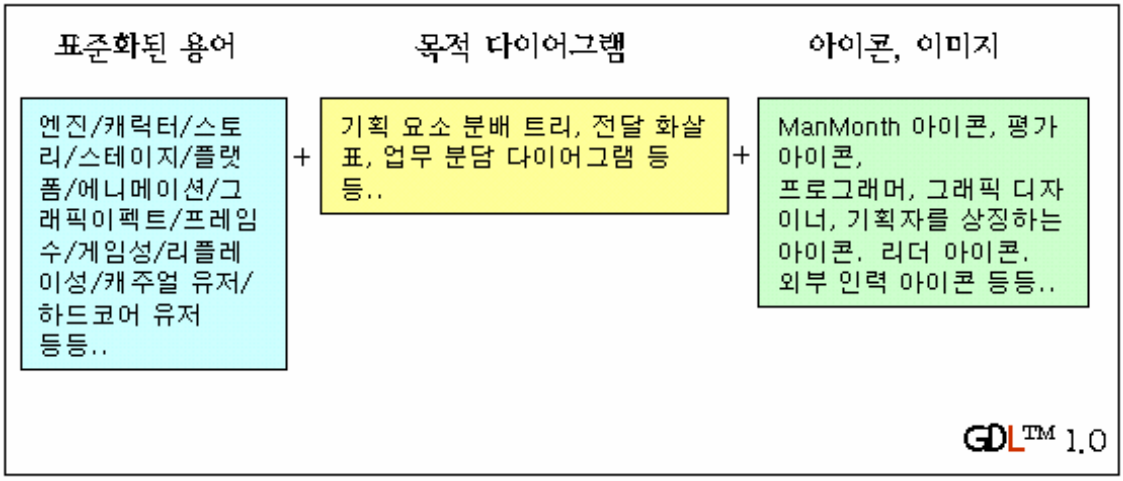
나. 프로젝트 팀원 간 의사소통 도구로서 ODT(Open Design Tool)

게임 개발의 분할된 조직으로 기획.프로그래밍.그래픽.마케팅 4가지로 구분된다. 게임 개발의 생산성은 이들 파트 사이의 유기적이고 정확하고 빠른 의사소통에 의해 크게 좌우된다.

이 파트(Part)들의 담당자간에 의사 전달이 잘 되지 않고 오해가 생기는 상황이 되

면 생산성이 극도로 떨어진다. 또한 사기도 같이 떨어진다. 지금까지 이런 의사소통의 문제는 같이 오래 일한 팀웍으로만 해결할 수 있는 것이라고 인식되어 왔다.

이에 본 연구에서는 ODT의 하나의 구성 요소로써 이들 간의 의사소통에 대해 표준화된 언어를 제시한다. 또한 GDL(Game Design Language)이라 명명한 이 언어는 프로그래밍 코드가 아닌 프로젝트 팀원 모두가 공통적으로 똑같이 인식할 수 있는 표준화된 용어와 다이어그램과 기호와 아이콘의 집합으로 나타낸다.



<그림 113> GDL의 구성 예제

● GDL의 조건

GDL에서 사용되는 모든 용어나 아이콘이 명확해야 하고 실제 그 사항과 정확하게 일치하는 용어는 아니라 해도 게임개발업체에서 사용하는 보편성을 확보하는 것을 우선으로 해야 한다. 세 파트(Part)중 어느 파트만 이해할 수 있는 용어가 되어서는 안 되며 만약 다른 파트에 없는 용어라면 그들 파트의 동의를 통해 정의해서 혼동의 소지가 없도록 한다. 또한 새로운 요구에 의해 발전하는 용어를 지속적으로 표준화하려는 움직임이 있어야 하고 이는 자발적인 형태가 가장 바람직하다.

ODT에서 팀원은 GDL이라는 표준화된 언어를 통해 의사소통한다. 따라서 그것이 신생팀이건 10년 경력의 베테랑 팀이건 용어의 주관적인 해석이 만들어내는 혼선을 없애고 정확한 의사소통을 도모할 수 있는 표준화된 길을 만들어 준다.

