[SR 경력채용] 화상 2차 면접

2021/7/22 (목)
10시~12시(GMT+09:00)

VIRGINIA TECH
DEP. OF COMPUTER SCIENCE
PHD (2021.5)
안기진

https://kjproj84.github.io

VIRGINIA TECH

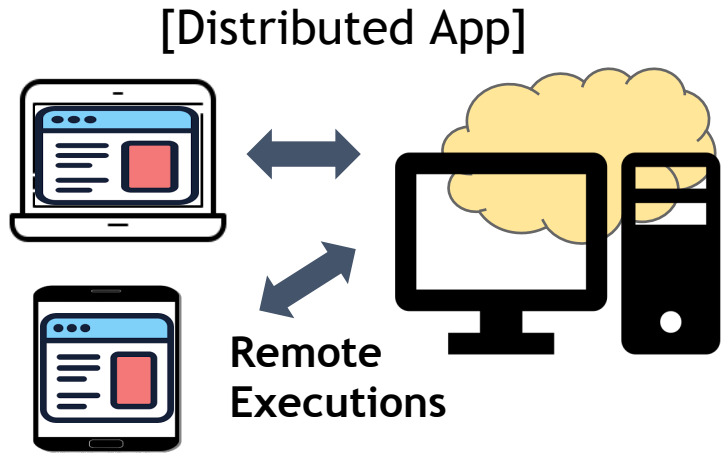
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
UT PROSIM

Dissertation Contributions

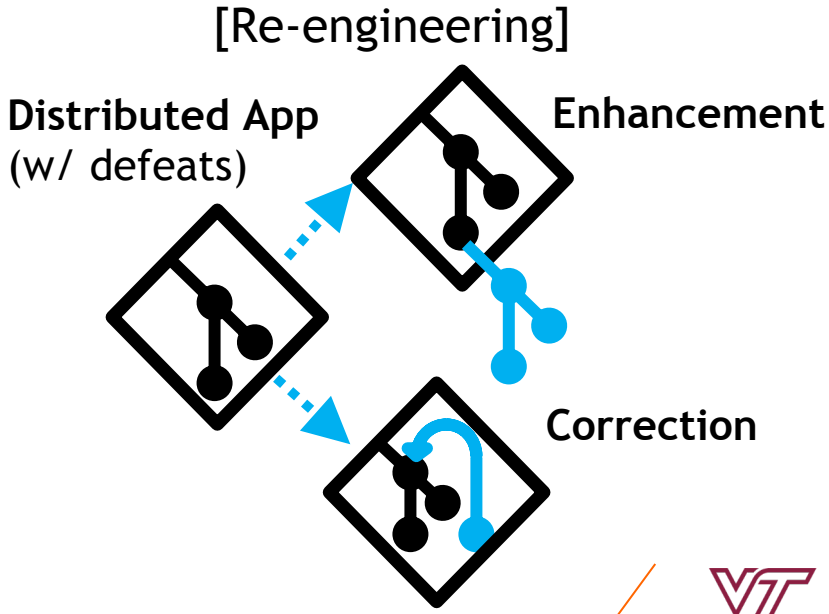
- My Research Topic:
  - Software Engineering
  - Distributed Systems (Web)
- Advisor: **Eli Tilevich**


Dr. Tilevich


Dr. Smaragdakis

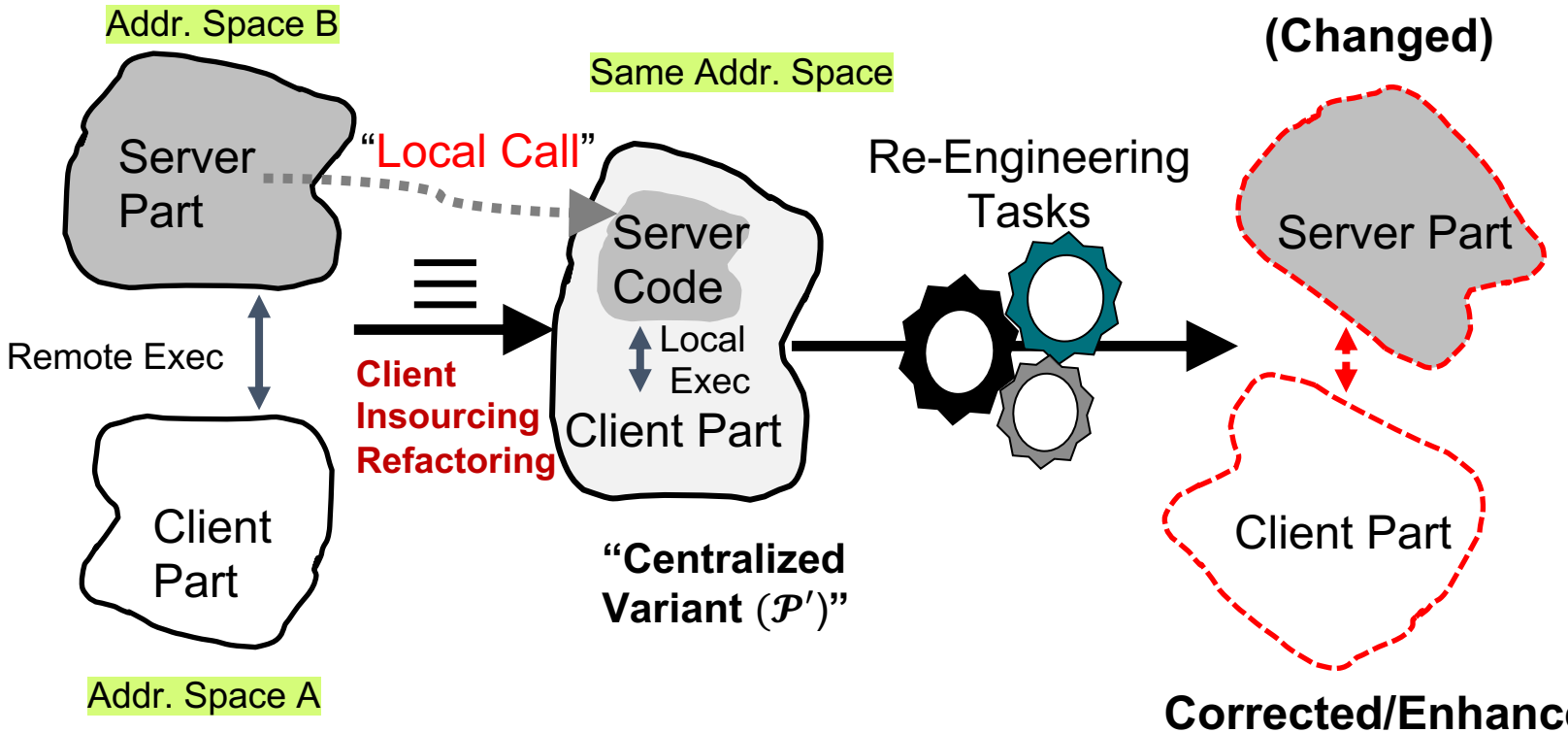


- New "**Refactoring**": "Client Insourcing"
  - *Creating a Centralized Variant ($\mathcal{P}'$) for the Distributed App ($\mathcal{P}$)*
- Value and Utility of "Client Insourcing"
  - *"Pinpointing"* Inefficiency of Distributed Programs and *"Assisting"* Programmers for their changes
  - Applying state-of-the-art techniques from **Software Engineering** to address problems in Distributed Apps



