ICSE 2026 Trip Report

Rio de Janeiro, Brazil

2026-04-19

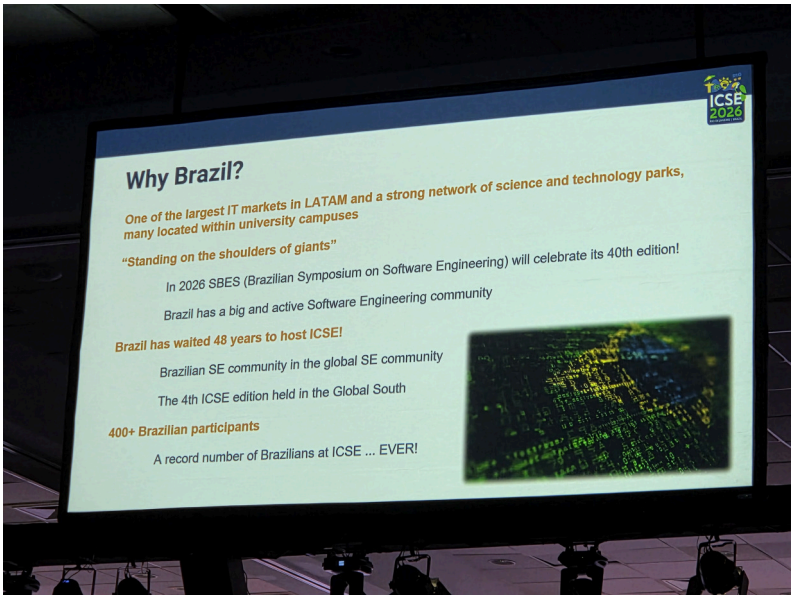
Wonseok Oh
Korea University

1. ICSE 2026

이번 ICSE 2026 학회는 브라질 리우 데 자네이루(이하 리우)에서 개최 되었습니다. 실은 출발 전 브라질 치안에 대한 걱정을 많이 했었습니다. 현재 브라질 정부와 갱단 간의 갈등이 있는 상황이고, 주변 사람들로부터 브라질을 갔다가 강도나 상해를 당했다는 이야기를 많이 들었기 때문입니다. 그래서 학회를 도대체 왜 브라질에서 하는지 의문이 들었습니다.

결론적으로 리우는 굉장히 아름다운 도시였습니다. 기본적으로 (강도를 하지 않는) 사람들이 굉장히 친절했습니다. 언어가 통하지 않아도 여행객인 저희를 많이 도와주려 하였고, 또 저희가 브라질어를 사용하면 아주 기쁘게 호응해주었습니다.

그림 1 — 왜 브라질에서 열었는지 설명을 해줬다. 보통 이런 설명을 했나 모르겠다.



리우에 관한 이야기는 Section 2.2에서 자세히 다루도록 하고, 여기서는 학회에 관한 이야기를 쓰도록 하겠습니다.

1.1. 정보

그림 2 — 약 1,500명이나 참가했다.

그림 3 — 제출 수도 역대 최고다.

올해 ICSE는 역대 최대 규모로 열렸습니다. 사실 처음 들었을 때는 우리나라 더위/추위 역대 갱신 같은 느낌으로 다가왔었는데, 실질적인 숫자를 보니 체감이 남달랐습니다. 논문 제출 수는 1,469편에 메인 학회 참석은 총 1,500명에 달한다고 합니다.

이렇게 크게 된 이유는 LLM의 발전이 한몫 했다고 생각합니다. 특히 Writing에서 엄청난 속도 상승이 있었을겁니다. (저도 많은 도움을 받고 있습니다) 요즘 논문 쓰는데 AI 도움을 받지 않으면 사실 뒤처져 있는거나 다름 없는데, 이걸 또 오용/악용하는 사람들이 있다고 합니다.

그림 4 — 이걸 봐주다니... 학회의 아량이 굉장히 넓은 것 같다.

