스터디를 진행하기에 앞서

1. LucidCore의 구성원의 역량을 강화한다.
   1. 금년 국내 경제사정이 나빠진다고 합니다. 제기억에 우리나라 경제 사정이 좋았던 적은 없습니다만, 뭔가 대비한다는 느낌만으로도 기분이 좋아지지 않을까 하는 기대감...
2. 미래에셋의 의존도를 줄인다.
   1. 대다수의 기업의 IT관련 부분을 가보면 거의 대부분의 인력이 외주인력입니다. 언제든지 회사 사정이 안좋으면 인력을 감축시키기 쉽도록 구성해 놓는 것입니다.
      1. 미래에셋도 마찬가지로 외주인력을 많이 사용하고 있습니다.
      2. 실제상황을 보면 외주인력이 많을 수록 쉽사리 외주인력을 감축시키기가 곤란합니다. 내부인력이 적으면 내부인력은 외주인원의 관리만 해도 시간이 부족하기 때문에 실제 업무 수행역량이 없어지게 됩니다. (혜화동 KT화재같은 상황이 됩니다.)
      3. 그래서 미래에셋은 내부인원을 보충하고 상당수의 업무를 내부인원에게 가져가도록 하고 있습니다.
      4. 상황이 어쨌든 무언가 힘들어지면 외주인력을 줄이기 시작할 것이고, 우리도 2024년에는 미래에셋과의 계약이 어떻게 될지 알수 없습니다.
      5. LucidCore에서도 계약처가 좀 더 다변화 될 필요가 있기 때문에 다른 증권사나 금융권의 프로젝트를 적극적으로 수행하려고 하고 있습니다.
      6. 현재 미래에셋에서 우리가 사용하고 있는 플랫폼은 ITGen이라는 회사에서 만든 모바일 플랫폼을 기반으로 미래에셋에서 수정한 것으로 현재 국내 대형증권사 일부만 사용하는 플랫폼입니다. 사용하는 Script도 Lua라는 스크립트로 게임분야 등에서 활발하게 사용되고 있으나 범용적인 스크립트라고는 할 수 없으며 플랫폼도 범용적이라고 할 수 없습니다. 국내에 대형증권사를 대상으로 만든 플랫폼이나 툴은 7개정도 팔고 나면 더이상 팔 곳이 없습니다.
      7. 플랫폼을 만들어두면 좋을 것 같지만, 실제로는 플랫폼이 아니라 기본라이브러리 정도의 형태로 판매하고 관리하기 때문에 플랫폼의 버전관리가 제대로 되지 않습니다. 각 사이트별로 커스터마이징을 하다보니 기본 플랫폼의 버전관리가 잘되지 않습니다. 각 사이트별로 별도의 프로그램을 작성해서 공급한 형태가 되게 되고 관리 비용이 너무 많이들어서 관리를 포기하고 있는 형편입니다.
      8. 앞으로 구성원을 화면 구성이 아니거나 다른 형태의 플랫폼위에 앱 등을 작성하는 프로젝트에 투입될 가능성이 있습니다.
3. 개발자의 입장
   1. IT업계는 지속적인 인력란에 시달리고 있습니다.
      1. 사람들은 구직난이라고 하지만 IT업계는 지속적인 구인난에 시달리고 있습니다.
   2. 경력신입이라는 허수
      1. 그러나 실제 구직하는 사람입장에서는 전혀 실감할수 없는데, 이유는 경력신입이라는 말도안되는 사람을 찾기 때문입니다.
      2. 경력신입은 말도 안되는데 실제로 구인시장에 가보면 흔하게 이를 모집합니다. 그러고 이를 충족하는 사람이 있다고도 합니다. (이는 근래 문제는 아닙니다. 제가 프리랜서 바닥에 들어온 이후 꾸준히 그렇습니다.)
      3. 경력신입은 초급단가를 받는 경력직입니다. (즉 개인이 자신의 등급을 낮추면 가능합니다.)
      4. 또는 부풀려진 경력입니다.
4. Script / 4GL언어에 대한 무시
   1. 프로그램공부는 C로 시작했지만(제대로 했다고 할수는 없습니다.), 회사에서는 VB로 화면작성을 시작했습니다. 당시 Windows3.1에 Network기능이 추가되고 이후 Windows95가 나오면서 Network기능이 기본으로 Windows에 탑재되던 시기 였습니다. Windows상에서 간단하게 다양한 화면을 구성하는데 있어서 VB는 쓸만한 도구였고 외국에서는 매우 적극적으로 사용되던 툴이었습니다.
