Out of the tar pit summary

20150560 Dongwon Kim

01. Intro

- “the single major difficulty in the successful development of large-scale software systems”
- State 관리, code volume, flow of control – complexity main cause
- 이를 해결하기 위해서는 behavior 와 state를 긴밀하게 연관시키거나/functional programming
- Relational DBMS 에서의 아이디어가 이를 해결할 수 있다?

02. complexity

- 모든 sw problem 의 root cause.
- “there is a desperate need for a powerful methodology to help us think about programs. … conventional languages create unnecessary confusion in the way we think about programs”
- Complexity of maintaining/creating system vs resource/time consume of computation

03. Approaches to understanding

- Two most popular ways – informal reasoning/ testing
- 이 둘은 필연적으로 많이 사용하게 되는 사고방식
- 하지만, bug에 대한 가장 좋은 대처법은 bug의 원인을 원천적으로 피하는 것.
- 처음부터 simple 하게 짜는 것이 testing을 반복하는 것보다 낫다. – testing 은 에러가 있음을 보이기에는 좋지만, 무결함을 증명하는데 사용할 수는 없다.

04. cause of complexity

- State : complexity는 가능한 state의 수와 비례한다,
  - Testing과 informal reasoning 둘 다 힘들게 만든다.
- Control : 발생하는 사건들에 대한 순서
  - Text의 order 와는 상관이 없다. Context 를 고려해야 이해할 수 있다.
- Code volume : state와 control 문제로 인해 발생.
  - Concurrent program 에서는 nonlinear 하게 volume 증가.
- Complexity로 인한 complexity ex) duplication
- Simplicity는 어렵다.
- Power corrupt : 아이러니하게도, 가장 많은 것이 가능한 언어로 짜여진 system일수록 이해하기 어렵다.

05. Classical approaches to understand complexity

- OOP
  - Encapsulation. State를 behavior 와 tight하게 엮는..
  - Traditional control flow를 따른다.
  - Intentional identity와 extensional identity가 있기 때문에, 이 둘이 state 처럼 작용하며 complexity를 증폭시킨다.
- Functional programming
  - Not traditional von nueman model, stateless lambda architecture
  - Stateless 하지만, parameter 가 state 처럼 작동할 수 있다.
  - Labda architecure의 문제점은 바로 stateless 하다는 점!. 어떠한 system들은 state를 태생적으로 가진다.
- Logic programming
  - Axiom을 기반으로 solution을 증명한다.
  - 하지만 control logic을 많이 요구한다.

06. Accidents and Essence

- Essential Complexity – problem 자체에 내재되어 있는 / Accidental Complexity – 개발 과정에서 생기는 여러 이슈로 인한 complexity.
- SW system 은 이 두 complexity 가 혼재되어 있음,
- 이 중 accidental complexity 를 최대한 줄이는 것이 좋다.

07. Recommended general approach

- 어떤 것이 essential 이고 accidental 한지 알아야 어떻게 complexity를 해결할지 알 수 있다.
- Ideal world 에서의 problem solving을 생각해보면 된다.
- Ideal condition 이기 때문에 formalize 과정만 거친다고 생각한다. – 즉 어떻게보다 무엇을 을 작성하는 pure declarative style.
- Ideal world 에서도 essential state 존재.
- Ideal world 에서 모든 control 은 essential. 이를 최대한 피하는 것이 중요하다.

08. The Relational Model

- 모든 data structure 를 표현하기 위해서 ‘relation’을 사용한다.

09. Functional Relational Programming