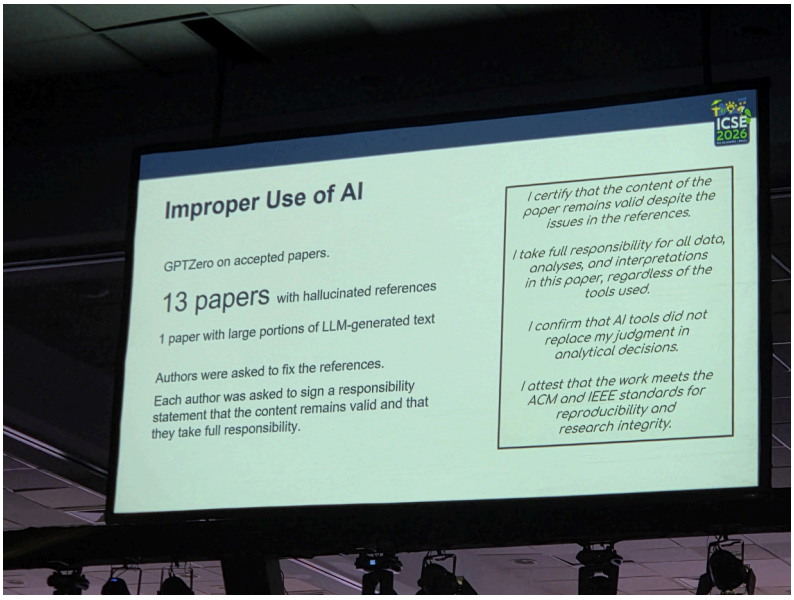


그림 4는 그 사례들을 보여줍니다. 13편의 논문에서는 **가짜 레퍼런스**가 발견 되었고, 1편의 논문은 그냥 **LLM 통짜로 썼다가** 걸렸다고 합니다. 수정 차원으로 일단락 되었는데, 사실상 모두에게 하는 경고였습니다. 다들 알아서 잘하고 있겠지만 '한번 쯤이야' 식으로 넘어갔다가 그 한 번이 걸리면 신뢰에 엄청난 타격을 주니, 조심 또 조심하도록 합시다.

그림 5 — 세션 목록인데 귀여워서 찍었다

이번 ICSE는 엄청 많은 세션이 동시다발적으로 진행되었습니다. 같은 시간 내에 거의 10개가 넘어가는 세션이 돌아가고 있었습니다. 그래서 듣고싶었던 발표들이 겹치는 일이 많아 모두를 들을 순 없어서 선택적으로 잘 들어야 했었습니다.

또 흥미로운 점이라면 AI4SE 세션만 스물몇개가 넘어갔으며 SE4AI도 열몇개의 세션이 있었습니다. 요새 컴퓨터과학의 트렌드가 잘 나타난 것 같으면서도, 이제 AI 주제가 아니면 논문 내기도 어렵겠구나라는 생각을 했습니다.

1.2. 흥미로운 발표

인상 깊게 들었던 발표들을 공유 드리겠습니다.

1. Jan Bosch Keynote: Towards an Awesome AI-driven Future for Software Engineering

ICSE 2026의 처음을 장식한 키노트였습니다. 주제는 SE의 미래입니다.

(1)

요즘 개발자들이 AI에 대체되고 있다는 말이 많습니다. 발표자는 전적으로 동의하지 않는다고 합니다. AI의 도움을 받아 프로젝트의 스케일이 커짐에 따라 오히려 많아지면 더 많아졌지, 절대 적어지지 않을거라고 합니다.

이 내용은 저도 어렴풋이 생각하고 있던 내용입니다.

제3차산업혁명을 생각해보면, 이 때 얼마나 많은 자동화가 일어났을까요. 과도기 때는 '나의 일자리를 보장해달라'하고 러다이트 운동도 일어났지만, 결국 인간의 노동력은 줄지 않고 오히려 더 많아졌습니다.

하드웨어도 한번 볼까요. 기술이 발전함에 따라 하드웨어들의 효율이 극대화 되었습니다. 과연 '아 효율이 올랐으니, 하드웨어 생산 덜 해도 되겠다'라는 명제로 흘러왔을까요? 답을 아시겠지만, 절대 아닙니다. 오히려 스케일을 키워서 더 많은 일을 하도록 하였습니다.

소프트웨어는 어떨까요. 과연 '아 AI로 효율이 올랐으니, 개발자 없어도 되겠다'라는 명제로 흘러갈까요? 세상만사 다 똑같지 않을까 생각합니다.

(2)

과거의 SE는 사람이 코드를 어떻게 쓸지를 궁금해 했다면, 현재의 SE는 기계가 코드를 어떻게 쓸지에 궁금해 하는 것 같다고 합니다.

여기서 굉장히 중요한 포인트를 짚습니다. 이제 미래의 SE(기계를 위한)로 나아가려 하는데 아직도 패러다임은 과거의 SE(사람을 위한)에 머물러있다는 겁니다.

