Trip Report on SPLASH 2010

Reno, Nevada, U.S.A. Oct. 16 ~ Oct. 23.

박창희(changhee.park@kaist.ac.kr)

1. 들어가는 말

SPLASH 2010 학회와 FOOL'10 워크샵 참석을 위해 미국 Nevada 주에 위치한 Reno라는 도시에 다녀왔습니다. Reno는 라스베이거스와 함께 미국의 도박의 중심지로 알려져 있으며 제가 머물던 John Asquaga's Nugget Hotel 역시 카지노 호텔이었습니다. 이 도시는 사막성 기후로 매우 건조하여 비도 많이 오지 않고, 일교차가 커 낮에는 덥고, 밤에는 추운 날씨가 계속 된다고 합니다. 혹시 몰라서 우산을 가지고 갔었는데, 입국심사에서 미국 심사원이 우산을 보고 Reno에 비가 오지 않는다고 우산을 가져오지 않아도 됐을 것이라고 하였지만, 제가 머무는 동안 하루 비가 오는 행운(?)을 맛볼 수 있었습니다.

SPLASH(Systems, Programming, Languages, and Applications: Software for Humanity)는 OOPSLA에서 이번 년도 바뀐 이름으로 좀 더 일반적인 의미로 학대되어 OOPSLA외에도 DLS(Dynamic Languages Symposium), ILC(International Lisp Conference), PLoP(Pattern Languages of Programs), Onward! 등과 같은 여러 학회들을 같이 개최하였습니다. 또한 학회들 외에도 크고 작은 여러 워크샵들이 함께 열렸는데, 저는 이 중에 FOOL(Foundations of Object-Oriented Languages) 워크샵에 논문발표를 위해 참석을 하였고 학회 중에는 OOPSLA와 Onward! 몇몇 세션에 참석을 하였습니다. FOOL에서의 제 발표 경험과 다른 사람들의 인상적이었던 발표를 위주로 보고서를 작성하도록 하겠습니다.

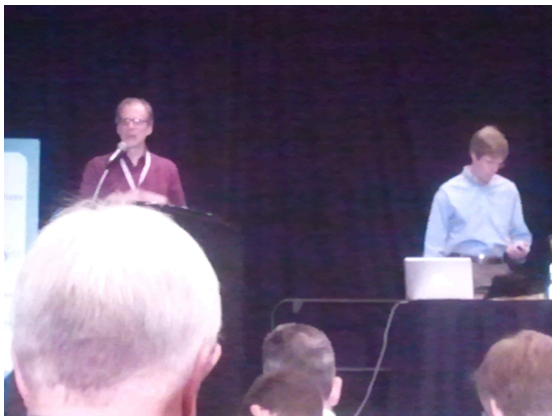

그림 1 . SPLASH General Chair이신 William R. Cook(왼쪽)과 OOPSLA Chair이신 Marti Rinard(오른쪽)

그림 2 . 호텔에 머물면서 종종 느끼하지 않은 음식이 필요할 때 이용했던 국수 집

2. FOOL'10에서의 논문 발표

그림 3 FOOL 워크샵 장소

FOOL은 제가 미국에 도착한 다음날인 17일 오전 9시부터 오후 5시까지 열렸는데, 두 Invited talk과 선택된 논문 발표로 구성이 되어있었고 자유로운 분위기 속에서 발표를 하고 질문을 주고받는 형식으로 진행되었습니다. 저의 발표순서는 제일 마지막인 오후 4시 반이었고, 주제는 ROSAEC 여름 워크샵에서 소개해 드렸던, "Adding Pattern Matching to Existing Object-Oriented Languages"로 함수형 언어들이 많이 가지고 있는 패턴 매칭을 이미 존재하는 객체 지향 언어들에 어떻게 추가할 수 있는지, Fortress 언어에 우리가 지난 몇 달간 패턴 매칭을 추가하였던 경험과 그 방법에 대해서 소개하는 내용이었습니다.