다. 기획 내용의 버전 콘트롤 도구(Version Control Tool)로써 ODT

게임의 사이즈가 클수록 기획에 참여하는 사람이 많아지고 그에 따라 기획의 항목도 늘어나고 복잡해진다. 많은 메인 기획자가 이렇게 생산된 수많은 기획 문서와 아이디어 제안서, 몇 달에 걸친 회의록, 직접 펜으로 수정된 기획안들 속에서 갈피를 잡지 못한다.

사례 2 - 현재 기획안이 도대체 뭐지?

Y사는 1년 남짓 개발 중인 게임 프로젝트를 진행하고 있다. 문제는 일정은 조금씩 늦어지는데 일이 어떻게 진행되어 가고 있는지 잘 보이지 않는다는 것이었다. 결국 H사에 컨설팅을 받기로 결정했다.

H사는 Y사가 1년 동안 작업한 기획안과 아이디어 문서, 회의록으로 구성된 한장의 CD를 전해 받았다. H사의 기획 컨설턴트 S는 우선 현재 최종 기획안이 어떤 것인지 확인하고 싶었다. 그렇지만 서로 상충하기도 하고 어떤 것이 다른 것을 수정한 것처럼 보이는 수많은 아이디어 문서들 속에서 현재 기획안을 결코 그려 낼 수 없었고 결국 Y사는 지불한 비용만큼 적절한 컨설팅을 받지 못한 결과가 되었다.

ODT는 첫째, 기획의 세부 항목을 명확하게 나누고 정의하는 것을 돕는다. 둘째, 수정된 사항을 시간별로 기억해서 하나의 항목에 대한 변경을 추적할 수 있게 해준다. 셋째, 기획 선택의 과정에서 왜 어떤 것이 선택되고 어떤 것이 선택되지 않았는지에 대한 선택 과정을 분기별로 유지한다.

사례 3 - 게임을 일관성있게 디자인하기

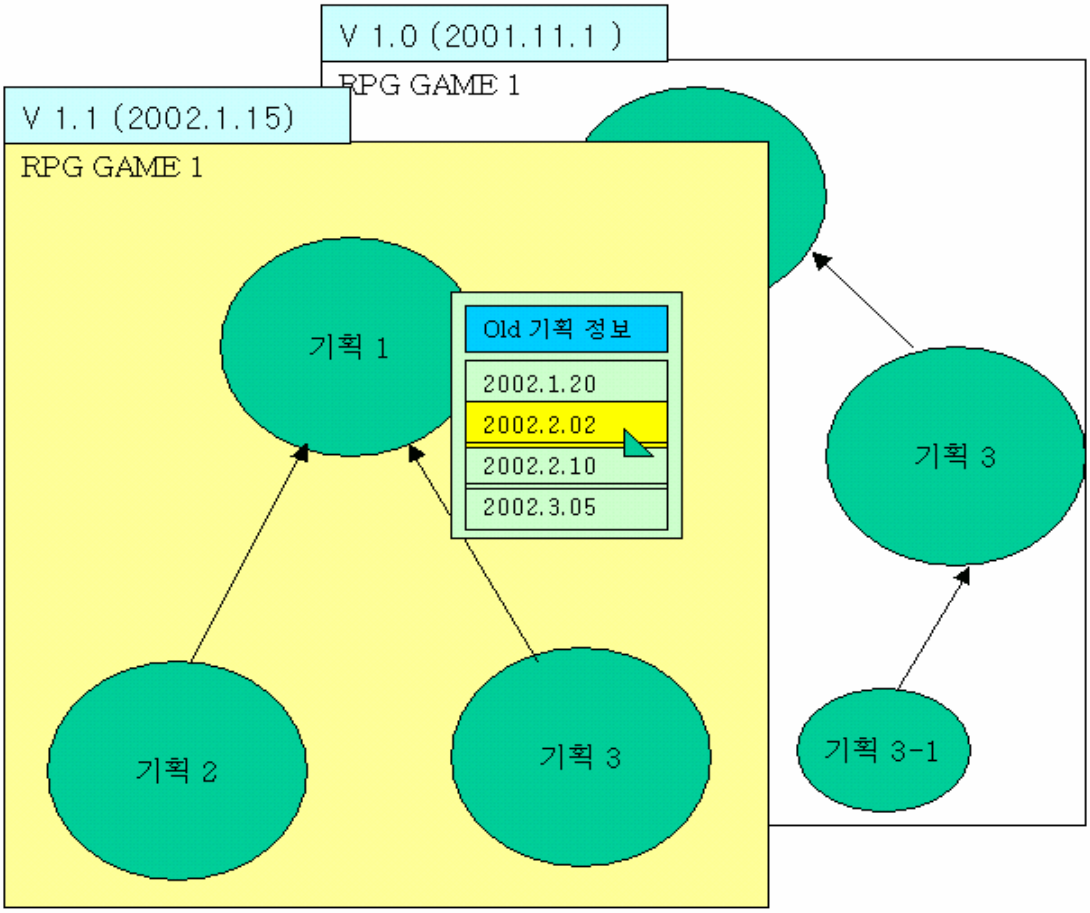
B사는 이번에는 정통 RPG를 만들기로 결정했다. B사의 기획자들은 각기 대단한 재능을 갖고 있어서 공동 기획을 통해 재미있는 아이디어를 무척 많이 확보해서 흡족해 했다. 보조 기획자 C는 평소 미스트(Mist)와 같은 게임의 퍼즐(Puzzle) 풀기에 재미를 느끼고 있었기 때문에 그와 관련된 좋은 아이디어를 생각해 냈다. 메인 기획자 D는 많은 키 조작을 아이들이 좋아하는 것을 보고 점프(Jump)나 대쉬(Dash)와 같은 다양한 액션성이 게임을 캐주얼(Casual)하게 만들고 어린 유저도 정통 RPG의 부담을 덜고 게임을 구입할 수 있게 만든다고 생각했다.