Agile, DevOps, CI/CD 등은 사람 중심 개발방법론이지, AI를 위한 것이 아닙니다. Architecture는 어떨까요. 지금까지는 예측을 중요시 했지만, AI는 확률적으로 돌아가고 있죠. 심지어 조직문화도 과거에 머물러 있습니다. 우리는 사람을 역할(팀장, 팀원, …)로 나누지만, AI는 무엇을 만들 수 있냐에 더 관심이 있습니다. (이 예시는 Agent를 생각해보면 될 것 같습니다)

(3)

우리는 무슨 목적으로 프로그램을 만들고 있는지 생각해봅시다.

지금까지는 '쓰면 좋을 것 같은' 프로그램을 많이 만들었다고 합니다. 그런데 이런 프로그램은 GitHub에 널렸고 관심조차 받지 못합니다. 앞서 말한 과거의 패러다임에 머물러 있으면, 이렇게 밖에 못 만든다고 합니다.

우리는 이제 '이미 하고 있는'걸 자동화 하는 (혹은 더 잘하는) 프로그램을 만들어야 한다고 합니다.

(4)

이제 솔루션을 제시할 차례입니다. 그런데 앞선 오프닝에서 시간을 많이 깎아 먹어서 솔루션 부분이 좀 주먹구구식으로 넘어 가버렸습니다.

그래서 솔루션을 제대로 이해하진 못했지만, 생각의 전환과 문제 의식에 있어 도움이 될까하여 이렇게 공유 드립니다.

2. Can SAT Solvers Keep Up With the Linux Kernel's Feature Model? (**Distingushed Paper**) [1]

이 발표는 사람들이 진짜 궁금해할만한 부분을 잘 건드려 조사한 발표입니다. 기술적인 기여는 적으나, 이런 측면으로도 Distingushed Paper까지 받을 수 있다는 점을 공유드리고 싶어 여기에 적습니다.

SAT Solver의 성능은 굉장히 많이 올라왔다고들 합니다. 그래서 이게 실제로 Linux Kernel의 옵션 선택 문제에 활용될 수 있는지 조사한 연구였습니다.

Linux Kernel 옵션 선택 문제는, Linux Kernel을 설치해봤다면 다들 알겠지만, 굉장히 짜증나고 귀찮은 작업입니다. 안 그래도 Linux 계열 형제들이 많은데, 이 모두가 요구하는 바가 조금씩 틀립니다. 그래서 하드웨어에 맞춰/원하는 Linux 계열에 맞춰 옵션을 잘 줘야하는데 뭐 조금만 잘못주면 빌드가 실패하거나 뻑 나거나 여간 짜증나는 일이 아닙니다.

즉, 우리가 자주 사용하는 SAT Solver가 실제로 (1) 많이들 사용하는 프로젝트이며 (2) 사람들이 고통받아하고 있는 Linxu Kernal에 적용될 수 있는지를 조사한 것입니다.

실험은 매년 Linux 버전을 가져와 그 시점까지 개발된 SAT Solver들로 약 10년 정도의 대규모 조사를 진행하였습니다.

결과는 SAT Solver는 전혀 Linux Kernel을 따라가고 있지 못한다였습니다. 오히려 해가 늘어가면 갈수록 Linux Kernel의 복잡도가 올라가는 것에 비해 SAT Solver의 성능은 더 떨어지고 있다고 합니다.

개인적으로 앞선 Keynote의 내용과 일맥상 통한다고 생각합니다. 최근 SAT Solver들은 Competition에서 우수한 성능을 보이는 것이 목적이라는데, '사람들이 실제로 원하는 것'과는 거리가 있음을 보여주는 것 같습니다.

다들 연구를 진행할 때 **자신이 푸는 문제가 정말 도움이 될까**라는 고민을 많이 해보면 좋겠습니다.

3. Automated Bug Frame Retrieval from Gameplay Videos Using Vision-Language Models [2]

Industry Track에 제출된 논문인데, 실생활에 일어나는 문제를 해결하기 위해 연구를 진행했다는 점에서 이 발표를 공유드리려고 합니다.

그래픽 엔진에는 다양한 버그들이 존재합니다. 게임을 하는 유저들은 그래픽 버그가 발생하면 이를 영상으로 찍어 고쳐달라고 보고합니다.