국제 워크샵에 처음으로 참석을 하여 발표 경험이 없었던 저로서는 매우 부담이 컸고 긴장을 많이 했지만, 발표를 하고 난후 깨달은 점도 많이 생기고, 자신감도 얻을 수 있었습니다. 우선 제일 걱정이 되었던 부분은 영어로 발표를 해야 하고, 질의응답도 모두 영어로 이루어진다는 것이었습니다. 발표야 정해진 시간만큼 가기 전에 연습을 해서 가기 때문에 준비한 대로만 하면 문제가 없다고 하지만, 발표 도중이나 이후에 자유롭게 질문을 하는 분위기에서 당황하지 않고, 영어를 이용하여 잘 대답하기란 경험이 많이 없는 사람에게는 여간 부담스러운 일이 아닐 것입니다. 다들 그렇게 하시겠지만, 저도 이점을 극복하기 위해, 교수님과 상의 후 예상 질문들과 그에 따른 예상 답변을 미리 생각을 해두었고, 특히 리뷰어들에 의해 논문에 취약점으로 지적되었던 부분에 대한 질문들에 대해서는 철저히 준비를 하여, 몇몇 질문들에 대한 답변을 아예 슬라이드 한 장으로 만들어 발표할 때 얘기를 하기도 하였습니다.

제 논문에서 리뷰어들에 의해 가장 많이 지적되었던 취약한 부분 두 가지는 제목이 "Adding Pattern Matching to Existing Object-Oriented Languages"인데 Fortress에 패턴 매칭을 추가하는 것이 어떻게 임의의 이미 존재하는 객체지향언어로 추가하는 것으로 확대될 수 있는지 하는 문제와 우리가 소개하는 패턴 매칭 추가 방법에 새로운 것이 무엇인지, 즉 contribution이 무엇인지 하는 문제였습니다. 첫 번째 문제에 관해서는 임의의 객체지향언어에 바로 적용시키기 보다는 우리가 소개하는 패턴 매칭 추가 방법을 적용하기 위해 존재하는 객체지향언어들이 가지고 있어야할 특징에 대한 전제를 슬라이드 한 장에 추가하였고, 두 번

째 문제에 관해서도 기존의 방법들에 비해서 패턴을 정의하기 위해 별다른 노력이 들지 않는다는 내용과 keyword pattern을 이용하여 이미 정의된 패턴을 확장시킬 수 있다는 점을 역시 슬라이드에 넣어 발표 때 미리 얘기를 하도록 하였습니다. 이렇게 가장 취약한 질문에 대해서 답변을 미리 발표를 통해 대답을 하게 되니, 편안한 마음으로 발표에 임할 수 있었던 것 같습니다.

또한 가끔씩 영어 자체를 못 알아듣거나, 내용이 이해가 되지 않는 질문이 나와 당황하는 경우가 있는데, 이럴 때 당황 하지 말고, 질문을 다시 해달라고 부탁을 한다거나, 그래도 못 알아들을 경우 솔직하게 얘기를 하고, offline에서 자세한 얘기를 하자고 제안하는 것이 좋은 방법이라는 것도 깨달을 수 있었습니다. 실제로 유럽권 나라에서 온 외국인이 질문을 하였는데, 발음이 알아듣기 힘들어 offline에서 얘기할 것을 제안하였고, 발표가 끝난 뒤 서로 충분한 얘기를 나누어 만족할 만한 답변을 줄 수 있었습니다.

걱정도 많았고 탈도 많았지만, 발표가 무사히 끝난 뒤 아쉬움도 많이 남았지만, 다음에도 이런 기회가 주어진다면, 더욱더 잘 할 수 있을 것 같다는 자신감을 얻을 수 있었습니다. 이런 좋은 경험을 할 수 있었다는 것이 이번 여정에 가장 큰 수확이었다고 생각합니다.