이런 아이디어를 살리는 과정에서 원래의 복잡한 매직 시스템과 넓고 방대한 맵은 간략화하게 되었다. 스토리도 같이 줄어들었으며 퍼즐에 의한 어드벤처성과 조작의 재미를 주는 액션성은 강화되었다.

개발이 윤곽이 드러나서 해볼 수 있게 된 시점에서 처음 지원하기로 했던 외부 마케팅 직원들이 와서 게임을 평가하는 시간을 가졌다. 그들은 이제 처음 얘기했던 정통 RPG가 아닌 복합장르의 게임을 보고 있었다. 나름대로 재미있는 요소들이 많다고 느꼈지만 그들이 본 틈새시장에 필요했던 정통 RPG도 아니었고, 다른 회사의 비슷한 게임을 이미 유통할 계획이 있었으므로 계약을 포기했다.

그림 114는 게임의 기획 버전이 변화하는 모습을 나타낸다. 게임의 기획은 아이디어가 발전함에 따라 , 또는 게임성을 위해서 바뀌게 되는 데, 그것을 필요시 추적할

수 있게 된다. 사례3은 게임의 일관성을 위해서는 기획자 노트가 첨부된 기획서 및 기획요소가 평가되어서 아이디어를 반영할 수 있도록 하면 게임의 일관성을 유지하면서도 신선한 아이디어를 수용할 수 있게 된다.