자신들이 겪은 문제는 이렇습니다. 영상이 핵심만 담은게 아닌 n분짜리 영상으로 업로드 되고, 이런 보고가 너무 많다는겁니다. 즉, 할 일이 많아 죽겠는데 이 영상을 계속 보고있을 순 없다는 게 문제 의식이었습니다.

이를 위해 버그가 일어나는 프레임을 짚어주는 도구를 만들었습니다. 방법은 프레임들을 주고 AI에게 '어디가 제일 문제인 것 같아?'라고 물어보는 것이었습니다. 단순해 보이지만, 가장 큰 문제는 영상에 프레임이 너무 많다는겁니다. 그래서 프레임을 샘플링할 필요가 있었고, 샘플링을 위해 키프레임 개념을 가져 왔다고 합니다.

키프레임은 영상의 변화가 일어나는 지점입니다. 즉 영상의 변화가 크게 일어나는 부분들을 추출하였고, 이는 프레임의 2% 정도이지만 99%의 버그를 담고 있었다고 합니다.

이러한 다운샘플링 기법을 통해 AI를 적극 활용하여 버그 영상들을 효과적으로 요약할 수 있었다고 합니다.

혹시나 연구 주제 잡기 어려워하는 학생들이 있다면, 이렇게 자신이 실제로 겪은 문제들을 해결해보는 것도 좋은 방향이라 생각합니다.

4. Testora: Using Natural Language Intent to Detect Behavioral Regressions [3]

해당 발표는 문제의 정의가 깔끔해서 공유하려 합니다.

풀고자 하는 문제는 이렇습니다.

GitHub PR로 인해 의도치 않은 버그가 발생하는지 확인하는 Regression Test를 만드는 것입니다.

간단해 보이지만 해결해야할 부분은 두 군데입니다. (1) 의도치 않은 버그가 무엇인지 (2) Regression Test는 어떻게 만들 것인지입니다. 특히 (1)이 왜 해결해야하는 부분이지라는 생각을 할 수도 있는데, 우리는 '의도치 않은'을 정의할 필요가 있기 때문입니다. 왜냐면 PR로 결과가 바뀌었는데 이게 해당 PR이 원하던 바인지 아니면 진짜 몰랐는데 바뀌었는지 판단을 해야하기 때문입니다.

이 연구에서는 '의도치 않은'을 'PR의 설명과 다른'으로 정의하였습니다. 굉장히 간단하지만 명료한 선택지였습니다.

그래서 (1) PR 전후로 결과 차이가 있는 Regression Test를 만드는 LLM과 (2) 이게 의도한 버그인지 아닌지를 판단하는 LLM을 만든 후, 두개를 이어 붙여 해당 연구를 마무리 하였습니다.

이 글을 읽는 모두 자신의 문제 정의에서 필요한 부분들이 다 정의가 되었는지 한 번 확인해보면 좋을 것 같습니다.

5. Hybrid Fault-Driven Mutation Testing for Python [4]

이건 Python 연구를 하는 사람들이 많아 공유 드리기 위해 가져왔습니다.

별건 아니고 Python에서 드디어 그럴듯한 Mutation Testing 기법이 나왔습니다. 지금까진 Cosmic-Ray가 가장 괜찮은 Mutation Testing이었는데 개인적으로 Python에 썩 적합하지 않다고 생각했습니다. Python 버그는 다양한 곳에서 벌어지는데 이를 모두 커버하지 못하고 있었거든요.

이러한 점을 지적하며 Python에 적합한 Mutation Testing을 만들었다고 합니다.

나중에 Python Test Generation 연구를 하게 된다면, 이 발표를 꼭 기억하고 써먹을 수 있으면 좋겠습니다.

1.3. 내 발표

이번 ICSE 2026에서는 제 연구도 같이 발표하였습니다. 이번 발표는 특별했던 점이 두가지 있는데 (1) Artifact Award를 받았다는 점 (2) 그리고 질문자를 직접 찾아갔다는 점입니다.

그림 6 — Artifact Award는 굉장히 뿌듯했다.

그림 7 — 더 잘해야 된다는 부담감이 있었다.

근데 뭔가 Award를 받고 나니 발표도 더 잘해야할 거 같아서 진짜 빈틈없이 준비하려고 했습니다. 특히 Q&A를 엄청나게 준비했는데요. 예상되는 질문 한 7개 정도를 뽑아 답변을 만들고 달달 외웠습니다.