3. The Case for Evolvable Software

OOPSLA의 keynote중에 하나로서 뉴멕시코 대학의 여교수님이신 Stephanie Forrest님이 발표를 해주셨습니다. 제목에 Evolvable에서 예상이 되는 만큼, genetic algorithm을 복잡한 소프트웨어에 적용을 하여 각종 버그를 찾아내어 치료를 하기도 하고, 더 나아가 프로그램을 개선할 수도 있다는 내용으로 현재 연구하신 내용에 대한 얘기를 해주셨습니다. 발표초반에 찰스다윈을 소개하면서 적자생존의 법칙을 얘기하고 프로그램도 이와 마찬가지로 진화할 수 있다고 하는 부분이 흥미로웠습니다. 정확한 메커니즘에 대해서는 복잡하여 내용이해가 잘 되지 않았지만, 커다란 아이디어는 코드의 부분들을 genetic algorithm을 이용하여 여러 가지 조합으로 결합을 해보고, 계속해서 더 나은 코드조합을 찾아나간다는 내용이었습니다. 실제로 마이크로소프트에서 만든 ZUNE이라는 mp3 플레이어에 내장된 프로그램에 이 방법을 적용해 보았는데, 다른 테스트 방식에서 발견되지 않았던 버그를 발견하고 고치기도 했다고 하였습니다. 발표 마지막 부분에서는 버그 감지와 치료를 넘어서 프로그램을 더 나은 방향으로 개선하는데 이 genetic algorithm을 이용하는 가능성에 대해서도 말씀을 하셨는데, genetic algorithm을 들어본 적은 있었지만, 이런 생물학적인 원리가 컴퓨터분야에도 적용되어 실용적으로 쓰일 수 있다는 사실이 참 신기하고 흥미로웠습니다. 하지만 아직 진화를 하는 방향을 정의하는 문제와 같은 해결해야 될 문제들이 많이 남아있어 이와 관련 연구들은 활발히 진행중에 있다고 합니다.

4. An Experiment About Static and Dynamic Type Systems

독일의 Duisburg-Essen 대학에서 오신 Stefan Hanenberg 님이 발표해 주신 내용으로,

static type system이 dynamic type system보다 소프트웨어를 개발하는데 좋은 영향을 미치는지 의심을 제기하는 논문 내용에 대한 발표였습니다. 일반적으로 static type system을 가진 언어들이 소프트웨어 개발 시간도 단축한다는 주장이 있었지만, 실행된 실험에서는 그렇지 않다는 결론이 나왔다는 것에 매우 흥미로웠습니다.

실험을 위해서 학생들을 두 그룹으로 나누고, 같은 프로젝트를 하는데, 한 그룹은 static type system을 가진 언어를, 다른 그룹은 dynamic type system을 가진 언어를 사용하게 하였습니다. 채택된 언어는 사전 지식이 없는 언어이여야 하기 때문에 발표자가 직접 간단한 객체지향언어를 만들었는데, 하나는 static type system버전으로 다른 하나는 dynamic type system 버전으로 작성을 하여 동일하게 언어에 대한 교육을 두 그룹에 실시하였다고 합니다. 그리고 주어진 작업시간 외에는 결코 다른 시간에 프로젝트를 하지 못하게 하였고, 프로젝트는 간단한 파서를 작성하는 일이었습니다. 그렇게 실험을 마친 후에, 평가는 개발하는데 걸린 개발시간과 테스트를 통해 제대로 구현했는지 확인해 보는 방식이었는데, 놀랍게도 개발시간에서는 dynamic type system을 이용한 그룹이 더 앞섰고, 테스트에서도 static type system을 이용한 그룹과 비교해서 별다른 차이점이 발견되지 않았다고 합니다. 이것은 static type system이 개발 시간도 단축한다는 기존 관념을 깬 테스트로 저도 이점이 실제로 많이 궁금하였는데, 테스트결과가 예상과 반대로 나왔다는 점이 매우 흥미로웠습니다. 하지만, 실험방식에 있어서 각 그룹의 개인의 능력의 차이와, 프로젝트를 하나만 실시했다는 점, 평가를 오직 개발부분에만 치중했다는 점에 한계가 있어 더욱더 많은 실험이 요구되어진다고 합니다.