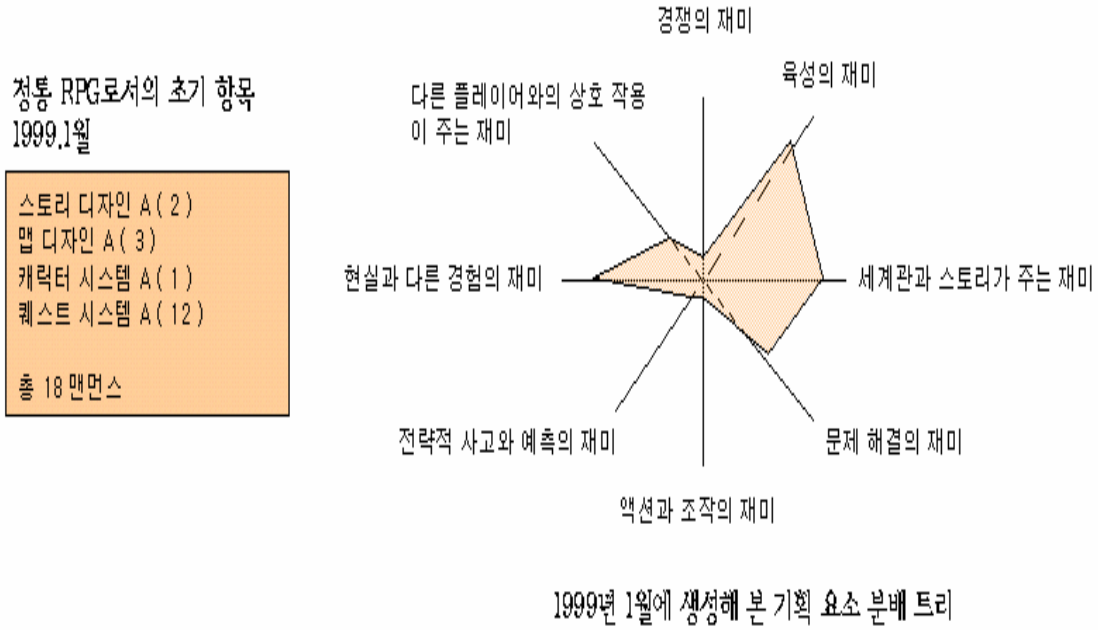


<그림 114> 버전 관리

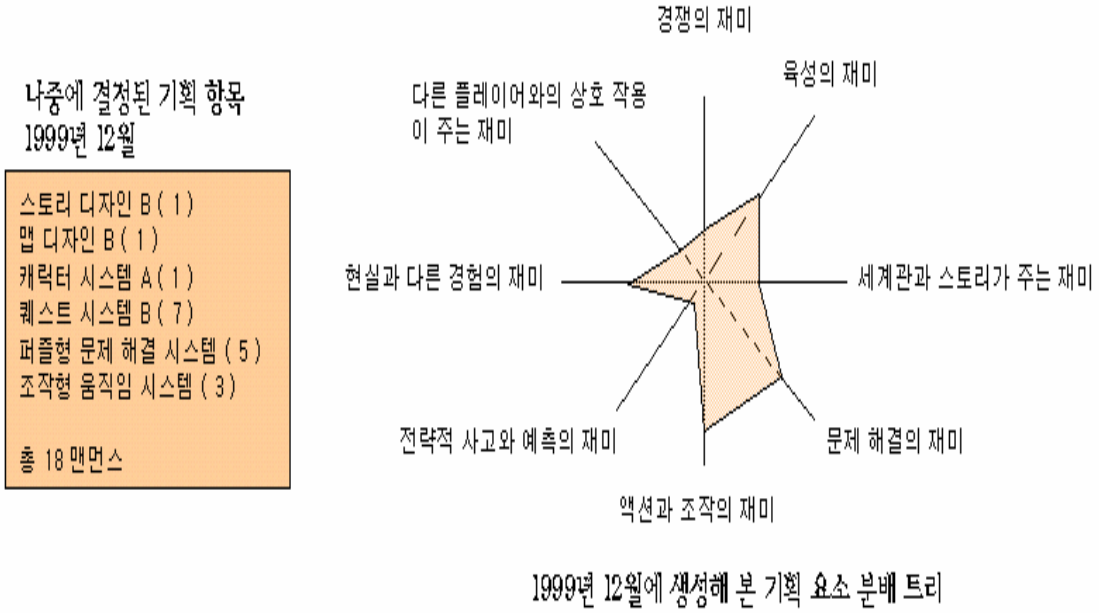
라. 게임 디자인 보조 도구

● 게임 요소의 할당 도구로서 ODT

위의 사례는 어떤 경험 많은 기획자라도 피할 수 없는 지뢰밭과 같다. 한 게임의 기획을 오래 생각하다 보면 현재 게임 기획이 어떤 상황에 와 있는지 객관적으로 분석하고 파악하는 게 무척 어렵기 때문이다. ODT는 특정 게임 기획안에 대해 어떤 장르적 특성과 게임성을 얼마만큼 주고 있는지 파악하고 조절할 수 있는 기능을 제공한다. 게임 요소 분배 트리라 불리우는 이 기능을 통해 기획자가 기획안에 대해 어떻게 유저에게 재미를 줄 지를 객관적으로 판단할 수 있다.