VIRGINIA TECH

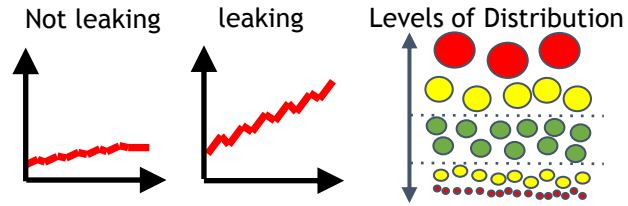
Client Insourcing Refactoring



- **Corrections Distributed Apps**

Debugging Memory Leakage/Performance Bottlenecks (very quickly)

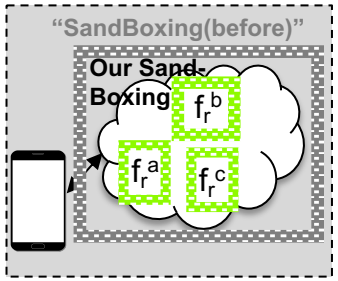
Optimizing Distribution Granuality

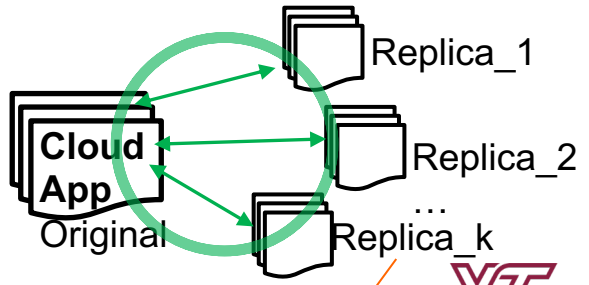


- **Enhancing/Adapting Distributed Apps**

Sandboxing with minimum overheads

Replicating Distributed Apps for Mobiles/Edges





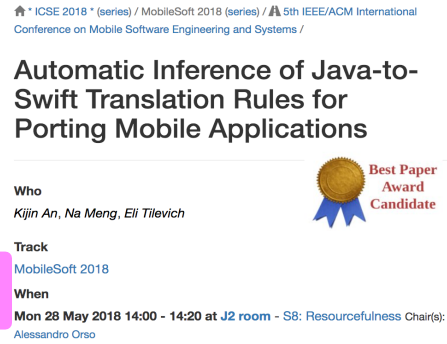
VIRGINIA TECH

Publications & Honors in PhD

(full list: https://kjproj84.github.io/publications)

No.	Paper	Conference	Area	
1.	Client Insourcing Refactoring	**WWW 2020** (19%, 217/1129, Top-tier)	System (Web)	1st Author/2
2.	D-Goldilocks	**SANER 2020** (21%, 42/199)	Software Engineering	1st Author/2
3.	Catch&Release (Debugging)	**ICWE 2019** (25%, 26/106)	System (Web)	1st Author/2
4.	Comm Web Vessels	**ICWE 2021** (17%, 22/128, Best Paper 🏆)	System (Web)	1st Author/2
5.	EdgeFy: Edge-based framework	**Submitted**	System (Middleware)	1st Author/2
6.	**[Appendix] Project1**: Differencing Cross-platform Apps	**MobileSoft 2018** (Nominated for Best Paper)	Software Engineering	1st Author/3
7.	**[Appendix] Project2**: Distributing Embedded Apps for Trusted Exec.	**GPCE 2018**	Software Engineering	2nd Author/3
8.		**Journal of Com. Lang.** (Nominated for Best Paper)	Software Engineering	2nd Author/3

- Main work presented in WWW 2020 (Top-tier)
- One Best Paper Award & Two Best Paper Nominations



- Two Doctoral Symposium Papers in WWW 2020 and ICWE 2019
- Two Spotlights from CS@VT

VIRGINIA TECH

Client Insourcing Refactoring [WWW 2020]

```
//CLIENT: app/../property-details.ts
unfavorite(event, property){
                    //Marshalling
 this.pServ.unfavorite(property)
 .subscribe(favorite //unMarshalling
    =>{ this.favorites =favorites;})
}
```

1) HTTP **Request** From Client

```
DELETE
/properties/favorite
HOST ..
[{"id":1,"city":..}]
```

Entry Point

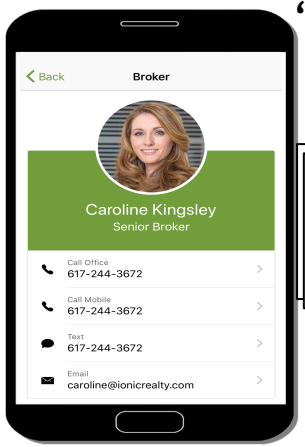
2) HTTP **Response** From Server

```
HTTP/1.1 200 OK
Content-t: json
Content-Len: …
[{"id":2,"city"…]
```

Exit Point

```
//SERVER:server.js
app.delete('/properties/favorites/'
, properties.unfavorite);
//server/properties.js
var favorites = require('./property').favs;
function unfavorite(request, response) {
 var id = request.body.id;//unMarshalling
 for (var i=0; i<favorites.length; i++){
   if (favorites[i].id == id){
       favorites.splice(i, 1);
       break;}}
   response.json( favorites );//Marshalling
}
//SERVER: server/property.js
exports.data = [{id: 1,...}];
exports.favs = [{id:2,…},…,{…}];
```

(Correspondence for "*Marshaling*" Points)



Client Code

Marshaling client-param

Unmarshaling server-return

Send REQ

Get RES

HTTP Traffics

Get REQ

Send RES

Unmarshaling client-param

Marshaling server-return

Server Code

(Client-side)