   2. C와 WIndows API를 이용해서 화면을 만드는 일은 지극히 귀찮습니다. 무언가 간단한 부분도 일일이 지정해 주어야 합니다. 그에 반해서 VB로 만드는 화면은 마우스 클릭 몇번만으로 화면을 만들어주기 때문에 훨씬 간단하고 쉽게 첫화면을 띄울 수 있습니다. 다만 해당 화면에서의 세세한 조정은 WIndows API를 사용하는 것 만큼 지원하지는 않습니다. (VB에서도 Windows API를 이용해서 C와 동일한 기능을 지원하게 하는 기능을 우회적으로 제공합니다.)
   3. 이렇기 때문에 VB프로그래머가 C를 이용하는 프로그래머에게 괄시를 받습니다. 저 또한 그랬습니다. 2000년대 초반에 옮기 회사에서 C++로 프로그래머가 VB는 프로그램언어로 인정할 수 없다고 했습니다. 당시 저는 상황을 알고 있기 때문에 가볍게 무시했습니다. 그러나 상당수의 프로그래머들은 그렇지 못할 수 있습니다. 왜냐하면 양쪽 모두의 언어로 수행해보지 않으면 실제 어떤지 알기가 곤란하기 때문입니다.
   4. 여러 환경과 언어를 다양하게 경험해보는 것은 흔한일은 아니면서도 흔한일일수 있습니다. 저는 제가 원해서 다른 환경과 언어를 택한 적은 별로 없습니다. C와 C++정도만 제가 공부한 언어이고 나머지는 해야만 해서 한 언어입니다.
   5. 적극적으로 원했다기 보다는 할 사람이 없어서 한경우가 대부분이고 제가 굳이 안하겠다고 하지 않았기 때문에 여러 언어를 한 것입니다. 제가 굳이 안하겠다고 하지 않은 것은 C로 화면구성을 하고 업무화면 등을 개발해본 경험 때문입니다.
   6. C와 Windows API로 화면을 구성해 보고 COBOL로도 화면을 구성해 보고 VB로도 화면을 구성하면서 느낀점은 화면 구성에서 모든 언어에 차이가 없다는 것입니다. C는 포인터 등이 매우 까다롭고 어려운 부분입니다. 메모리영역을 잡고 해제하는 부분 같은 것을 직접 해줘야 하기 때문에 무언가 귀찮은 지점이 많습니다. 그러나 실제 업무용 프로그램을 만들게 되면 무언가 귀찮은 부분은 생략이 됩니다. 포인터를 사용하지 않고 배열을 사용합니다. 모든 데이터는 고정길이배열에 구조체에 들어갑니다. 사용하는 명령어도 매우 제한됩니다. strcpy나 memcpy같은 명령어만 반복적으로 사용됩니다. 나중에 가면 이전에 작성한 모듈을 복사해서 붙여넣기로 거의 모든 작업이 이루어지는 반복작업이 주게 되게 됩니다. 책에서 보고 어렵게 느꼈던 부분은 대부분 사용하지 않습니다. 나중에는 복사 붙여넣기가 VB같이 4GL언어보다 더 빠르게 화면을 구성할 수 있게 됩니다. 즉 업무 화면의 개발 부분은 화면 배치와 데이터 체크등이 거의 모든 작업을 하게 되며 각 언어가 가진 특징적인 부분이나 어려운 부분은 거의 사용할 일이 없게됩니다. 그래서 굳이 다른 언어나 환경이라고 해서 처음부터 발을 때지 않고 일단 한번 문제를 검토해 보겠다고 (어차피 문제을 해결해줄 사람이 없기 때문에) 한 것이 여러 언어와 환경을 경험한 토대가 된 것입니다.
   7. 업무구현 때도 보면 Web / Html 개발자(Script)개발자 보다 Native관련 개발자가 더 나은 보수를 받거나 더 나은 개발자로 인식되는 것이 흔하게 발생합니다. 종종 Engine이나 Core개발자가 더 중요하게 취급되고 심지어는 화면개발자는 없어도 프로젝트가 진행되는 것으로 인식하는 경우도 종종보게 됩니다. 그러나 서비스의 실제 핵심적인 부분은 화면 개발자가 더 많이 알고 있는 경우가 많습니다. 이는 실직적인 기능의 차이라고 보다는 화면개발자가 더 많이 필요하기 때문에 더 많이 훈련되었고 최초 허들이 조금 더 낮기 때문에 보이는 현상입니다. 결코 화면 개발자의 코딩 능력이 코어개발자보도 낮다고 볼 수 없습니다.
5. 황당한 프리랜서
   1. 어떤 사이트에서 앱 개발시 iOS프리랜서가 있었습니다. iOS담당 프리랜서였는데 XCode를 열고 빌드할 수 있다는 것 이외에는 아는 것이 거의 없었습니다. 어디선가 iOS Native담당을 했었던 이력이 있다는 것이 전부였고 그 것 때문에 뽑혀서 iOS를 담당하고 있었습니다. 프로젝트는 하이브리드 앱이었는데 웹페이지가 PC를 중심으로 만들어진 페이지를 억지로 모바일에 앉히다보니 근본적인 문제가 있었습니다. 안드로이드 담당은 별도로 있었는데 안드로이드 담당은 자신의 것에서는 문제가 없기 떄문에 관여하지 않고 있었습니다.(제가 조사한 결과 Native개발자 만으로는 해결할수도 없는 문제였습니다.) 앱이 제대로 동작하지 않는데 iOS개발자는 오히려 당당했습니다. 외주로 인원을 채운 프로젝트의 문제점이기도 합니다.
