2019-2 소프트웨어공학개론 DWS 프로젝트 소감 및 프로젝트 리뷰

Team #1

201711376 컴퓨터공학 김경진

201810568 컴퓨터공학 박용준

201811263 컴퓨터공학 백종원

[김경진]

개인 소감:

실제 구현 과정에서 설계 당시에는 예상치 못했던 문제들이 종종 발생해 실제 설계 내용에서 수정 사항들이 여럿 생겼는데, 다음에는 설계를 할 때 구현 시 문제가 될 점들을 생각해 SA, SD를 하면 더 좋을 것 같다. 그리고 팀원들이 너무 똑똑해서 회의 할 때 대화의 흐름을 따라가기 바빠 적극적으로 아이디어를 많이 내지 못한 점이 아쉽다. 다음 프로젝트 시에는 실력을 많이 쌓아 팀에 도움이 되는 조원이 되어야겠다고 다짐했다.

프로젝트 리뷰:

시계가 잘 돌아가서 정말 놀랍고 기쁘다. 이렇게 설계를 자세히 했던 프로젝트는 처음이었는데, 처음에 DFD를 그리고 SRA를 작성하는 과정이 어려웠지만 설계를 꼼꼼히 잘 해 놓으니 구현할 때 이전 프로젝트들에서보다 수월했던 것 같다. OOAD에 대해서도 배워보고 싶다.

[박용준]

DWS(Digital Watch System) 팀 프로젝트에서 저희 팀은 SASD를 이용하여 디지털 시계를 구현하는 팀 프로젝트입니다. DWS의 SRS(System Requirement Specification)를 분석한 뒤 DFD, STD를 포함한 SRA(System Requirement Analysis)와 Structured Chart를 포함한 SDS(Software Design Specification)를 각각 완성하고, 이 두 문서를 기반으로 DWS를 C언어로 구현하였습니다.

SRS를 분석하고 DFD를 그리는 과정에서 처음에는 많은 어려움을 겪었습니다. DFD 자체를 이해하고, 그것을 활용하여 결과물을 내기까지 많은 시행 착오를 겪으며 부딪혔습니다. DFD단계에서의 실수나 부정확한 명세가 이후 단계에서 얼마나 많은 혼란을 가져다 줄 지 예상할 수 있었기에, 여러 번 검토하며 신중하게 DFD를 작성했습니다. 물론 이렇게 작성한 DFD 역시 다수의 부정확한 명세를 포함하고 있었고, 특히 Data Store의 경우, 구현 단계에서 추가되는 경우가 여러 번 생겼습니다.

개인적으로는 이러한 Top-Down approach를 적용하여 팀 프로젝트를 한 것이 처음은 아니었습니다. 지난 학기 MVC 모델을 이용한 게임을 만드는 프로젝트를 하였고, 당시 조장의 지도에 따라

성공적으로 프로젝트를 마칠 수 있었습니다. 다만, 저는 그 때 컴퓨터공학과로 전과한 직후였고, 그때 겪었던 개발 방법들에 대한 인식이 부족했었습니다. 하지만 이번 팀 프로젝트를 진행하면서 직접 SASD와 같은 개발 방법에 따라 프로젝트를 설계해볼 수 있었고, 지난 학기에는 미처 겪지 못했었던 시행 착오와 실패들을 겪으면서 SASD가 굉장히 섬세하고 까다로운 방법이면서도 중요하다는 것을 깨달았습니다. 평소 혼자 프로그램을 작성하듯이 SA없이 코딩에 돌입했다면 DWS를 제대로 완성하지 못했을 것입니다. 구현에 앞서서 명확한 분석을 마치고 그것을 표현할 수 있어야 한다는 것은 어찌 보면 자명하기도 합니다. 그래서 저는 항상 더욱 더 명확하게 이해하고 분석하고자 노력할 것입니다.

이번 프로젝트를 진행하면서 또한 많은 자신감을 얻었습니다. 이번 프로젝트를 진행하기 전에는 SA과정이 중요하다는 것을 인지하고는 있었지만, 분석 방법이나 경험이 빈약했기에 빈약한 명세를 작성하고 불안함을 느끼기 일쑤였습니다. 하지만 이번 프로젝트는 기존과 달리 구체적인 표현 목표(DFD 등)의 체계를 잡고 SA를 진행하였기에, 지난 프로젝트보다 훨씬 더 명확하고 안정적으로 SA를 마칠 수 있었습니다. 또한, 분석만으로 끝나지 않고, 직접 프로그램을 구현함으로써, 우리의 SA의 유효함도 점검해볼 수 있었습니다.

구현에 있어서도 많은 깨달음을 얻을 수 있었습니다. 앞서 언급했듯 저는 지난 학기에 전과하였고, 주로 JAVA를 사용하였기에, C언어로는 간단한 알고리즘 정도만 작성해 보았었습니다. 따라서 이러한 규모의 팀 프로젝트를 C언어로 진행한 것이 처음이었습니다. 물론 구현 과정에서 제 비중이 적지는 않았습니다. 저는 시계에 저장된 현재 시각이 흐르는 것이 절대적인 시간의 흐름과 같은 속도로 흐르도록 구현하였습니다. 즉, 연산 과정에서 생길 수 있는 시간의 흐름에 대한 오차를 보정하였습니다. 또한 stopwatch가 millisecond단위까지 정확하게 흐를 수 있도록, 보정된 현재 시간을 이용하여 구현하였습니다. 또한 현재 시간을 기반으로 stopwatch가 진행되기에, stopwatch측정 도중에 시간이 변경되는 경우 또한 오류가 생기지 않도록 하였습니다. 그리고 화면에 표시될 값들을 가공하고 구성하고 올바르게 표시되도록 구현하였습니다. 그리고 Alarm이 다양한 조건에 따라서도 정확하게 울리고, beep을 재생하느라 프로그램이 멈추지 않도록 구현하였습니다. 또한 프로그램이 동시에 입력 받았을 경우 충분히 빠르게 우선 순위에 따라 대응할 수 있도록 구현하였습니다. 이외에도 팀원과 협력하여 아이디어를 구상하고 코드를 교차 검증하며 순조롭게 구현을 마칠 수 있었습니다.