1999년 1월에 생성해 본 기획 요소 분배 트리



1999년 12월에 생성해 본 기획 요소 분배 트리

<그림 115> 기획 요소 분배 트리를 통한 확인 프로세스

● 게임 기획 레이팅(Rating) 통로로서 ODT

ODT는 각 기획 항목에 대해서 팀원이 평가할 수 있는 레이팅 시스템을 제공해 준다.

이런 평가는 부서별, 연령별, 특성별로 지정해서 판단해 볼 수 있어서 팀원들이 어떤 시각으로 게임의 디자인을 판단하고 있는가를 파악할 수 있으며, 기획 내용에

대해 팀 내적 객관성을 확보할 수 있다.

사례 4 - 다른 사람이 어떻게 생각하는지 알 길이 없군

S는 메인 기획자로 새로운 전략 게임을 구상하고 있었다. 초기에 회의를 통해서 전체적인 방향을 잡았으며, 게임의 규모가 크고 복잡하므로 S를 주축으로 공동 기획의 형태로 기획을 진행하기로 하였다. S는 기획하는 과정에서 여러 가지 아이디어들을 다른 사람들은 어떻게 생각하는지를 알고 싶었다. 그는 시간 나는 대로 팀원들에게 의견을 구했고, 정기적으로 회의때마다 물어보곤 했다. 그렇지만 팀원들은 각자 업무에 바빠서 별 신통치 않게 얘기하는 경우가 잦았으며, 잘 이해하지 않고 대답하는 경우조차 있었다. A는 예리한 시각을 갖고 있는 마케팅 부서의 C의 의견도 듣고 싶었지만 그는 지방 출장이 많아서 회의 시간에 거의 참석할 수 없었고 회사에 있는 시간이 적었다. A는 결국 크게 도움을 받지 못하고 나름대로의 기획을 할 수 밖에 없었으며 차라리 그냥 혼자서 생각해서 기획하는 편이 더 효율적이겠다는 생각을 했다.

차후 팀 외적 객관성을 확보한다. 즉, 완성된 게임의 항목에 대한 유저(User) 및 마켓(Market)의 반응을 통해서 가능하다고 했을때 그것과 대조해서 가장 유저와 가까운 시각을 가진 팀원이 누구였는가를 찾아내는 도구로써 의미도 있다. 즉 관리자 입장에서는 거꾸로 팀원에 대한 마케팅 및 기획 능력 평가 도구로써 활용될 수도 있다.