5. Lime: a Java-Compatible and Synthesizable Language for Heterogeneous Architectures

IBM 연구팀(Joshua Auerbach et al.)에서 연구한 내용에 대한 발표로, heterogeneous system에서 실행되어질 수 있는 Lime이라는 새로운 언어에 대한 얘기를 해 주었습니다. 현재 heterogeneous system에서는 다양한 구성요소로 컴파일 될 수 있는 언어, 많은 다른 병행성을 잘 표현할 수 있는 능력, 필요로 될 때 마다 프로그램을 각 구성요소로 분리할 수 있는 dynamic run-time system, 시스템의 각 구성요소로 쉽게 옮겨질 수 있는 능력, 많은 사람들이 쓸 수 있도록 하는 익숙함 등이 요구되어지고 있는데, 이 Lime이라는 언어가 이러한 요구사항을 최대한 반영하도록 디자인된 언어라고 이야기를 해주었습니다. Java에 기초를 두고 있고, side effect를 최소로 하는 함수형 언어의 특징들을 가지고 있으며, 각 실험을 통해 CPU나 FPGA에서도 실행될 수 있다는 것을 보여주었습니다. 하지만 아직 최적화에 관한 부분은 연구에 진행 중이라고 하였습니다. 점점 더 복잡해지고 있는 컴퓨터하드웨어 구조에 맞추어서 그 요구사항에 맞는 새로운 언어 디자인의 필요성이 증가하고 있고, 그에 따라 계속해서 새로운 언어들이 디자인되고 개발되고 있다는 것과 언어 디자인 연구의 중요성을 깨달을 수 있었습니다.

6. A study of Java's non-Java Memory

일본의 IBM 연구팀(Kazunori Ogata et al.)에서 발표를 해주신 내용으로, JVM의 heap 메모리 영역이 아닌 메모리영역을 non-Java 메모리 영역이라고 하는데, 많은 응용 프로그램들이 이 메모리 영역을 쓰고 있고, 이 영역에 대한 분석을 최초로 다룬 연구라고 소개한 데에 굉장히 흥미로웠습니다. 실제로 DaCapo benchmarks와 Apach DayTrader 같은 Java 응용 프로그램을 대상으로 하는 실험결과 이런 응용프로그램들이 많은 non-Java 메모리 영역을 사용하고 있고 모두 소진한 경우에는 메모리부족 런타임 에러가 발생할 수 있는데, 이런 예 중에 하나는 많은 클래스들을 한꺼번에 불러오려고 하는 경우라고 합니다. 연구팀은 이런 non-Java 메모리영역을 효율적으로 분석을 하기 위해 운영체제의 메모리 정보까지 제공하는 MARUSA(Memory Analyzer for Redundant, Unused, and String Areas)라는 툴을 만들었고, 실험은 이 툴을 이용해서 이루어졌다고 합니다. 그리고 이 툴을 통해 JIT 컴파일러의 경우에도 컴파일 과정에서 최적화를 위해 순간적으로 이 non-Java 메모리영역을 많이 사용한다는 것이 발견되었다고 합니다. 이러한 툴을 이용하게 되면, 메모리 관리에 있어서 더 안정적이고 효율적인 소프트웨어 개발이 가능하겠다는 생각을 했습니다. 하지만 실험이 오직 IBM J9 JVM과 Linux 환경에서만 이루어 졌는데, 다양한 환경에서의 실험부족은 아쉬운 점이었습니다.

7. Art, Science, and Fear

그림 4 발표 후 질문자와 질의응답 시간을 가지고 있는 Benjamin C. Pierce

Onward!의 keynote로서 지난 학기 스터디로 Types and Programming Languages 책을 공부한 적이 있었는데, 이 책의 저자인 Benjamin C. Pierce님을 뵙게 되어 신기하기도 하고 영광스러웠습니다. 이 발표에서는 슬라이드 내용의 95%이상이 모두 사진으로 이루어져 있었습니다. 사진 찍는 것을 좋아하신다고는 들은 것 같은데, 실제로 작품들을 보니 아마추어인 제가 봐도 상당한 실력을 가지고 계신 것 같았습니다. 학습하기 힘들 것으로 보이는 창의성, 직관 같은 능력이 어떻게 일상적인 것에서 창출되어지고 배워질 수 있는지 여러 사진 작품들을 통해서 보여주려고 하시는 것 같았습니다. 실제로 발표는 불을 다 끄고 어두운 상태에서 사진 감상을 하면서 발표자가 사진을 어떻게 찍게 되었는지, 설명하는 형태로 진행되었는데, 사진만

으로도 이야기를 잘 풀어나갈 수 있다는 것과 새로운 발표 스타일에 신선한 충격을 받을 수 있었습니다.