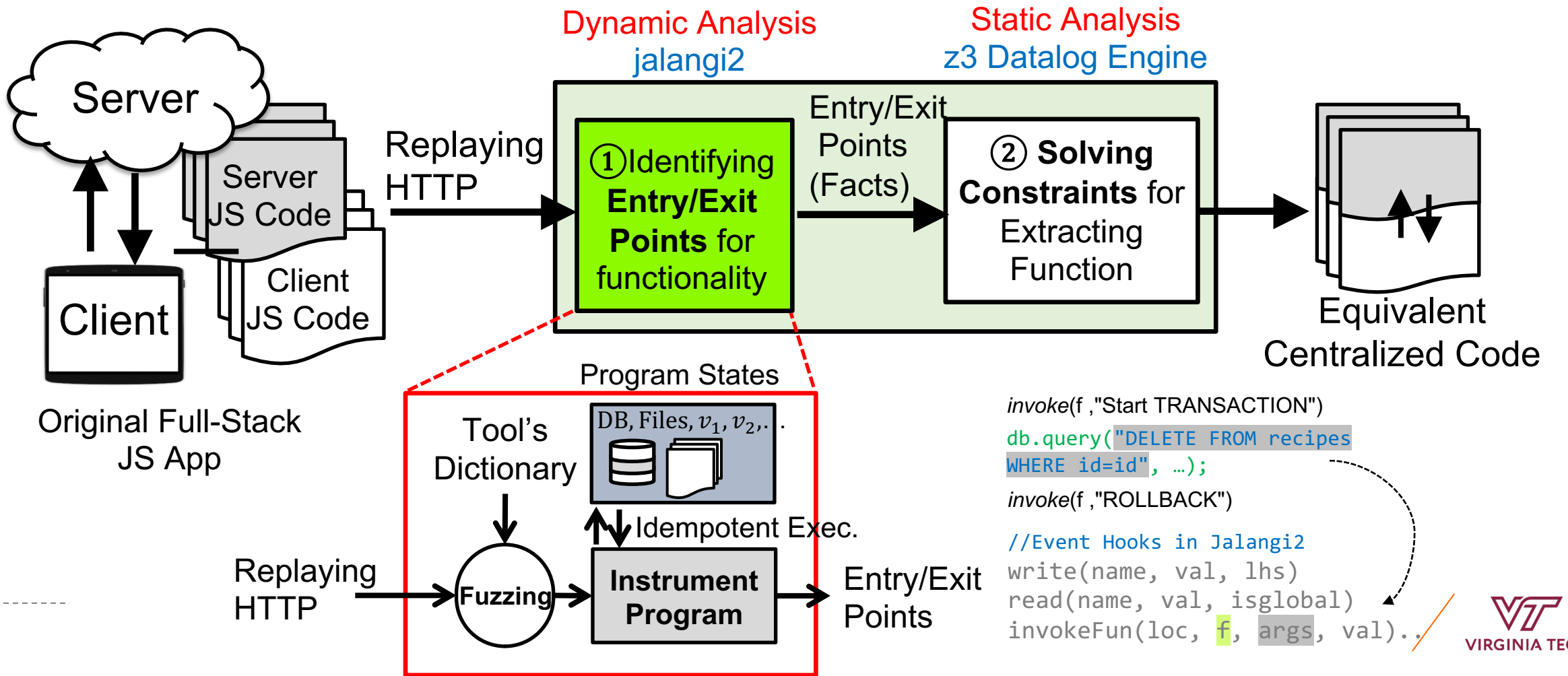
(Server-side)

Replaying/Capturing

1. Decode {client-parameter, server-return}
2. Instrument code parts that RWs these values

Client Insourcing Refactoring [WWW 2020]

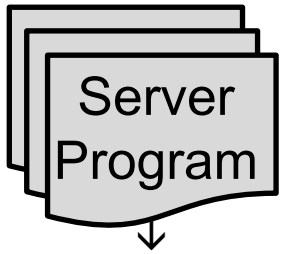
- **Fuzzing HTTP** records, **Idempotent** Executions



Client Insourcing Refactoring [WWW 2020]

- Extracting Function: Searching all dependent JS code in Entry/Exit points
- Extending *Declarative approach for Program* Analysis (z3py, Datalog Engine)
  - JavaScript: **GATEKEEPER** [Security'09]: Point-to-Analysis, **JSDep** [FSE'15]: Dependency Analysis

;;Generating Facts



$\mathbf{Write}(s_1, v\_a)$
$\mathbf{Read}(s_2, v\_b)$
$\mathbf{Write}(s_2, v\_b)$
...
$\mathbf{Ref}(s_c, v\_c, V_1)$
$\mathbf{Ref}(s_d, v\_d, V_2)$

;;Rules for Client Insourcing Refactoring

$$\mathbf{DataDep}(s_1, \mathrm{stmt}_2) \leftarrow \mathbf{Read}(s_1, v_1) \wedge \mathbf{Write}(s_2, v_1)$$

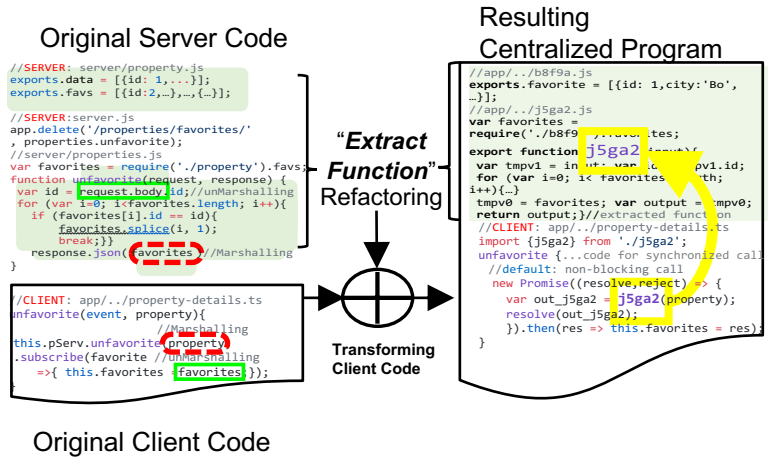
;;JS-Dep, GATEKEEPER

$$\mathbf{UnMar}(s_1, v_{\mathrm{unMar}}, V^{u_{\mathrm{id}}}_{\mathrm{unMar}}) \leftarrow \mathbf{Write}(s_1, v_{\mathrm{unMar}}) \wedge \mathbf{Ref}(v_{\mathrm{unMar}}, V^{u_{\mathrm{id}}}_{\mathrm{unMar}})$$

$$\mathbf{Marshal}(s_1, v_{\mathrm{Mar}}, V^{u_{\mathrm{id}}}_{\mathrm{Mar}}) \leftarrow \mathbf{Write}(s_1, v_{\mathrm{Mar}}) \wedge \mathbf{Ref}(v_{\mathrm{Mar}}, V^{u_{\mathrm{id}}}_{\mathrm{Mar}})$$

$$\mathbf{ExecutedStmts}(s_n, V^{u_{\mathrm{id}}}_{\mathrm{unMar}}, V^{u_{\mathrm{id}}}_{\mathrm{Mar}}) \leftarrow \left(\mathbf{DataDep}(s_n, s_1) \wedge \mathbf{Marshal}(s_1, v_{\mathrm{Mar}}, V^{u_{\mathrm{id}}}_{\mathrm{Mar}})\right) \wedge \left(\neg\mathbf{DataDep}(s_n, s_2) \wedge \mathbf{UnMar}(s_1, v_{\mathrm{unMar}}, V^{u_{\mathrm{id}}}_{\mathrm{unMar}})\right)$$