라. 게임개발 공정 도구

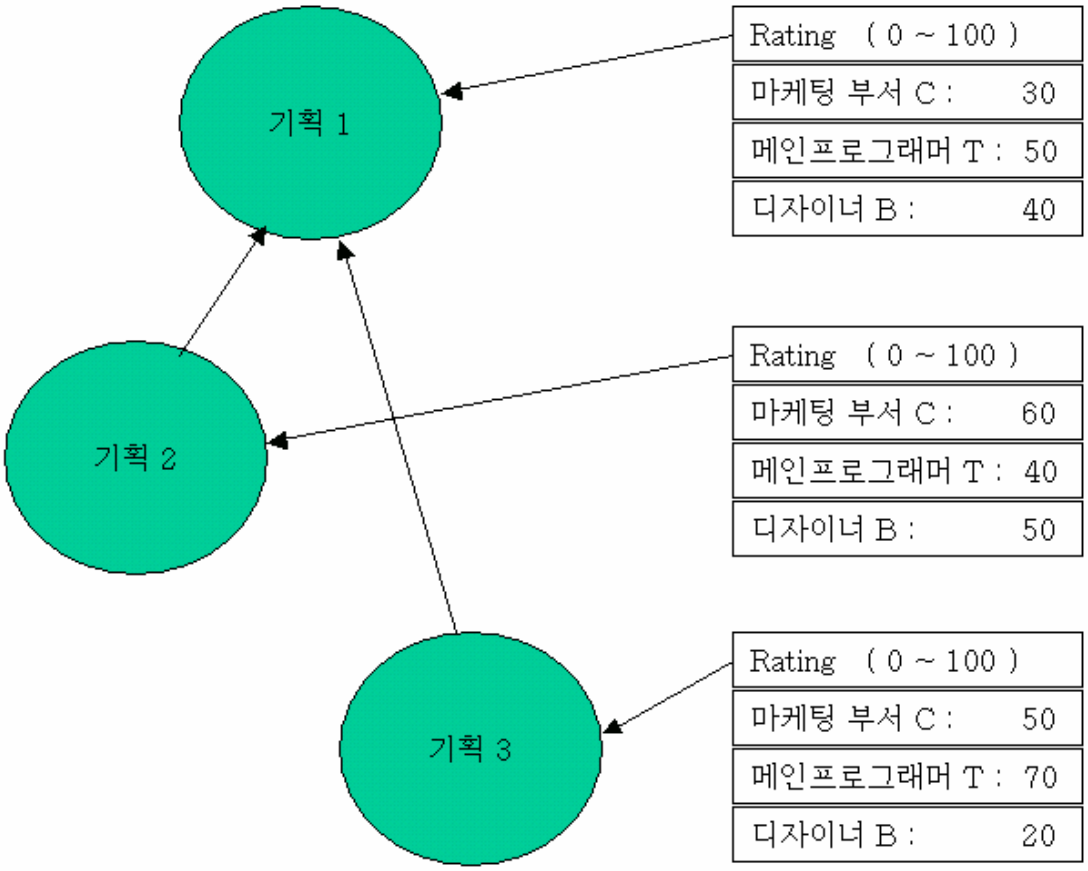
● 프로토타이핑 수단으로서 ODT

사례 5 - 여러개의 기획서를 구성해내는 어려움

B사는 차기작으로 간단한 미니 게임을 만들기로 결심했다. 기획 회의를 통해 좋은 아이디어를 여러개 발굴하고 각각의 게임성을 테스트해보기로 결정했다. 프로토타입 기획서를 제작하는 임무를 맡게 된 기획자 S는 회의 때 나온 아이디어들을 그가 이해한 나름대로 조직화해서 몇 가지 프로토타입 문서를 만들었다. 그중 많은 변형안을 갖고 있는 경우 일일이 프로토타입 문서를 작성할 수 없어서 그냥 “변형 가능성”이라는 토만 덧붙였다.

개발을 맡은 프로그래머들은 각각의 프로토타입을 성실하게 구체화 시켰다. 어느 정도 지나자 문제들이 드러나기 시작했다. 기획 당시에 나왔던 몇몇 좋은 아이디어들이 적절하게 프로토타입으로 반영되지 않았다는 의견이 나왔다. 어떤 프로토타입

은 기획서로 작성되지도 못했으며, S가 변형안으로 달았던 가능성들은 하나의 독립적인 기획안이 아니었으므로 프로그래머들에 의해 철저히 무시되었다. 또한 작업도중에 나온 새로운 아이디어들은 기존의 아이디어들과 조합해서 새로운 프로토타입을 제시할 수 있었음에도 불구하고 기획자 S의 추가 노력이 많이 든다는 이유로 배제되었다.



<그림 116> 특정 기획 항목에 대한 3 Part 평가 예제

게임 기획의 초기 단계에는 많은 양의 기획 요소들이 쓸지 안쓸지 확정되지 못한 상태에서 널려 있게 된다. ODT에서는 기획 아이디어들이 나온 후 기획서 생성 기능을 통해 기획 요소를 조합해서 다양한 버전의 프로토타입 안을 기획자의 문서화 및 체계화 작업을 거치지 않고도 쉽게 만들어 낼 수 있다. 또한 프로그래머에게 있어서는 각 프로토타입이 무엇을 테스트하기 위한 것이며, 어떤 성향을 지닌 기획안인지 이해하는 데 도움을 준다.

이 기획요소로부터 기획서를 생성하는 기능은 비단 프로토타이핑 시에만 사용하는 것이 아니라 현재 버전의 기획서를 확인하는 기능으로서 본래의 의미가 있다. 그렇

지만 가장 효율이 높은 경우가 많은 변형이 가능한 기획 작업이나 변형을 여러 가지로 시도해 볼 수 있는 제작 초기 단계 즉, 프로토타입 단계라 할 수 있다.