8. Understanding Reduced Resource Computing

MIT 연구진(Martin C Rinard et al.)들이 연구한 내용으로 이번 OOPSLA의 의장이신 Martin C Rinard 님이 발표를 해주셨습니다. 말씀이 빠르셔서 알아듣기 힘들었지만, 재치 있는 언변과 묘한 카리스마로 청중들을 사로잡는 모습이 인상적이었습니다. 이 발표에서는 원래 계산에서 쓰는 자원(컴퓨터의 경우 메모리 등)을 줄여 계산을 수행할 수 있는 그 메커니즘에 대해 소개를 해주었는데, 이러한 계산들은 어떤 집합에 대한 병렬 계산을 할 경우 부분 집합에 대해서만 수행하고, 루프의 경우에는 모든 인덱스에 대해 수행하는 것이 아니라 일부 인덱스에 대한 계산만 수행하는 등 이러한 부분 계산으로 자원을 줄이는 계산 방식을 말하고 있습니다. 언뜻 드는 생각에도 이러한 계산 방법은 계산의 정확성에서 떨어질 수밖에 없습니다. 그래서 이런 메커니즘은 계산의 오차를 허용하는 응용분야에 적용될 수 있다고 합니다. 그리고 실제로 여러 응용프로그램을 통해 실험을 해본 결과 그 정확성의 오차가 충분히 허용될만한 정확도를 보여주었다고 합니다. 계산이라고 하면 정확도가 우선이라고 생각을 해왔었는데, 유연성을 우선으로 두는 생각의 방향 전환이 이러한 연구 분야도 낳을 수 있다는 것이 흥미로웠습니다.

9. Searching Without Objectives

SPLASH의 keynote로서 Central Florida 대학의 Kenneth Stanley 님이 발표를 해 주셨습니다. 제목부터 범상치 않은데, "무엇인가를 마음에 품지 않을 때 이룰 수 있다."라는 말과 함께 발표를 시작하셨습니다. 목적 없이 탐색을 할 때 원하는 곳에 도달할 수 있다는 것이 반직관적임에도 불구하고 발표자는 탄탄한 논리로 청중들을 설득하기 시작했습니다. 인류의 많은 발명들이 처음부터 목적을 가지고 탐색을 시작해 이루어지지 않았다는 많은 예를 통해 보여주면서, 오히려 목적을 가지고 시작하는 것은 정해진 틀에 생각을 가둬 원하는 것을 더욱 얻지 못하도록 한다고 합니다.

이를 재미난 예를 통해 보여주었는데, 실험을 위해 한 인터넷 사이트를 만들었는데, 여기에서 사용자들은 여러 개의 추상적인 그림들을 받게 됩니다. 그럼 사용자들은 버튼을 눌러 각 그림에서 일부분을 띄어내고 이 조각들을 다시 합친 다음 임의적으로 섞어 새로운 그림을 얻게 되는데, 이렇게 얻어진 그림들이 구체적인 모습으로 나왔을 때, 사이트에 그 그림들을 올리게 했다고 합니다. 그러자 놀랍게도 자동차 모양에서부터 정교한 손잡이 달린 주전자까지 구체적인 사물의 모습을 띄는 그림들이 생성되었다고 합니다. 한 사용자는 속임수가 있는 것이 아니냐고 묻기도 했는데, 그림들이 섞여지는 방식은 삼각함수로 이루어진 임의의 수학적 랜덤 방식이라고 합니다.

