

# SPLASH/OOPSLA 2011 여행기

Portland, Oregon, USA, October 22-27

임현승 (genilhs@postech.ac.kr)

지난 10 월 논문 발표를 위해 SPLASH 2011 에 다녀왔습니다. SPLASH 는 POPL, PLDI, ICFP 와 더불어 프로그래밍 언어 분야의 최고 학회 중 하나인 OOPSLA 를 객체지향언어 중심에서 프로그래밍 언어 전문야로 확장시키려는 노력의 일환으로 작년에 탄생하였습니다. 올해의 SPLASH 는 OOPSLA 를 비롯하여 Onward, Wavefront, PLoP, GPCE 등 굵직한 학회 이외에도 Scheme, Hackathon, Doctorial Symposium, Educators & Trainers Symposium, Dynamic Languages Symposium, Demos, Posters, ACM Student Research Competition 등의 프로그램, 그리고 FOOL 을 비롯한 10 개의 워크숍 등으로 구성된 대형 이벤트였습니다. SPLASH 2011 에 대해 총평을 하자면 전반적으로 상당히 활발하고 친근한 분위기였으며 몇 개의 프로그램을 제외하고는 아무 프로그램이나 듣고 싶은 프로그램에 마음대로 참석해서 들을 수 있다는 점이 특히 좋았습니다. (이에 반해 ICFP 와 함께 개최되는 프로그램의 경우 등록된 프로그램에만 참석할 수 있었습니다.)

## 1. 가자 포틀랜드로



그림 1 카이스트 친구들과 함께 지낸 숙소 Hotel Fifty 전경

우리나라에서 포틀랜드까지 바로 가는 직항 비행기는 없습니다. 그래서 시애틀을 거쳐 포틀랜드로 가는 여행 일정을 잡았습니다. 여정 계획 자체는 꽤나 괜찮았으나 포틀랜드까지 가는 길은 쉽지 않은 않았습니다. 시애틀로 오는 10 시간 남짓한 시간 동안 한 숨도 못 자고 깨어 있었으며, 입국 심사는 기다리는 줄은 짧았으나 앞에 계신 대다수의 분들이 시간을 질질 끌어서 1 시간 10 분이나 걸렸기 때문입니다. 또, 국제공항에서 국내공항으로 넘어가는 길에는 random inspection 을 당했으며 포틀랜드행 비행기를 타기 위해 보안검색대를 통과할 때는 전신스캔을 당했습니다. 나중에 알고 보니 시애틀 국내공항에 보안검색을 하는 곳이 두 군데가 있는데 제가 통과한 곳은 일반검사기는 없고 전신스캐너 밖에 없어서 전신스캐너를 사용할 수 밖에 없는 구조였습니다. 한국으로 돌아올 때는 다른 곳을 이용해서 그냥 일반검사기만 통과할 수 있었습니다. 포틀랜드 공항에 도착해서는 KAIST 류석영 교수님과 박창희군, 이흥기군, 한명희군과 합류해서 Max Light Rail 전철을 타고 숙소까지 이동하였습니다. 도착한 날이 토요일 저녁이어서 그런지 포틀랜드 시내는 꽤나 한산했습니다.

## 2. SPLASH/OOPSLA 2011 총평



그림 2 (좌) OOPSLA 논문 발표장 (우) 논문 발표

OOPSLA 는 확실히 변하고 있습니다. 지난 학회와 비교해서 특기할 만한 점은 우선 첫째로 accept 된 논문의 수입입니다. 올해에는 총 165 편의 논문이 제출되었는데 그 중에서 61 편의 논문이 accept 되었습니다. 제출된 논문의 37% 정도가 accept 된 것입니다. 작년에 45 편의 논문이 accept 된 점을 감안하면 수치상으로 크게 늘었다고 볼 수 있습니다. (작년에도 사실상 그 전보다 훨씬 많은 논문이 accept 된 것이었습니다.) 단지 낮은 accept ratio 를 유지해야 한다거나 또는 발표 여건이 안 된다는 이유로 좋은 논문을 reject 시키지 말자는 것이 전반적인 분위기였으며, 프로그램 커미티 멤버 모임에서 실제로 논문의 accept 여부를 결정할 때 논문을 랜덤하게 고려함으로써 좋은 논문을 최대한 accept 시킬 수 있었다고 합니다.