● 패턴화 도구로서 ODT

사례 6 - 공정을 모르면 일정을 모른다.

H사는 Windows의 DirectX 기반에서 게임을 만들어 본 경험이 많아서 어떤 플랫폼으로도 개발할 자신이 있었다. 그러나 다른 플랫폼인 Mobile 게임을 개발하는데 얼마나 걸릴지는 당장 예측할 수 없었다.

그들은 만들 확신이 있었음에도 불구하고 정확한 일정을 제시하지 못했고 용역을 수주하는데 실패했다.

ODT는 플랫폼, 장르, 주요 기술에 대해 표준화된 템플릿을 제공한다. 처음부터 필요한 기획 항목과, 공정 항목을 추가해서 일정을 예측하고 인력을 배치하는 방법이 있고, 어느정도 데이터가 확보된 후라면 그 상황에서의 보편적인 인력과 일정과 항목을 배치해 주는 것이 보다 빠른 접근을 가능하게 한다.

시나리오 ODT 1

H 사에서 다음 제작할 게임을 WindowsCE/ PPC 환경에서 정통 RPG 장르이며 3D시점을 제공하는 것으로 결정했다. 프로젝트 매니저인 A는 ODT의 Template 기능을 선택하고 플랫폼을 WindowsCE/ PPC, 장르를 정통 RPG, 주요 기술 중에 3D를 선택해서 프로젝트를 생성시켰다. 그 결과 디자인 측면에서는 RPG의 기본적인 하위 항목인 스토리(Story), 캐릭터(Character), 맵(Map), 퀘스트(Quest)와 같은 것들을 대략적인 맨먼스와 함께 포함시켜 놓았고, 공정 측면에서는 IsoMetric View System, WinCE-GAPI Module. WinCE-MultiChannel Sound Module과 같이 필요하다고 생각되는 항목과 기간을 배치해 놓고 있었다.

이런 데이터들은 자신의 회사와 다른 업계의 많은 ODT 프로젝트를 통해 조절된 항목이므로 일단 신뢰를 해서 수정하지 않고 대략적인 전체 일정을 알아낼 수 있었다. 향후 상세한 부분은 이 템플릿(Template)을 놓고 파트별 팀장 회의를 거쳐서 조절하기로 결정했다. 이렇게 함으로서 그는 잘 알지 못하는 내용에 대해 일일이 물어보고 추가할 필요가 없었으며 처음부터 프로젝트의 항목을 구성할 때 드는 많은 회의와 노력을 절감할 수 있었다.

마. 게임 디자인 역공학 도구로서의 ODT

Game Architecture and Design 이란 책의 과대망상증 부분을 살펴보면 게임 기획 회의를 하다 보면 얘기하려는 내용의 이해를 돕고자 이미 제작된 게임을 언급하는 경우가 많다. 의사소통 과정에서 각자의 경험과 주관적인 언어를 매개로 할때 서로 오해의 소지가 많기 때문에 모두 잘 알고 있다고 생각되는 스타크래프트(Starcraft)나 테트리스(Tetris)를 떠올리는 것이다.

문제는 이미 제작된 게임에 대해서조차 모두가 똑같이 생각하는 것은 아니라는데 있다. 기획회의때 오고가는 주관화된 언어와 마찬가지로 특정 게임에 대해서 역시 그들이 플레이할 때 본 것과 그것을 구체화한 내용이 모두 다르다. 오래된 게임일수록 더욱 그렇다. 게임 디자인의 어떤 항목이 무척 좋아서 언급했지만, 과거에 유명하지 않았던 게임이라 회의 탁자에 앉아있는 극소수만이 경험해 봤다면 그 역시 도움이 되지 않는다. 모르는 사람은 게임을 추측하므로 오해의 소지만 더 생길 뿐이다.

ODT는 GDL이라는 표준화 언어와 기획 항목의 데이터화를 토대로 기 개발된 게임 디자인에 대한 역공학 작업을 돕고 그 작업 결과에 대한 신뢰를 부여한다. 이렇게 만들어진 기획의 데이터중 게임 요소 분배 트리를 통해 기획자는 해당 게임이 어떻게 유저에게 재미를 주려 했는가, 그리고 실제로 어떤 항목이 그렇게 했는가를 확인해 볼 수 있다.