;;z3 Datalog Engine
Query ExectedStmts for specific HTTP method
{Client Param, Server Return}

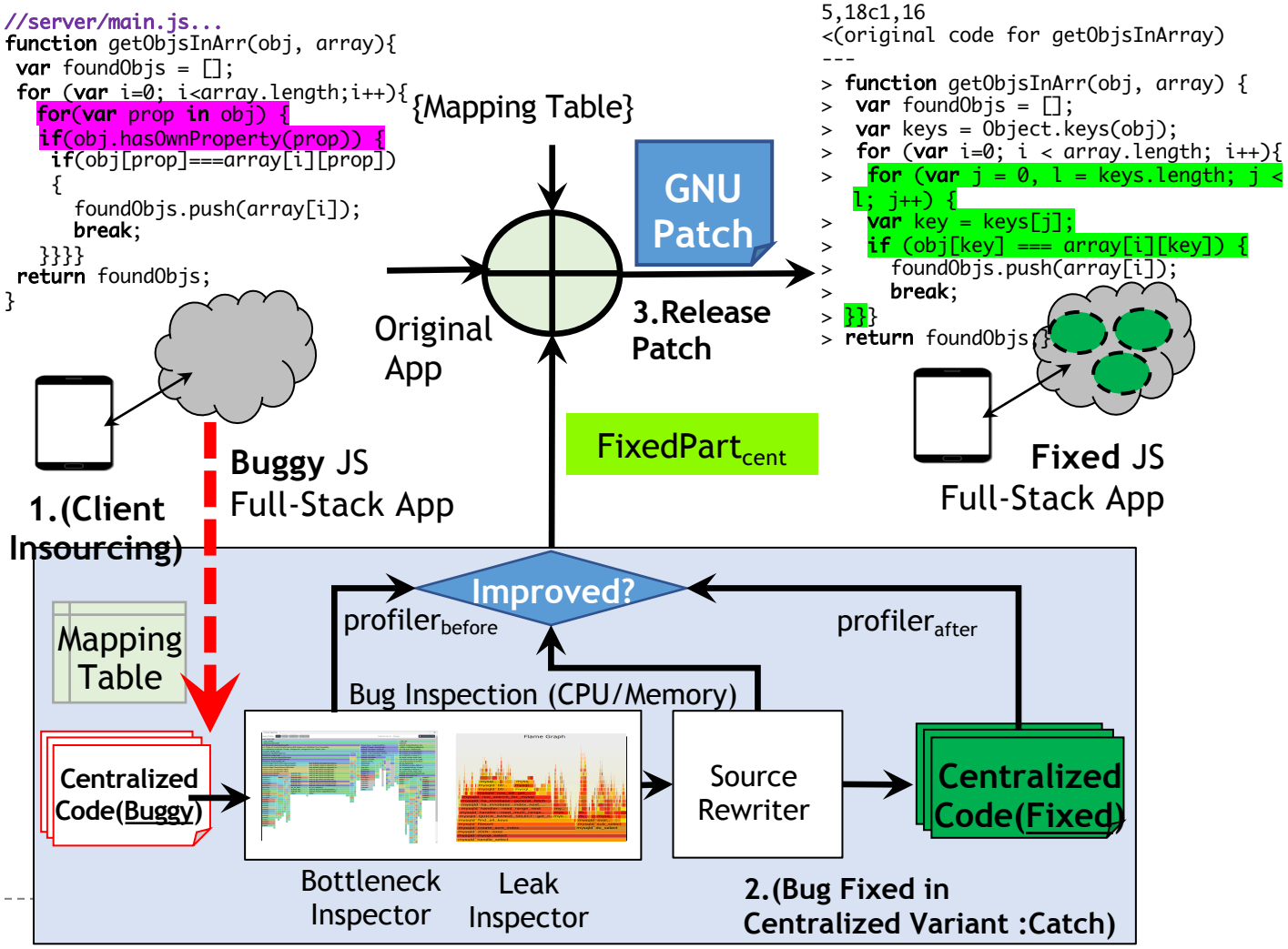


VIRGINIA TECH

Application 1: Bug Fixes [ICWE 2019]

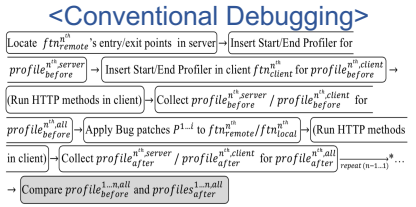
```
//server/main.js...
function getObjsInArr(obj, array){
 var foundObjs = [];
 for (var i=0; i<array.length;i++){
   for(var prop in obj) {
   if(obj.hasOwnProperty(prop)) {
    if(obj[prop]===array[i][prop])
    {
      foundObjs.push(array[i]);
      break;
   }}}}
 return foundObjs;
}
```

```
5,18c1,16
<(original code for getObjsInArray)
---
> function getObjsInArr(obj, array) {
>   var foundObjs = [];
>   var keys = Object.keys(obj);
>   for (var i=0; i < array.length; i++){
>     for (var j = 0, l = keys.length; j <
  l; j++) {
>     var key = keys[j];
>     if (obj[key] === array[i][key]) {
>       foundObjs.push(array[i]);
>       break;
> }}}
> return foundObjs;
```

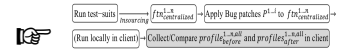


- Fixing Bugs in Centralized Variants and Generating Patches
- **90%** Reduced Time to execute Debugging Task

