Homework Essay

4190.310 Programming Language

2013-11422

이은하

1. 요약

Programming Language는 C, Java 등의 명령형 언어와 Haskell, OCaml 등의 함수형 언어로 나뉜다. 명령형 언어의 대표격인 C언어가 개발되는 과정을 알아보자. 반면 함수형 언어를 보면 수학적이고 논리적인 흐름에 입각하여 만들어져 있다. 함수형 언어는 이와 같이 프로그래머를 편하게 해주는 인간 중심형 언어이다. 과거와 달리 현재에는 컴퓨터의 메모리가 여유롭고 코어가 여러개인 경우도 흔하다. 이에 따라 적합한 언어가 달라지는데, 이쯤에서 최근 Facebook의 스팸 거름망 프로그램을 만드는데 쓰인 Haskell에 대하여 알아보자. 이와 같이 시대의 흐름에서 함수형 언어가 명령형 언어보다 유리한 고지를 차지하고 있다고 하더라도, 시대는 얼마든지 바뀔 수 있으므로 지속적으로 꾸준히 새로운 언어를 개발하고 사용해야 할 것이다.

2. 본문

A. Programing Language의 종류

Programming Language는 C, Java 등의 imperative language(명령형 언어)와 Haskell, OCaml 등의 functional language(함수형 언어)로 나뉜다. 각 언어의 특성을 알아보기 위해 개발 과정을 살펴보자.

i. Imperative Language

Imperative language의 대표격인 C언어가 개발되는 과정을 알아보자. Dennis M. Ritchie의 "The Development of the C language"를 보면, C언어는 BCPL과 B언어를 거쳐 C언어로 오면서 Unix 운영체제의 진화와 함께해왔다. 때문에 기계 중심적인 언어로 성장할 수밖에 없었고, 필연적으로 인간과는 멀어지게 되어 프로그래머에게 여러모로 고통을 안겨주게 되었다. 또한 C언어에 B언어에 없었던

type을 추가했지만 프로그래머들 모두가 알다시피 그 type들은 그렇게 엄격하지 않다. 이것저것 추가되면서 덕지덕지 붙은 지저분한 문법은 직관적이지 않은 언어 구성과 함께 다루는 사람의 머리를 아프게 한다.

ii. Functional Language

반면 Functional language들을 보면, 수학적이고 논리적인 흐름에 입각하여 만들어져있다. Philip Wadler의 "Proofs are Programs: 19th Century Logic and 21st Century Computing"을 보면 함수형 언어의 근간을 이루는 Alonzo Church의 lambda calculus에 대해 알 수 있다. 1932년에 자신이 발표한 lambda calculus에 더해, Alonzo Church는 1940년 type 개념을 추가한 lambda calculus를 소개했다. 물샐 틈 없는 타입 추론 알고리즘 덕분에, 요새의 함수형 언어들은 컴파일 단계에서 함수의 타입이 맞지 않으면 가차없이 에러를 내뱉는다. 타입 에러들을 고치지 않으면 프로그램은 돌아가지 않고, 프로그램이 돌아간다는 것은 프로그래머가 의도하지 않은 무언가가 프로그램에서 일어나지 않는다는 말이 되었다.

함수형 언어는 이와 같이 프로그래머가 편하게 해주는 인간 중심형 언어이다. high-level programming language에서는 그러지 않지만 C같은 low-level에서는 메모리도 일일이 할당해주고 배열도 일일이 만들어주어야 하며 그 크기를 초과하면 아주 비참한 결과가 일어나는 등 힘든 일이 많은데, 이런 머리아픈 과정 없이 타입만 맞추면 깔끔하게 프로그램을 돌려주는 함수형 언어는 그야말로 '인간 중심'이라고 말하는게 적절한 언어인 것이다. 함수형 언어를 씀으로써 얻는 이득은 이뿐만이 아니다.

B. Facebook이 선택한 Haskell

과거와 달리 현재에는 컴퓨터의 메모리가 여유롭고 코어가 여러개인 경우도 흔하다. 이에 따라 적합한 언어가 달라지는데, 이쯤에서 최근 Facebook의 스팸 거름망 프로그램을 만드는데 쓰인 Haskell에 대하여 알아보자. Haskell은 순수한 함수형 언어로 정확성, 동시성 등에 강점을 보이는 언어이다. "Facebook's new spam-killer hints at the future of coding"이라는 글을 보면 필자와 개발자 Brandy와의 대화에서, Brandy는 Haskell이 동시에 서로 다른 많은 일들을 처리하는 데에 능하기 때문에 Spam filtering에 이상적이어서 개발 언어로 채택했다고 말하고 있다. 이와 같이 무작정 사람들이 많이 쓰는 imperative language를 쓸 것이 아니라 하는 프로젝트의 특성에 따라 적당한 언어를 골라 써야 하는데, 요즘의 추세는

큰 규모의 자료를 다루며 프로그램들을 코어 여러개에서 돌리는 것이므로 Haskell을 쓰는 것이 적합할 경우가 많게 되었다.

C. 미래의 언어

이와 같이 시대의 흐름에서 함수형 언어가 명령형 언어보다 유리한 고지를 차지하고 있다고 하더라도, 시대는 얼마든지 바뀔 수 있으므로 지속적으로 꾸준히 새로운 언어를 개발하고 사용해야 할 것이다. 언젠가는 함수형 언어도 역사의 뒤안길로 사라질 날이 오게 될 지도 모른다. 과거에 쓰이던 C언어가 특정 분야를 제외하곤 high-level language에 밀려 찬밥 신세가 되었듯이, 새로운 시대에 맞는 새로운 언어가 또 개발되어 널리 쓰이게 될 것이고, 그렇게 되어야 한다. 새로운 언어의 개발과 그에 따른 학문적 연구를 생각해 볼 때, 항상 공부를 놓을 수 없는 것이 프로그래머의 숙명인 듯 하다.