프로그래머들은 공정에 필요한 기술을 확인하는 과정에서 비교적 정확하게 일정을 산출해 볼 수 있다. 비슷한 장르의 같은 규모의 게임을 제작 한다면, 그리고 가급적이면 실제 그 게임을 제작하는데 몇 명이 얼마나 걸리고 어떤 식으로 작업이 되었는가에 대한 자료까지 확보해서 역공학 과정에 적용시켰다면 무척 정확한 보조 자료가 될 것이다.

바. 게임개발 공정의 Knowlegde Base로서 ODT

하나의 게임 프로젝트가 완료되었거나 심지어는 중도 포기라도 제작 공정 중 파트별로 사용된 인력과 시간들을 산출할 수 있다. 이런 공정상의 경험은 비슷하거나 더 작은 규모의 프로젝트를 이끌어 나가고 위험요소를 미리 파악하며 일정을 예측하는데 큰 자산이 된다. 그러나 대부분 게임회사에서 기(旣) 진행된 제작 공정의 결과를 구체화(具體化)하지 않는다. 이런 경우 공정상의 중요한 예측 기준이 될 수 있는 데이터는 개발에 참여했던 사람들과 특히 일정 관련 매니지먼트(Management)를 담당했던 사람에게 부분별로 막연하게 기억되어 있다. 이것은 게임 제작을 다른 산업이나 어플리케이션(Application)과 같은 공정을 통한 산업적 시각으로 봤을 때 대단한 문제가 아닐 수 없다.

개발 단계 초기부터 ODT를 통해 제작을 한다는 것은 게임의 내용, 장르, 플랫폼에 따른 규모 및 기간과 인력의 변화 요인들을 제작 기간 전 과정에 걸쳐서 라이브러리화(化)한다는 의미이다. 이는 즉 공정 결과가 자연스럽게 구체화가 되며, 그것이 어느 한 사람의 경험단계에서 멈추는 것이 아니라 표준화된 프로세스 형태로 모두가 투명하게 보고 판단할 수 있는 키 데이터가 된다. 이는 어느 한 게임 회사에게는 다음 프로젝트를 진행하는 데 있어 예측 라이브러리의 역할을 충실히 수행하며, 이런 표준화된 제작 공정의 데이터가 업계 전반에 쌓였을 때 ODT는 해당 국가의 게임 제작이 진정한 산업화로 가는 매개체가 될 수 있다.

3) 도구 개발 외적 지원(Support)

가. 개발 도중의 의견 수렴

개발 도중에 공개 포럼을 통해서 GDL에서 사용할 용어와 다이어그램들에 대해 최대한 보편성을 확보하고 의견을 수렴할 필요가 있다.

나. 공개 공간을 통한 지원과 관리

게임 개발은 기술의 변화의 첨단을 달리는 분야이다. 새로운 플랫폼과 주요 기술들이 등장하고 게임에 적용할 수 있게 될 때마다 그에 필요한 공정의 일반적인 절차와 필요 항목들을 템플릿과 ODT가 알고 있어야 도움이 된다. 또한 게임의 디자인적 측면에서도 RTS나 FPS와 같이 새로운 장르가 나타나서 대세를 이루는 경우에 그 장르에 대한 기본적인 디자인 항목과 그에 따른 기술 항목을 인지하고 있어야 한다. 어떤 한 회사나 단체에서 이런 작업을 직접 다 하는 것보다 자발적으로 참여할 수 있는 공간을 만들어 주고 관리와 통일성을 보장해 나가는 것이 적절하다고 판단되며 따라서 개발 후 사이트(www.ODT.org)의 공간을 통해 지속적으로 새로운 용어의 확정, 템플리트의 수정과 템플리트의 타당성을 획득한다.

다. ODT 표준 공정 결과 데이터의 공유화

마찬가지로 게임 개발 공정의 결과에 따른 ODT 프로젝트 데이터를 그 회사 내에서만 활용하는 것 보다는 업계 차원에서 게임 개발 공정의 Knowlegde Base를 구축하는 것이 ODT가 가진 기능들을 통해 국내 나아가서는 전 세계의 게임 개발 필드의 생산성을 극대화할 수 있다.