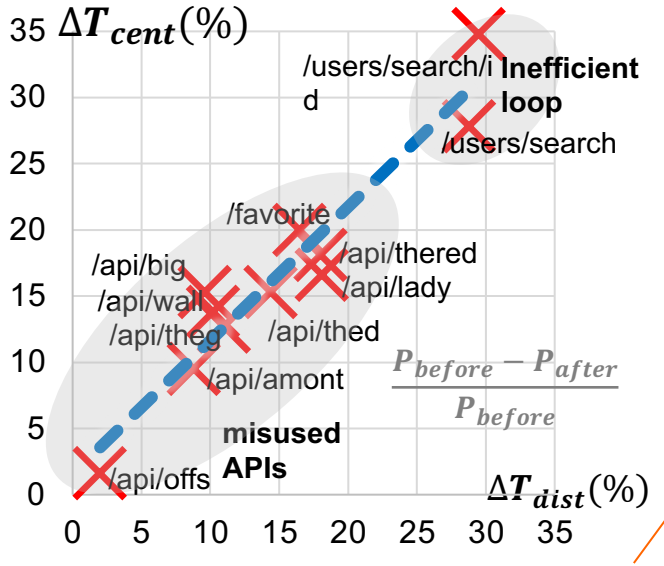
<Conventional Debugging>



<Our Simplification>



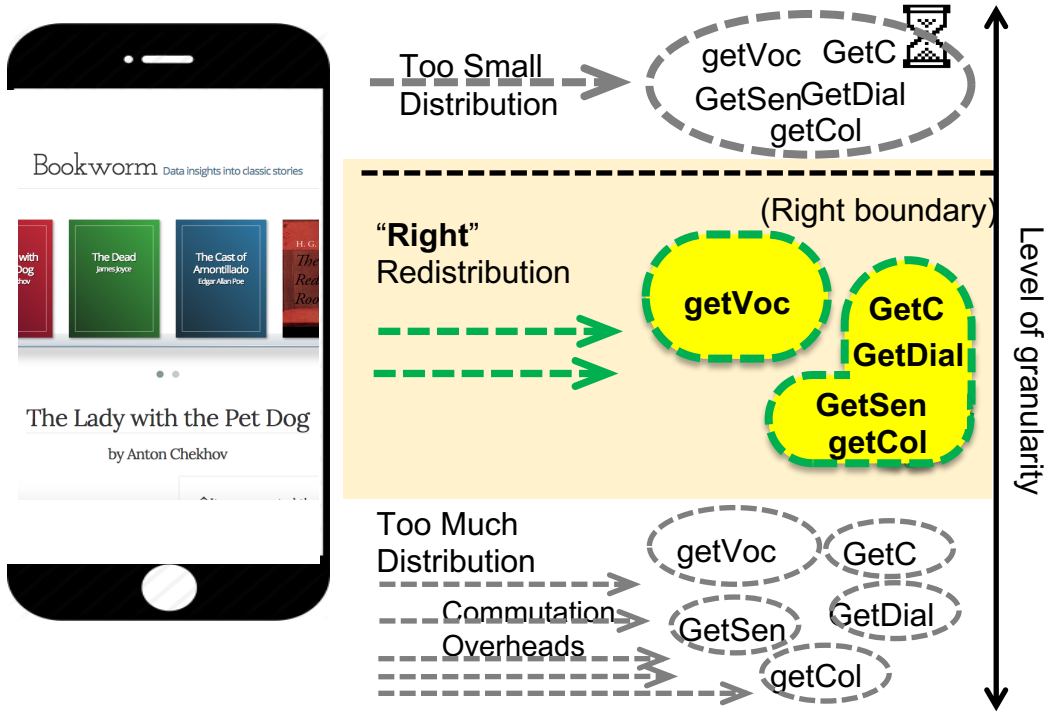
Reduced Steps



VIRGINIA TECH

Application 2: D-Goldilocks [SANER 2020]

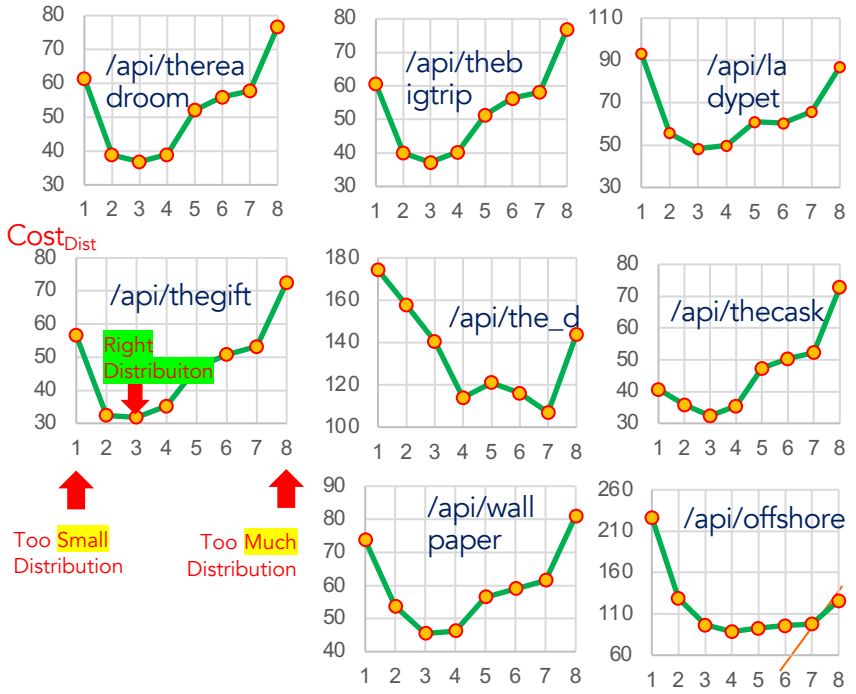
- Correcting ill-conceived Distributions
  - Ex) Nano-service anti pattern



Client-Side

Server-Side (/api/ladydog)