또 한 가지 특기할 점은 예전에는 없던 타입 시스템 (Type Systems) 세션이 한 개도 아닌 두 개나 생긴 것입니다. 타입이론을 연구하는 한 사람으로서 기분 좋은 일이 아닐 수가 없습니다. 이번에 발표하게 된 논문도 타입이론에 관련된 것으로, 이미 ROSAEC 센터 워크숍에서 여러 번 소개한 적이 있는 ML 언어에서 재귀 모듈을 지원하기 위한 타입 시스템에 대한 것이었습니다. 사실 논문 공저자인 Jacques Garrigue 교수님이 올해 ICFP 프로그램 커미티 멤버여서 논문을 ICFP 2011 에 제출할 수 없다는 이야기를 들었을 때는 다소 좌절했었습니다. 그런데 OOPSLA 가 타입이론까지 그 영역을 확장해서 우리 논문을 제출할 수 있었고 또 논문이 accept 되어서 결과적으로는 정말 다행이라고 생각합니다.

## 3. Parallelism and Concurrency

최근 프로그래밍 언어 연구의 경향은 역시 parallelism 과 concurrency 가 아닌가 싶습니다. OOPSLA 에서는 총 17 개의 세션 중에서 4 개의 세션이 parallelism 과 concurrency 관련 논문에 할당되었으며 OOPSLA 2011 Distinguished paper award 를 수상한 총 세 편의 논문 중 한편이 parallel computation 에 관련된 논문이었습니다. Educators' and Trainers' Symposium 에서는 parallelism 을 어떻게 하면 대학교 커리큘럼에 잘 추가해서 학생들을 잘 가르칠 수 있을까에 대해 깊이 있게 다뤘으며 TMC 워크숍에서는 (Workshop on Transitioning to Multicore) parallel programming 에 대한 다양한 발표가 있었습니다. 이 밖에도 SPLASH 2011 기간 내내 여기저기서 parallel programming 에 대한 이야기를 들을 수 있었습니다.

저 역시 parallel programming 에 대한 지대한 관심으로 Educators' and Trainers' Symposium 에 참석했는데, 참석하고 보니 잘못 참석했다는 생각이 들었습니다. 사람이 적어서 그런지 모든 사람들에게 자기소개를

부탁했는데 적잖이 당황스러웠습니다. 청중 대부분이 교수거나 기업체 교육 담당자였기 때문입니다. 학생은 딸랑 총 3 명이었습니다.

### Educators' and Trainers' Symposium

#### Dick Brown – Surviving in the Wild: Teaching and Training for the Parallel Future

Parallelism 이 필요한 가장 큰 이유는 scale 때문입니다. 달리 말해 single processor 의 성능 향상은 더 이상 기대할 수 없는데 해결해야 되는 문제의 규모는 점점 커지고 복잡해지고 있기 때문입니다. 앞으로 CS 를 전공하는 학생이 parallel programming 을 모르고서는 취업하기 힘든 시대가 조만간 오지 않을까 조심스럽게 짐쳐봅니다. 이런 상황 속에서 CS 교육과정에 parallel programming 을 추가하려는 노력이 많이 있는데, 학생들이 parallel programming 에 대해 빨리 접할수록 더욱 효과적이라고 합니다. 생각해보면 당연한데, 학생들이 sequential program 에 갇히기 전에, 즉 학생들의 사고 방식이 sequential programming 패러다임에 고착화되지 않았을 때, parallel programming 을 더 빨리 습득할 수 있기 때문입니다. 발표의 결론은 현실적으로 parallel programming 에 대한 새로운 과목을 개설하기 어렵다면 다양한 교과목에서 2~3 시간씩 할당해 parallel programming 에 대해 소개하는 것이 좋다는 것이었습니다.



그림 3 (좌) 카이스트 박창희군과 이흥기군 (우) 박창희군과 함께

### OOPSLA Distinguished Paper Award

#### Sebastian Burckhardt et al – Two for the Price of One: A Model for Parallel and Incremental Computation

논문 저자 중 한 명이 학부 선배여서 더욱 관심이 갔던 발표였습니다. 이번 OOPSLA 는 발표논문이 많아서 각 논문당 20 분의 발표시간과 2 분의 질문답변시간만이 주어졌습니다. 짧은 시간 동안의 소개였지만 상당히 흥미로운 연구였습니다. 대부분의 reactive 또는 interactive 프로그램은 아래에서 보듯이 compute()와 mutate()의 반복으로 구성됩니다.

```
while (true) {  
    compute();  
    mutate()  
}
```

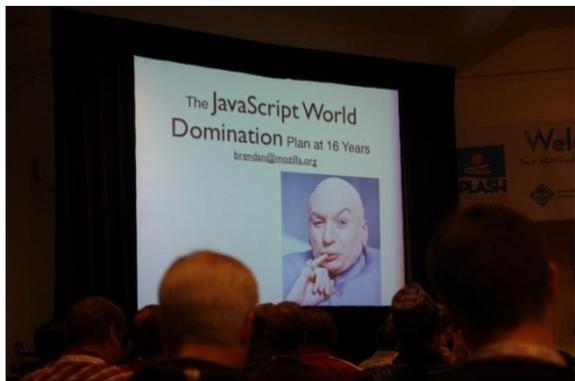
compute()는 주로 deterministic 계산을 수행하며 I/O 가 없고 병렬화가 가능한 경우가 많습니다. 반면에 mutate()는 nondeterministic 계산을 수행하며 I/O 를 동반하는 경우가 많습니다. 예를 들어 웹 브라우저에서 사용자는 페이지의 일부분을 수정하도록 요청할 수 있는데 이는 mutate()에 의해 처리됩니다. 이에 따라 페이지 레이아웃을 다시 그리는 것은 compute()가 처리합니다. mutate()에 의해 변경되는 부분이 작으면