   2. 프로젝트가 시작되면 PM의 입장에서는 인원을 세팅하는 것이 큰 숙제입니다. 현재 대부분의 SM이나 SI계약체계는 머릿수를 세기 때문입니다. 일을 잘하든 못하든 제시한 인원수를 채워서 투입하는 것이 가장 큰 과제입니다. 그래서 실제로 상당히 많은 SI프로젝트의 인원이 허수로 채워지게 됩니다. 그리고 이 문제는 쉽사리 없어지지 않을 것입니다. 위의 황당한 프리랜서 처럼 무언가 해봤다는 것만으로 프로젝트의 인원 채우기에 동원될 수 있습니다. 일없이 자리만 지키는 것도 괴롭기는 하지만 무언가 기술이 있다는 이력만으로 쉽게 자리를 얻을 수 있을 수도 있습니다.(물론 운이 따라야 하고, 정말 실력이 부족하다면 업계에 소문이 안좋게 날 수도 있지만, 좁기도 하고 넓기도 한게 이바닥입니다.)
6. 최신기술을 꼭 해야 하는가?
   1. 최근 기술쪽으로 사람을 만나보면 Swift나 Kotlin관련 이야기를 많이하고 관심들이 많습니다. (AI쪽은 Python도 많이 사용하고 있어서 그쪽도 많이 이야기를 합니다.) 그런데 많은 기술들이 짠하고 나타났다가 소리 소문없이 사라졌습니다. 괜히 헛심썼던 기술도 많이 있습니다.
   2. 아직 금융권에서는 Objective-C를 많이 사용하고 있습니다. Swift로 잘 넘어가지 못합니다. Android도 Java를 사용하고 있고 Kotlin으로 넘어가지 못하고 있습니다. 과연 할 필요성이 있을지 의문이 듭니다.
   3. 모바일 개발에서 언어는 사실상 큰차이가 아닙니다. 모두 if문과 for문으로 거의 모든 것을 처리하고 있습니다. 어려운 부분은 iOS는 UIKit, Android는 컨트롤 Widget, Windows는 Windows API에 있는 함수들과 사용용도이며 이는 관련 레퍼런스를 찾고 사용법을 별도로 익혀야 합니다. iOS를 예로 든다면 UIKit을 알고 있다면 Objective-C로 프로그램을 짜던 Swift로 프로그램을 짜던 별차이가 없습니다.
   4. 그렇다고 돈이 있고 기술이 있는 놈들이 밀기 때문에 마냥 무시하기는 곤란합니다. Android개발이 처음에는 Eclipse에 plugin을 설치하는 것으로 시작했습니다. 지금은 모두 Android-Studio를 사용합니다. 초창기에는 Eclipse plugin이 로딩속도가 컴파일 속도가 더 빨랐고 사용하는데도 큰무리가 없었습니다. Eclipse가 WIndows에 최적화가 더 잘되어 있었기 때문입니다. Android-Studio는 무겁고 속도고 느렸으며 Gradle의 컴파일 속도는 최악이었습니다. 그런데 Google이 Eclipse지원을 중단하고 Android-Studio를 꾸준히 반전시켜서 이제는 Android Studio만으로 개발이 가능합니다.(물론 WIndow-Phone과 같이 큰놈도 말아먹고 튀는 경우가 있습니다만,) Google이 Kotlin을 지속적으로 밀고 있고 Apple이 Swift를 지속적으로 밀고 있기 때문에 관심을 꽤 가져둘 필요가 있고 현재 꽤 오래 성공적으로 밀고 있기 때문에 더더욱 관심을 가질 필요가 있습니다.(개인적으로는 Google이 GO를 왜 밀지 않았는지는 의문입니다.)
7. 결론
   1. Native기술은 별게 아닙니다. StackOverflow와 넘치는 github예제로 극복할 수 있습니다. 문제는 투여하는 시간입니다. Script기술자가 무언가 전혀 다른 것을 배워야 할 수 있는 것은 아닙니다.
   2. 어디선가 들어는 봤다는 기분을 가지는 것도 나쁘지 않습니다.
   3. 안해보고 익숙하지 않다는 것은 고생길인 것은 분명합니다. 하지만 프리랜서시장에서는 자세한 검증이 이루어지지 않습니다. 그리고 실제로는 요구하는 기술 역량이 높지 않은 경우도 많습니다. 안해본 것 같아도 익숙한 것인 경우가 많으므로 너무 쫄지 맙시다.