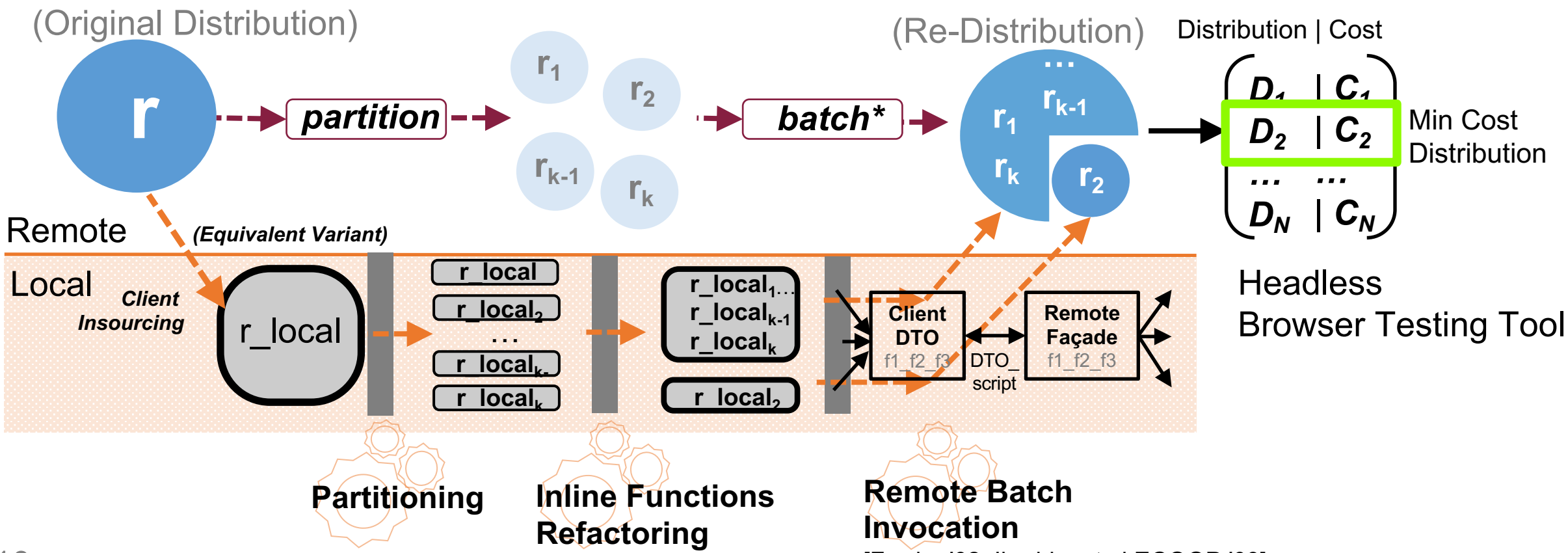
- Determine which functional distribution would minimize the cost of distributions
  - $C_{Dist\_Exec}(\mathbf{r}) = \alpha \cdot latency(\mathbf{r}) + (1-\alpha) \cdot \Sigma\ resource(\mathbf{r})$
- Large Distribution Space: Our Tool automates!
  - Ex) 394 × **4139** ~= **1.6 × 10^6** ULOCs



VIRGINIA TECH

Application 2: D-Goldilocks [SANER 2020]

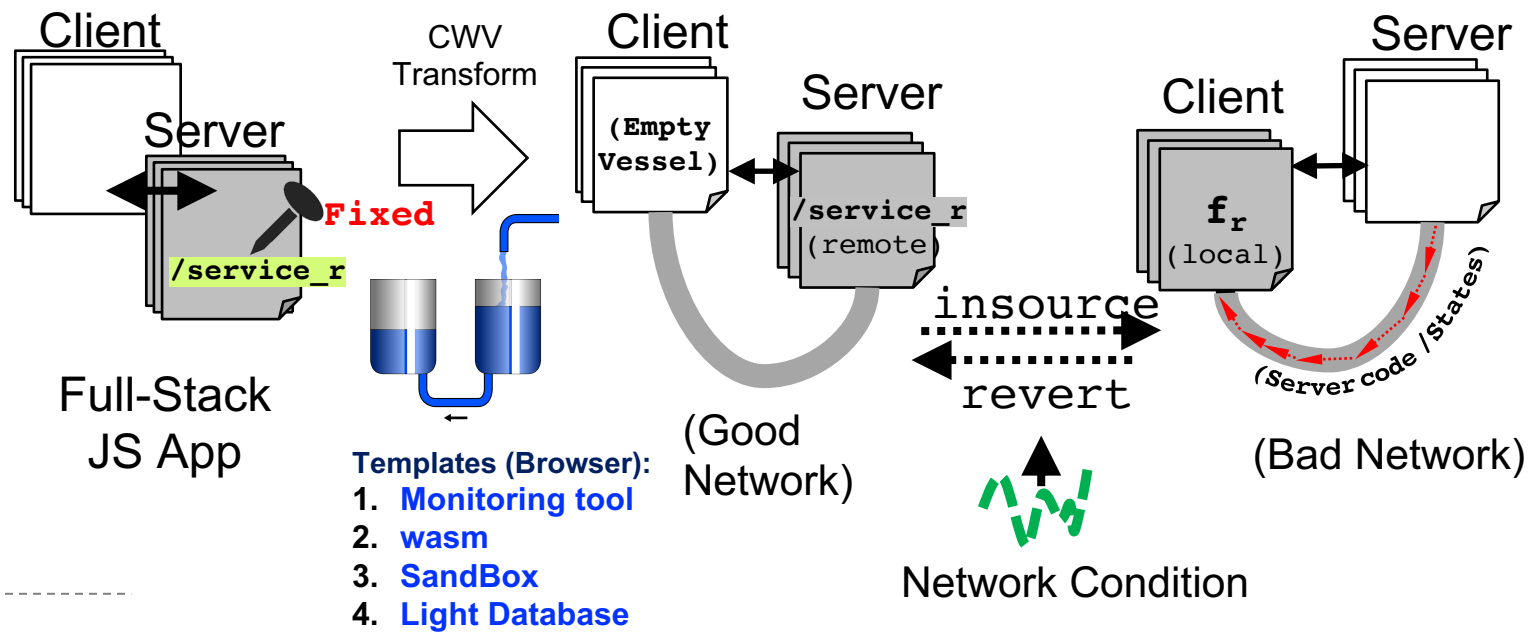
Restructuring Distribution

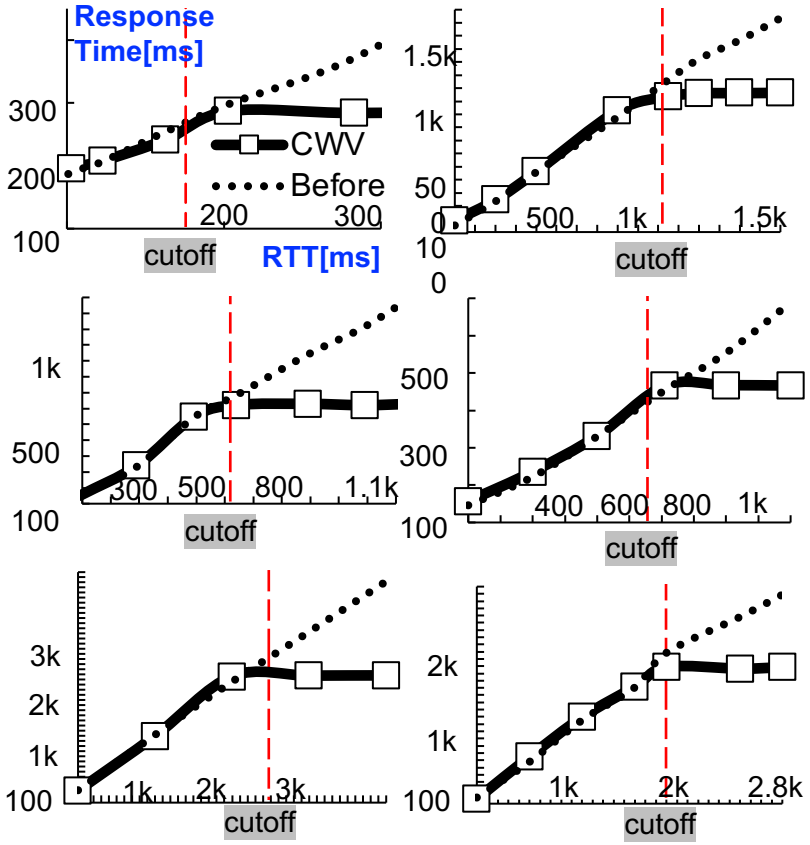


Application 3: Communicating Web Vessels (**CWV**)

Best Paper Award 🏆 [ICWE 2021]

- **Design** and **Execution Time** Mismatch – Client/Server Arch.
- Client *Insource* or *Revert* {**Code**$_{\textbf{Server}}$, **State**} based on Exec Conditions
- Automated Program Transformation for Adaptive Arch.