하지만 세상은 결코 쉽게 흘러가지 않더라구요. 예상 밖의 질문이 들어왔습니다. 일단 질문의 의도를 파악하지 못했습니다. 대충 Multiple Type을 지원하냐는 질문이었는데, 제 연구에서 Multiple Type이 끼어들 그런게 없었거든요. 그래서 일단 답변으로 지원한다고 했는데 이게 굉장히 찝찝했습니다.

그래서 발표가 끝난 후 난생처음 제가 직접 질문자를 찾아가서 다시 물어봤습니다. 질문은 제 연구가 Tie가 일어났을 때 혹은 특정 상황에서 Union을 자동으로 작성해주냐는 거였습니다. 대답은 No였고, 자칫 잘못하면 잘못된 정보를 줄 뻔 한거였습니다.

예전 같았으면 회피하듯이 그냥 넘겼을텐데, 용기를 내서 처음 질문자를 찾아가서 물어보고 대답을 정정해보니 굉장히 기쁜 일이란걸 알게 되었습니다. 더군다나, 찾아가서 알게 되었는데 질문자의 친구 분이 제 연구를 굉장히 관심있게 보던 외국인 분이셨고, 이런 적극적인 모습 덕에 저에 대한 호감도가 더 상승하지 않았을까 싶습니다.

1.4. 네트워킹

이번 ICSE 2026에서 처음으로 네트워킹다운 네트워킹을 했습니다. 관련 이야기들을 여기에 풀어보려고 합니다.

(1)

제 가장 큰 숙제는 발표가 아닌, 마이클 프라델 교수님께 안부인사 겸 연구진행 상황 말씀 드리기였습니다. 스몰톡에 부담감을 갖고 있던 제가 난생처음으로 스몰톡을 걸어야하는 상황이었습

니다. 영어도 미숙해서 긴장이 엄청 되긴했지만 '뭐 결국 사람사는 곳인데 대화할 의지를 보이면 되겠지'라는 마음으로 부딪혀 보기로 했습니다.

근데 이걸 어쩌나. 1,500명이나 되는 학회장에서 마이클 프라델 교수님을 찾기란 사막 속에서 바늘 찾는 꼴이었습니다. 이렇게 가다가는 영원히 못 만나겠다 싶어 마이클 프라델 교수님이 세션 체어일 때를 노렸습니다. 그리고 세션이 끝나고 인사드리려 하는데 웬 아시아계 여성 분이 가로채가는 거 아니겠습니까. 심지어 마이클 프라델 교수님도 약속이 있는지 그 여성 분 보고 자기 바쁘니까 좀 있다 이야기하자고 했습니다. 여성 분은 아랑곳 않고 끝까지 따라가면서 말 걸기는 했지만, 저는 뭐 이렇게 해서는 제대로 된 인사도 못 나누겠다 싶어 다음 기회를 노렸습니다.

이번엔 마이클 프라델 교수님이 발표하는 타임을 노렸습니다. 그래서 미리 가서 오실 때까지 기다리고 있었습니다. 타이밍은 아주 좋았습니다. 여유로워 보이시고 사람도 얼마 없어 말 걸기 딱 좋았습니다.

처음에 인사를 거니 저를 알아보지는 못했습니다. 근데 그도 그럴 것이 저도 체어 세션에서 처음 봤을 때 마이클 프라델 교수님을 못 찾았습니다. 마이크를 쥐시고 진행을 하자 그제서야 알아챘거든요. 이게 동양인이 서양인을 볼 때 구분 못하듯이, 서양인도 동양인을 볼 때 구분을 못해서 그런 것 같습니다.

하여튼 그렇게 인사를 드리고 망가진 영어 스피킹으로 여차저차 연구 상황도 공유 드렸습니다. 마이클 프라델 교수님은 '나 여기 발표하는데 너 연구랑 관련 있어 보이거든? 한번 듣고 참고 해봐바'라고 해주시며 저를 반갑게 맞이해주셨습니다. 그렇게 제가 생에 처음으로 건 스몰톡이 마무리 되었습니다.

이걸 하며 느낀 것이 우연히 지나가다가 만나는 인연은 거의 없으며, 네트워킹을 얻기 위해 노력을 굉장히 열심히 해야한다는 점을 깨달았습니다. 이 글을 읽는 분들 중 네트워킹을 해야겠다는 마음가짐이 있으시다면, 많이 노력하여 필연을 우연처럼 가장해야한다는 점을 꼭 염두하면 좋겠습니다.