또한 개인적인 감상으로, 이번 프로젝트가 진행해본 프로젝트 중 가장 규모가 큰 프로젝트였고, 구현 과정을 무사히 마치고서는 정말 행복했다. 많은 실패와 좌절을 겪기도 했지만, 배우고 느낀 점이 많은 팀 프로젝트였다.

요약:

Top-Down approach를 직접 설계하고 진행하는 것의 섬세함과 까다로움을 느꼈다. 하지만 SASD 없이 프로젝트를 진행했다면, DWS를 제대로 완성하지 못했을 것. 전과를 한 탓에 이러한 규모의

C언어를 이용한 프로젝트가 처음이었는데, 순조롭게 구현 과정에 적극적으로 참여할 수 있어서 다행이었다.

[백종원]

작년에 학교를 들어와서 이제 2학년이 곧 끝나기 때문에 학교를 오래 다녔다고는 절대 말할 수 없다. 그렇지만 이번 프로젝트는 상당히 많은 노력을 들였다고 생각한다. 여태껏 들었던 얼마 안 되는 수업들 중에서 가장 바빴던 수업이라고 느낄 정도로 말이다. 하지만 그렇게 노력한 것이 헛된 삽질이 아니라 얻어가는 것이 많아서 다행이라고 생각한다.

학기 초반에는 SA/SD 의 개념을 계속 제대로 이해하지 못해서 DFD라던가 STD라던가 그림이 무엇을 의미하고 어떻게 그려야 하는지 잘 알지 못했다. 분명 수업을 듣고 과제를 했는데도 무언가 애매한 것이 있었다. 그러다가 팀프로젝트가 시작이 되면서 DFD를 계속 수정하게 되었는데, DWS에 대한 SA를 하면서 어떻게 그려야 하고 어떤 것을 의미하는지 파악할 수 있었다. 역시 백문이 불여일견이다.

팀 운은 좋았다. 본래 어느 정도 아는 사람들인 것도 있지만, 아는 사람들끼리 팀을 짰을 때는 보통 잘 되거나 혹은 완전 망해서 서로 싸운다. 그런데 이번 팀은 정말 괜찮았다. 처음에 가장 실력이 좋은 분 한명이 휴학을 해서 걱정을 많이 했었는데, 3명으로도 문제 없이 충분히 잘 했다고 생각한다. 모여서 회의를 여러 번 했었는데, 회의 때마다 의견 갈등이 심하거나, 대충 하거나 하는 일이 없어서 큰 문제 없이 잘 진행했던 것 같다. 오히려 내가 계속 이해를 못하고 못 따라갔다는 느낌이 좀 있을 정도로 다들 잘했다.

설계와 코드 공유를 나름 제대로 진행한 것은 매우 좋았다. 지난 학기에 들었던 수업에서는 설계도 제대로 안되고, 따라서 성급한 구현과 함께 뒤집어지는 일이 여럿 있었다. 동시에 협업에 있어서도 뒤죽박죽이었다. 각자 맡은 부분을 짠다고 해놓고 코드를 카톡으로 공유해서 정말 혼란스러운 상황이 많았다. 하지만 이번에는 설계도 촘촘히 해볼 수 있었고, 그래서 분업하기도 편했다. 동시에 분업한 내용들에 대해서 코드 공유 및 버전 관리를 깃허브를 사용하면서 말끔하게 처리했다. 개인 프로젝트에서만 깃허브를 써봤었는데, 왜 기업들이 깃허브 사용자를 우대하는지 알 것 같았다. 프로그램의 변경 사항을 업데이트하기 쉽고, 각자 분업하여 짜기로 한 부분들을 합치는게 정말 편했다. 앞으로의 프로젝트에서는 꼭 이런 식으로 코드 공유 및 버전 관리를 할 수 있도록 해야겠다.

구현 단계에서 가장 기억에 남는 부분은 realtime manager() 이다. 우리는 이를 실제 시간의 흐름을 표현하기 위해 구현했었는데, 1ms 안에 처리하도록 하는 것이 정말 어려웠던 것 같다. 그리고 거기에서 시간 계산을 해준 다음 시간의 범위 체크를 해주는 것을 어떻게 해야 할지 처음엔 정말 하나도 생각이 나지 않았었다. 그만큼 빠르게 시간이란 것을 처리하는 것이 어려웠다고 생각한다.

다행히 time.h의 기능들을 활용해서 해결했다. 진짜 왠만한 것들은 다 라이브러리에 있다. 그리고 그만큼 이걸 처음부터 만들기 위해 노력한 개발자 분들이 있었을 것이다. 정말 존경스럽다.

<요약>

많은 시간과 노력을 투자했고 나쁘지 않은 결과를 얻었다고 생각한다. 팀원분들이 괜찮았기에 내가 운이 좋았다고 생각한다. 설계와 코드 공유를 문제 없이 해봐서 스트레스 받는 일이 없었고, 구현에서도 에러들이 여러 번 발생했었는데, 이를 빠르게 빠르게 해결해나가서 좋았다.