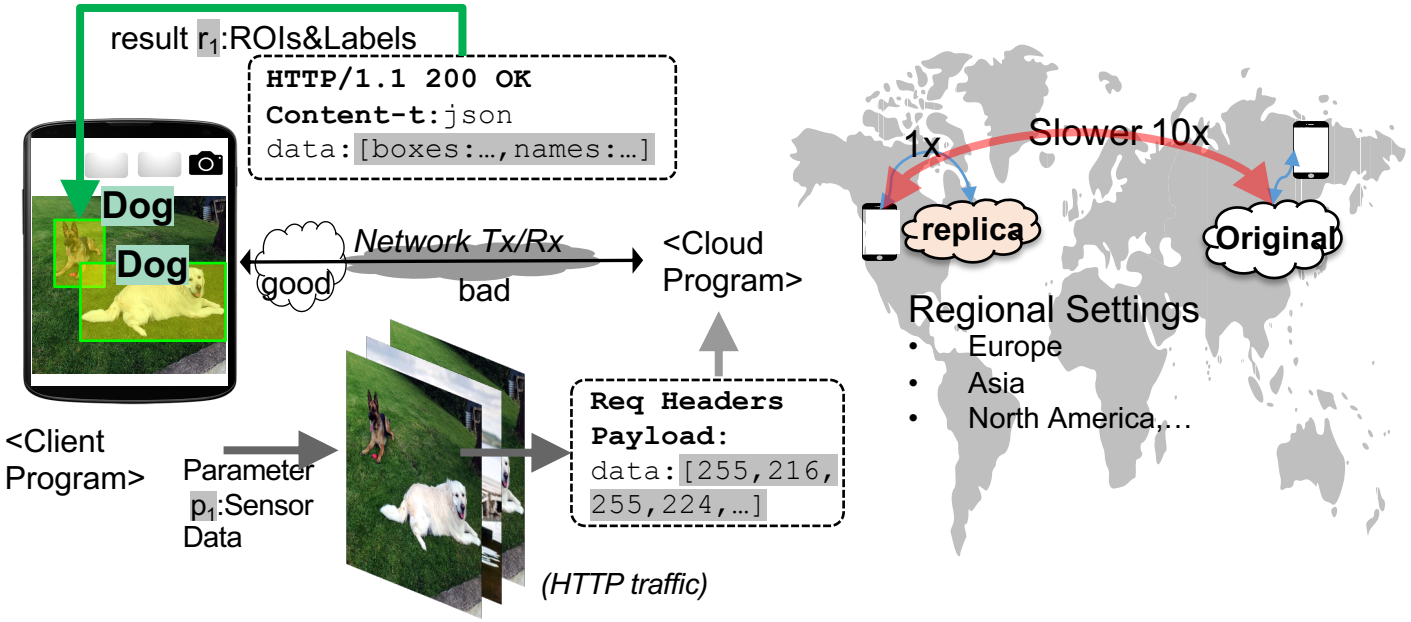




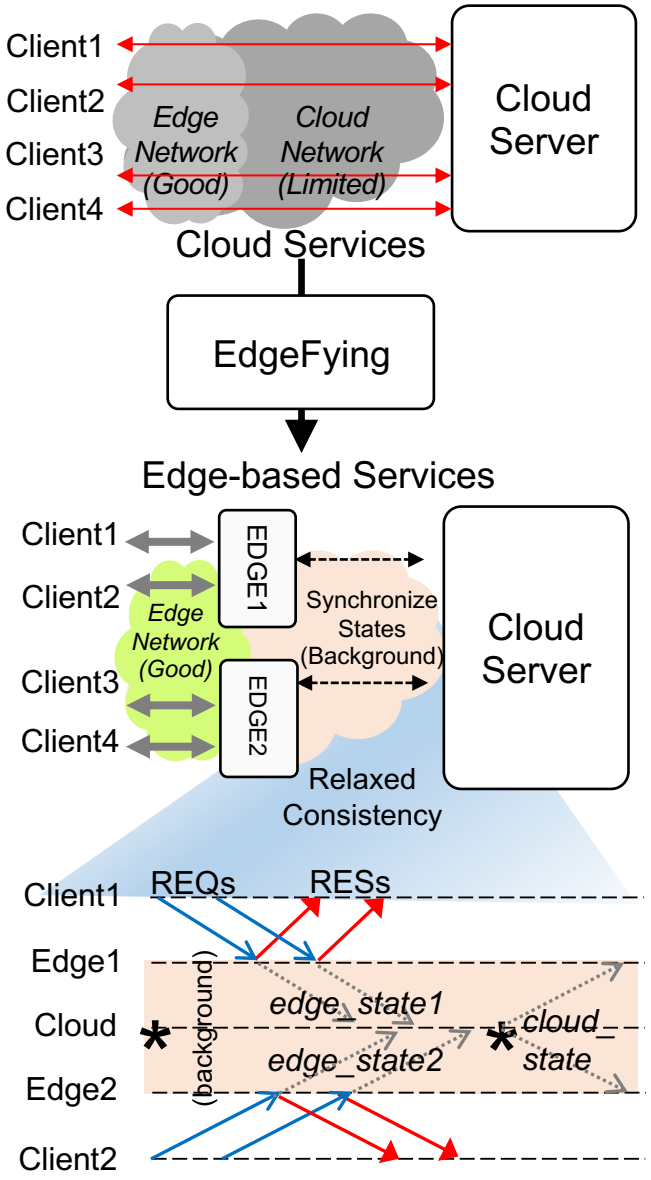
VIRGINIA TECH

Application 4: EdgeFy (Submitted)

- **Locality** of Cloud services, **Data Deluge** on Network Bottleneck
  - Replicating {$state_{init}$, ftn_{init}} of Cloud Service
  - Synchronizing States: Cloud and Edge Replicas
  - Correctness of Transformation: Isabelle HOL framework
  - Performance Compared to other Proxy Techniques



ex) Apps of ML (tf.js)



ex) Heroku: Cloud Platform

VIRGINIA TECH

Thank you!