자동차 그림이 탄생된 방식은 매우 흥미로웠습니다. 사용자는 외계인 그림에서 또 다른 그림을 생성을 하려고 했습니다. 이렇게 계속 버튼을 누르다 보니 어느 순간 외계인 눈은 바퀴가 되고 외계인 얼굴은 자동차 선체가 되어 자동차가 탄생하게 되었다고 합니다. 여기서 중요

한 점은 처음부터 자동차 그림을 생성할 목적이 있었다면, 결코 외계인 그림에서 시작을 하지 않았을 것이고, 자동차는 탄생하지 못했을 것입니다. 자동차 그림을 만들 목적이 처음에 없었기 때문에 자동차 그림을 발견할 수 있었다는 것입니다.

생물이 탄생했던 것도 여러 물질들이 결합과 결합을 통해 결국 유전자가 만들어져 생물이 탄생하게 되었는데, 자연이 시간은 많이 걸렸을 지라도, 수억 년의 생활동안 임의로 물질들을 결합했기 때문에 예상치 못했던 생물이 탄생했다는 것으로, 처음부터 생물을 만들 목적으로 물질들을 결합하기 시작했다면 결코 생물을 만들 수 없었을 지도 모른다는 예도 보여주었습니다.

이렇게 많은 발견들이 최초에 특별한 목적 없이 이루어졌다는 예를 많이 보여주면서 중요한 것은 관련된 일을 하다가 중요한 발견에 이르게 되는 징검다리를 발견하는 일이라고 하였습니다. 논문을 쓰는 많은 연구가들에게 항상 목적이 있는 문제를 찾는 것이 제일 어려운 일인데, 이런 저런 관련된 혹은 관련되지 않은 일을 하다가 문제를 찾는 경우도 많은 것 같아 이 발표에 많은 공감을 하였습니다. 이번 여정 중 제일 재미있게 들었던 발표였습니다. 특히 발표자의 재치를 곁들여서 청중을 끌어들이는 발표 스타일이 굉장히 인상적이었습니다.

10. Back to the Futures: Incremental Parallelization of Existing Sequential Runtime Systems

James Swaine et al.이 연구한 내용에 대한 발표로 많은 sequential runtime 시스템에 parallelism을 가능하게 하기 위한 메커니즘을 소개하였는데, Racket runtime 시스템에 적용해본 결과를 보여주었습니다. 기존의 방법들이 존재하는 runtime 시스템에 parallelism을 추가하기 위해 대대적인 시스템 보수를 요구하고 추가를 한다고 해도 안정성이 보장이 되지 않는 반면에, 소개되는 메커니즘은 점진적인 방식으로 안정성이 보장된 operation(이런 operation을 fast-path operation, 그렇지 않은 operation을 slow-path operation이라고 부름.)을 우선 parallelism이 가능하도록 하고 있고 이 방법을 slow-path barricading이라고 명명하였습니다. Racket의 경우에는 'future'라는 keyword를 이용하여 뒤따르는 계산이 병렬적으로 이루어짐을 명시하고, 'touch'라는 keyword를 통해 실질적으로 계산된 결과를 다른 계산과 동기화되어 받게 됩니다. 병렬적으로 수행이 된다고 해도 안전하다고 정해진 operation만 병렬적으로 수행이 될 수 있기 때문에 안전성이 보장된다고 합니다. 그리고 실제로 Racket에 future를 구현을 해본 결과, 적정 수준의 노력이 개발자들에게 소요가 되었다고 합니다. Multi-core architecture가 주류를 이루고 있는 상황에서 속도 향상을 위한 parallelism의 중요성이 점점 증가되고 있고, 이를 지원하기 위한 언어개발과 runtime 시스템에 대한 연구가 활발히 진행되고 있다는 것을 실감할 수 있었습니다.

11. F#: Taking Succinct, Efficient, Typed Functional Programming into the Mainstream

SPLASH의 keynote로서 마이크로소프트의 연구원인 Don Syme님이 발표를 해주셨습니다.