작을수록 compute()는 불필요한 연산을 더욱 많이 수행하는데, 이러한 불필요한 연산을 줄이기 위한 방법이 많이 연구되고 있습니다. Self-adjusting computation 은 이를 위한 한 방법으로 대략적으로 말해 mutate()에 의해서 영향을 받지 않는 부분은 기존 결과를 그대로 이용하고 영향을 받는 부분만 다시 계산하는 방법론입니다. 이 논문에서는 몇 개의 primitives 를 이용하여 self-adjusting computation 을 지원하면서 동시에 compute() 부분은 parallelize 할 수 있는 방법을 제시합니다. 작년 OOPSLA 에서 발표된 deterministic parallel programming model 인 current revisions 을 확장하여 self-adjusting computation 을 지원하는 모델을 제안하였는데 실험 결과가 상당히 고무적이었습니다.



그림 4 포틀랜드에서 유명하다는 Voodoo 도넛; 12 개들이 한 박스를 주문했는데 신기한 도넛이 몇 개 들어있었습니다. 카이스트 학생들과 함께 먹었지만 결국 5 개의 도넛이 남았습니다.

### 3. JavaScript VS. Dart web programming language



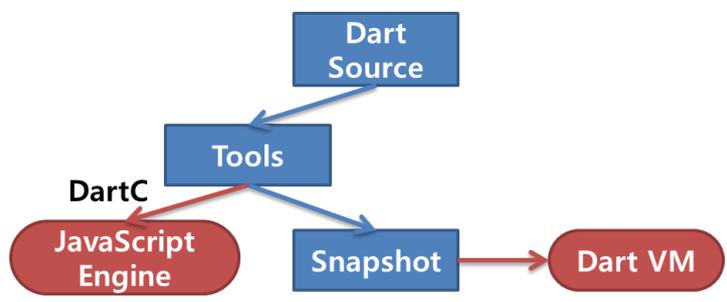
자바스크립트가 탄생한지도 벌써 16 년이 되었습니다. 저는 자바스크립트를 제대로 배워보거나 사용해본 적이 없지만 자바스크립트가 웹 프로그래밍 영역에서 매우 많이 사용된다는 것은 익히 들어 잘 알고 있었습니다. 일례로 지난 10 월에 발표된 TIOBE Programming Community Index 에서 자바스크립트는 10 위를 차지하는 위상을 보여주었습니다. (제가 좋아하는 ML 은 안타깝게도 42 위에 머물렀습니다. Haskell 의 경우 그나마 나은데, 36 위였습니다.) 아무튼, 자바스크립트가 많이 사용되고 있기는 하지만 많은 단점을

가지고 있는 것 또한 사실입니다. 이를 극복하기 위해 지금도 많은 연구가 진행되고 있으며, 이번 OOPSLA 에서도 두 편의 자바스크립트 관련 논문 발표가 있었고 자바스크립트의 창시자인 Brendan Eich 가 직접 “The JavaScript World Domination Plan at 16 Years”라는 제목으로 초청 강연도 했습니다. OOPSLA 위성 워크숍과 다른 프로그램까지 합치면 자바스크립트 관련해서 꽤 많은 논문이 발표되었습니다. 카이스트 박창희군도 FOOL 워크숍에서 “An Empirical Study on the Rewritability of the with Statement in JavaScript”라는 제목으로 논문 발표를 했습니다.

저는 자바스크립트가 정말 문제가 많다면 자바스크립트를 대체할 수 있는 좋은 웹 프로그래밍 언어를 처음부터 잘 설계할 수는 없을까 하는 의문을 갖고 있었습니다. 물론 이미 널리 사용되고 있는 프로그래밍 언어를, 설사 그 언어가 나쁜 언어라 할지라도, 새로운 좋은 언어로 대체하는 것은 쉽지 않은 일임을 잘 알고

있지만 말입니다. 그런데 구글에서도 저와 같은 생각을 했나 봅니다. 구글에서 최근에 Dart 라고 불리는 새로운 웹 프로그래밍 언어를 발표했더군요. Dart 는 OOPSLA 위성학회인 Dynamic Languages Symposium 과 위성워크숍 Virtual Machines and Intermediate Languages 의 초청 강연에서 자세하게 소개되었습니다.