(2)

이번엔 반대로 제가 우연히 만난 외국인이 있었습니다.

메인 홀에 서서 점심을 먹고 있었는데, 외국인 한 분이 와 스몰톡을 걸었습니다. 마이클 프라델 교수님과의 스몰톡을 위해 시원스쿨을 들어서인지 생각보다 스몰톡이 자연스럽게 흘러갔습니다.

외국인 분은 생각보다 저에게 관심이 많았고 제 연구가 무엇인지, 발표는 언제하는지, 심지어 Award를 받았다는 것까지 알고 있었습니다. 이유가 있었는데, 이 외국인 분도 JS/TS에서 타입을 달아주는 연구를 했기에 Python 타입 달아주는 연구가 궁금했다고 합니다.

제가 마이클 프라델 교수님을 찾아다녔듯이, 이 외국인 분도 제가 보이면 말을 걸어야지라는 생각을 했을거라고 생각하니 뿌듯하면서도 참 기뻤습니다.

(3)

한국인 분들도 꽤나 많았습니다.

차수영 교수님을 통해 성대 다른 교수님과 학장님을 만나뵐 수 있었고, 고려대학교 정보보안학과에서 ICSE 제출하신 분도 만나뵐 수 있었고, 유신 교수님네 졸업생 분들도 만나뵐 수 있었습니다.

외국에서 만나는 한국인 분들은 어찌나 그리 반가운지. 덕분에 새로운 분들도 만나고 이런저런 정보 공유도 할 수 있었습니다.

그림 8 — 차수영 교수님과 함께 한 컷.

2. 베뉴

리우까지 가는데는 크게 두 가지 루트가 있는데, 하나는 애틀란타 경유고 하나는 두바이 경유입니다. 그런데 두바이는 상황이 좋지 않다보니 저희는 애틀란타 경유를 하게 되었습니다.

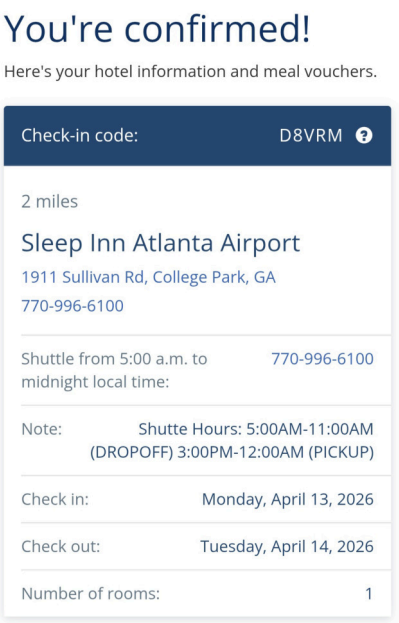
2.1. 애틀란타 경유

웬 경유지를 Trip Report에 넣냐라고 할 수 있는데, 애틀란타에서 정말 일이 많았었습니다.

일단 비행기를 탔는데 전원이 들어왔다 나갔다를 반복 했습니다. 몇 번을 반복하고 나니 비행기에서 다 내리라고 합니다. 그리고 2시간 뒤에 다시 비행기에 태워 출발하겠다고 합니다.

그렇게 기다리다가 방송이 나오는데 갑자기 8시간 뒤인 오전 10시에 출발하겠답니다. 지금 새벽 2시니까 자기들이 호텔 바우처를 줄테니 나가서 묵고 오랍니다.

그림 9 — 바우쳐로 호텔 예약에 성공했다.


You're confirmed!

Here's your hotel information and meal vouchers.

Check-in code: D8VRM

2 miles

Sleep Inn Atlanta Airport

1911 Sullivan Rd, College Park, GA

770-996-6100

Shuttle from 5:00 a.m. to midnight local time: 770-996-6100

Note: Shutte Hours: 5:00AM-11:00AM (DROPOFF) 3:00PM-12:00AM (PICKUP)

Check in: Monday, April 13, 2026

Check out: Tuesday, April 14, 2026

Number of rooms: 1

그림 10 — 애틀란타의 밤 풍경.

난생처음으로 10시간 딜레이가 걸려보고, 난생처음으로 공항을 나가 우버를 타고 바우쳐로 제공되는 숙소에 묵어보는 경험을 하게 됐습니다. 썩 유쾌하진 않았지만 나름 신선한 경험이었습니다.