튜토리얼과 같이 F#에 대한 전반적인 소개를 해주는 발표였습니다. F#은 C#의 진화물로서 OCaml에 영향을 받은 .NET기반의 함수형 언어라고 합니다. F#을 통해서 복잡한 코드를 간단하게 표현할 수 있게 해주는 함수형 함수의 전반적인 장점에 대해 설명을 하면서도 오브젝트를 표현하는 방법, generic을 사용하는 방법과 parallelism을 이용하는 방법 등을 소개한 다음 Visual F#을 통해서 어떻게 F#을 이용하여 코딩을 하고 디버깅을 할 수 있는지 보여주었습니다. 요새 객체지향언어에 함수형 언어의 특징을 결합하거나 함수형 언어에 객체를 표현할 수 있는 특징을 결합하는 시도들이 많이 행해지고 있는데, 이쪽에 관련된 연구를 할 경우 F# 언어에 대해서 알아보는 것도 좋은 연구 자료가 될 것 같다는 생각을 할 수 있었습니다.

12. 맺는 말

뿌듯함과 아쉬움이 교차하였던 여정이었습니다. 일단 그동안 가 볼 기회가 없었던 미국에 혼자 가게 되고 워크샵에 발표를 무사히 마치고 왔다는 점은 정말 값진 경험 이였습니다. 하지만 다음과 같은 점들이 아쉬움으로 남았고, 다음에 이런 기회가 또 주어진다면 꼭 보완해서 가야겠다고 다짐했습니다.

① 영어 : 영어를 열심히 공부해야겠다고 생각했습니다. 발표를 위해서도 중요하지만, 국제학회에서 청중으로서 발표자와 의사소통을 무리 없이 해내기 위해서는 원활한 영어 실력이 요구되어진다는 것을 절실히 깨달을 수 있었습니다.

② 예습 : 이번에는 발표 준비를 하는 데에 너무 급급한 나머지, 듣기로 예정했던 다른 발표에 관련된 자료들을 미리 보지 못해 후회를 하였습니다. 듣기 전에 미리 내용을 파악하고 듣는 다면 더욱더 알찬 시간을 보내고 그 분야에 대한 이해도를 높이는데 많은 도움이 되었을 텐데 그 점이 아쉬웠습니다.

③ 적극적인 자세 : 학회에서 발표를 하고 발표를 듣는 것도 중요하지만, 그것 외에도 참석한 다름 사람들과 안면을 트고 연구에 대한 정보를 공유하는 것도 매우 중요하다는 것을 깨달았고, 그것을 위해서는 먼저 다가가 자신감 있게 말을 걸 수 있는 적극적인 자세가 필요하다는 것을 절감했습니다. 또한 발표를 들을 때에도 궁금한 점이 있으면 발표 도중에 적극적으로 질문을 하든지 발표 이후에 발표자를 찾아가서 질문을 하여 모르는 것에 대한 이해를 넓히는 것이 필요하다는 것도 깨달았습니다. 이번에 모든 것이 새로운 것이라 겁을 먹고 주눅이 들어 적극적인 자세로 임하지 못했던 것이 아쉬웠습니다.

④ 여유로운 마음가짐 : 엄숙하고 정중한 분위기가 아닌 자유로운 분위기 속에서 발표가 이루어진다는 것이 상당히 인상적이었습니다. 발표자들은 발표를 즐기는 듯이 여유롭게 하고 청중들은 궁금한 것이 있으면 발표 어느 때든 자유롭게 질문을 하여 수시로 의사소통을 하는 모습이 엄숙한 공식적인 자리를 예상했던 저로서는 신선한 충격이었고 마음을 여유롭게 가지고 즐기는 자세를 가지는 것이 필요하다는 것을 알게 되었습니다.

도착해서 일정 첫 날에 발표를 마친 후 계속해서 다른 사람들의 발표내용을 들으면서, 이 큰 학회에 언젠가 제 자신도 서서 발표를 하는 그림을 계속해서 그려보았습니다. 상상만 해도 설레고 떨리는 일이지만, 내년이고 내후년이고 꼭 도전을 해보리라 생각을 해보았습니다. 이번 여정은 두려움을 물리치게 하고 자신감을 북돋아주는 '경험'이라는 강력한 무기를 얻게 해준 소중한 자산이 되었습니다.