비교적 큰 크기의 애플리케이션을 자바스크립트로 개발하기 어려운 이유는 첫째 자바스크립트로 프로그래밍 할 경우에 프로그램 구조를 명확하게 이해하기 어렵고, 둘째 타입 시스템이 없으며, 셋째 모듈 시스템이 없어서 라이브러리 지원이 미약하며 (이 때문에 최근 자바스크립트에 모듈 시스템을 도입하려는 움직임이 있으며 카이스트 류석영 교수님 연구실에서도 이에 대해 연구 중입니다), 넷째 여러 가지 분석도구 지원이 부족하며, 마지막으로 startup 성능이 나쁘기 때문이라고 합니다. 반면에 Dart 는 대형 애플리케이션 개발을 목표로 설계되었으며, 그 특성은 첫째 Scala, Clojure, 또는 Erlang 에서 제공하는 것과 같은 concurrency primitives 를 제공함으로써 대형 프로그램 개발을 지원하고, 둘째 기본적으로는 dynamic language 이지만 선택에 따라 정적 타입을 지원하고, 셋째 모듈 시스템을 지원함으로써 라이브러리를 언어 차원에서 지원하고, 넷째 eval 이 없으며, 다섯째 가상머신 뿐만 아니라 컴파일을 지원함으로써 빠른 startup 성능을 이끌어낼 수 있다는 점입니다. Dart 의 기본적인 구조는 아래 그림과 같습니다.



Dart 가 매력적으로 보이는 것은 사실이나 아직 시작 단계이고 자바스크립트가 여러 가지 측면에서 부족한 점이 많지만 계속 좋은 방향으로 발전하고 있다는 점을 생각했을 때 Dart 와 자바스크립트의 미래에 대해서 성급한 결론을 내리기는 어려운 것 같습니다. 하지만 만약 구글이 안드로이드와 크롬 개발에 Dart 를 적극적으로 사용한다면, Dart 와 연동했을 때 매우 효과적인 웹 프로그래밍을 위한 킬러 라이브러리를 제공한다면, 막대한 자금을 Dart 를 홍보하고 지원하는데 투자한다면 웹 프로그래밍 언어의 왕좌를 차지할 수 있지 않을까 싶습니다. 구글과 Dart 의 행보를 앞으로 좀 더 관심을 갖고 지켜봐야겠습니다.

#### 4. Onward! Film



그림 5 Presenting A Day in the Life of Video-based Requirements Engineering

Onward! Film 은 개인적으로 가장 기대한 프로그램 중 하나였습니다. Onward! Film 은 슬라이드 발표 위주의 학회 형식에서 탈피하여 영상을 이용해서 프로그래밍 언어와 소프트웨어 공학 이야기를 풀어나가자는 것이 그 취지였습니다. 공교롭게도 프로그램 시간이 Best Student Paper award 를 수상한 논문과 Distinguished Paper award 를 수상한 논문들 발표세션과 겹쳤지만, 짬을 내어 Onward! Film 세션에 들어가 볼 수 있었습니다. Siemens industrial solution 에서 준비한 영상을 봤는데 requirement engineering 에 대한 이야기였습니다. 다소 신선하기는 했지만 슬라이드 발표를 동영상 발표로 대체했다는 느낌이 강하게 들었습니다. 그래도 역시 동영상으로 보여주니까 슬라이드 발표에서는 전달하기 힘든 부분들을 좀 더 쉽게 전달할 수 있는 것 같았습니다.

## 5. 맺음말

열심히 연구한 결과가 좋은 학회 논문으로 결실을 맺는다는 것은 참으로 기분 좋은 일이 아닐 수 없습니다. 다른 사람과 협력해서 함께 연구를 진행하는 것 또한 즐거운 일이라 생각합니다. 에스토니아에 계시는 Keiko Nakata 박사님, 일본에 계시는 Jacques Garrigue 교수님과 함께 연구를 할 수 있어서 굉장히 큰 힘이 되었으며 또한 즐거웠습니다. Keiko 박사님께서서는 SPLASH 에 참석하셨는데, 학회기간 중간에 짬짬이 시간을 내어 진행 중인 일에 대해 의견을 나눌 수 있어서 더욱 좋았습니다. Keiko 박사님과 Jacques 교수님께 깊은 감사의 말을 전하고 싶습니다. 또한 학회에 참석할 수 있도록 지원해준 ROSAEC 센터에 감사를 드립니다. 끝으로 연구, 논문작성, 논문발표, 학회참석에 이르기까지 모든 면에서 아낌없이 지원해주시고 응원해주신 박성우 교수님께 감사를 드리며 본 글을 마칩니다.



그림 6 Portland Pioneer Square 에 있는 전 세계 이정표; 그런데 왜 우리나라는 울산이 이정표에 나와 있는지 모르겠습니다.