2.2. 리우 데 자네이루

리우를 돌아다닐 때는 다음 수칙만 지키면 안전하게 다닐 수 있습니다.

> 사람들이 가라는 곳만 갈 것
> 우버로 도어 투 도어로 이동할 것 (아무리 짧더라도)
> 해 지면 돌아다니지 말 것

주변인들의 도움을 받아 리우를 잠깐 탐방할 수 있었습니다.

그림 11 — 시내에 있는 대성당.

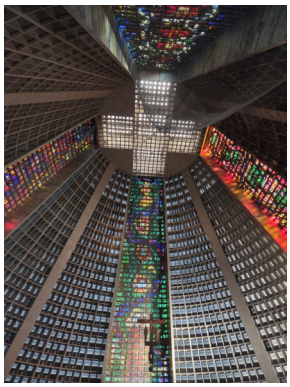


그림 12 — 알록달록 셀라론 계단.

그림 13 — 빵산 가는 케이블카.

그림 14 — 슈하스코.

건물도 큼직큼직해서 시원했고 전통 음식인 슈하스코도 굉장히 맛있었습니다. 특히 슈하스코는 여러가지 고기 부위를 들고와 잘라주는 뷔페 형태로 이루어져 있는데, 브라질에 오면 꼭 먹어보라고 추천 드립니다. (저희가 간 곳은 슈하스코 궁전이라는 곳입니다)

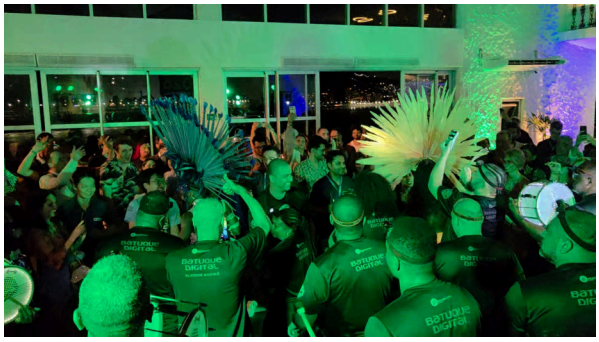
그림 15 — 빵산에서 구경한 리우 야경.

그림 16 — 이파네마 해변.

리우의 가장 큰 장점이라고 하면 자연의 모습이라고 할 수 있습니다. 높은 데서 바라보는 리우 야경은 주변 자연과 어우러져 특색있는 모습을 보였고, 해변에는 맑은 바닷물에서 서핑하고 비치발리볼하는 사람들을 볼 수 있었습니다.

그림 17 — 쇼핑몰에서 공연하는 모습.

그림 18 — 뱅퀫에 초청한 쌈바 뮤지션 그룹.



또 리우는 음악에 굉장히 열정적인 나라였습니다. 브라질은 쌈바로 유명한 도시로 알려져 있는데요. 그걸 의식해서인지 ICSE 2026 뱅퀫에는 쌈바 뮤지션 그룹을 초청하여 공연을 하였습니다. 이 공연 때문에 귀가 먹먹했지만, 언제 또 쌈바 음악을 현장에서 들어볼까요. 굉장히 재밌는 경험이었습니다.

3. 마치며

이번 학회는 애틀란타 10시간 지연을 제외하고는 모든 방면에서 만족스러웠던 학회였습니다. 재밌는 발표도 많았고, 제 발표도 만족스러웠고, 네트워킹도 열심히 시도해보고. 이렇게 좋은 경험을 할 수 있게 적극 지원해주신 오학주 교수님께 감사의 인사 올리며 이상 Trip Report를 마치겠습니다.

참고문헌

[1] E. Kuiter, U.-B. Braun, T. Thüm, S. Krieter, and G. Saake, “Can SAT Solvers Keep Up With the Linux Kernel’s Feature Model?,” 2025.

[2] W. Lu, A. Senchenko, A. Hindle, and C.-P. Bezemer, “Automated bug frame retrieval from gameplay videos using vision-language models,” arXiv preprint arXiv:2508.04895, 2025.

[3] M. Pradel, “Testora: Using Natural Language Intent to Detect Behavioral Regressions,” arXiv preprint arXiv:2503.18597, 2025.

[4] S. Alimadadi and G. Gharachorlu, “Hybrid Fault-Driven Mutation Testing for Python,” arXiv preprint arXiv:2601.19088, 2026.