Q & A

13

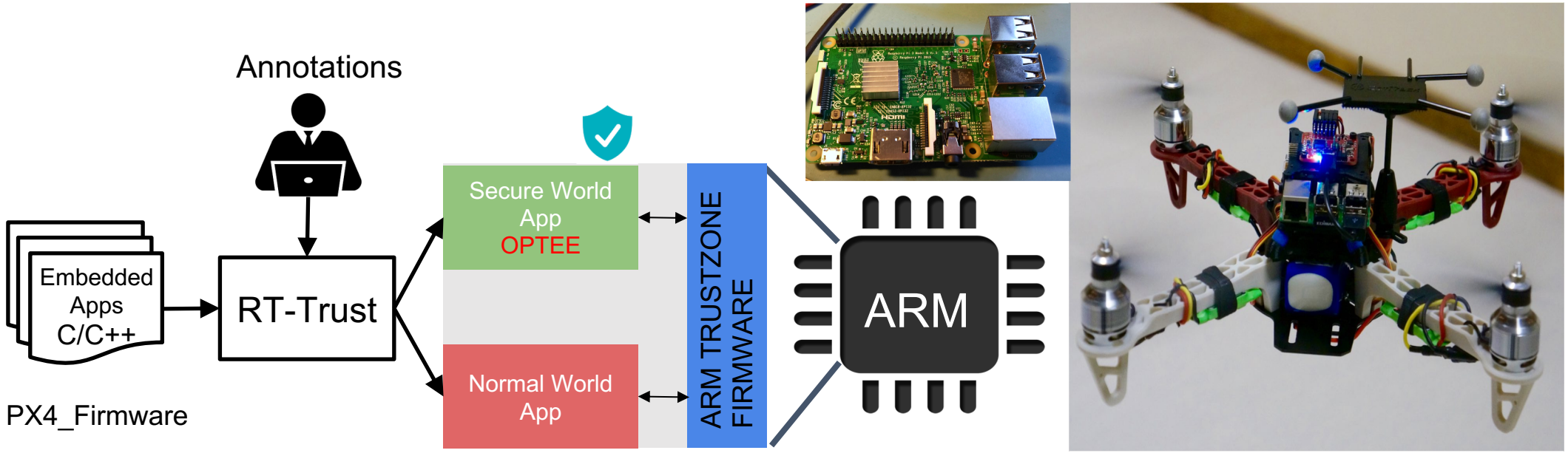
Appendix

- ✓ Other Projects
- ✓ System Design Experience

14

Appendix:
Refactoring Embedded Apps for Trusted Execution
[GPCE 2018], Best Paper Nomination at [COLA 2020]

- Programmers: **Annotating CPI** portions
- Partitioning C/C++ code into the regular and trusted parts
  - LLVM/Clang based Analysis/Refactoring, OPTEE (SGX in [COLA 2020])



ex) PX4_Firmware

VIRGINIA TECH

Appendix:

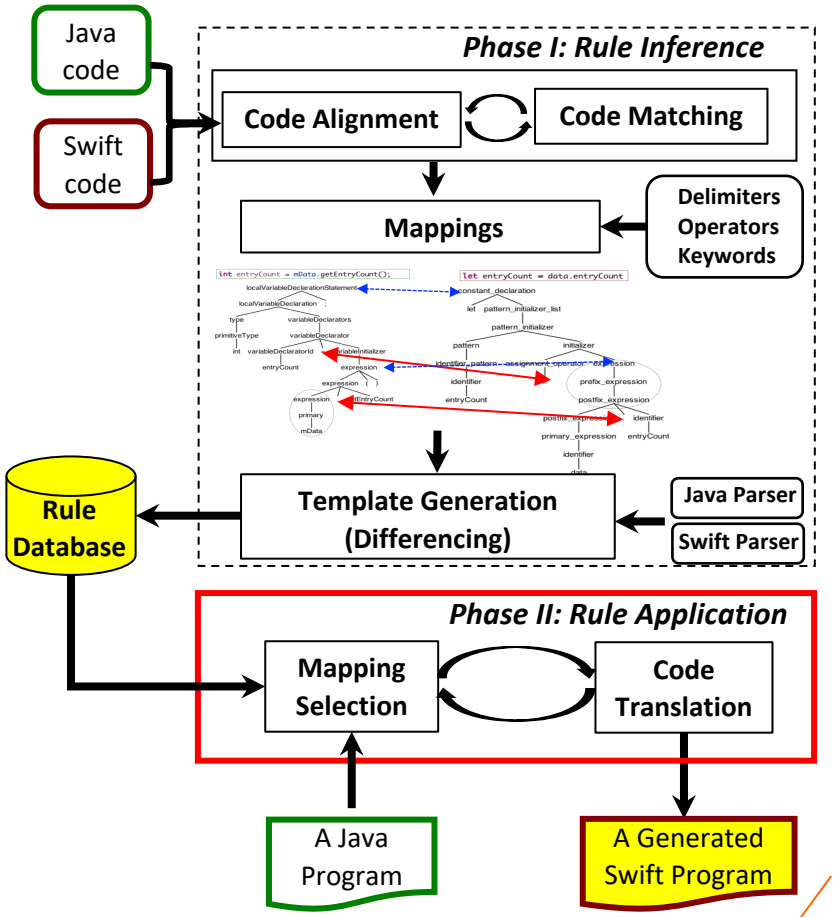
Learning Translating Rules/APIs from Cross-platform Apps

Best Paper Nomination at [MobileSoft 2018]

```
//PieChart.java
1 public class PieChart extends PieChartBase<PieData>
{...
2   private void calcAngles() { ...
3     int entryCount = mData.getEntryCount();
4     int cnt = 0;
5   for(int i = 0; i < mData.getDataSetCount(); i++){
6   IPieDataSet set = mData.get(i);
7   …}}}
```

```
//PieChartView.swift
public class PieChartView: PieChartViewBase
{…
  private func calcAngles() { ...
    let entryCount = data.entryCount
    var cnt = 0
    for i in 0 ..< data.dataSetCount {
      let set = data[i]
    ...}}}
```

No.	Java Syntax type	Swift Syntax type	Java template	Swift template
1	typeDeclaration	class_decl	public class $p10 extends $p11 {...}	public class $p10: $p11{...}
2	classBodyDecl	function_decl	private void $p20() {...}	private func $p20() {...}
3	**localVarDeclStmt**	**cnst_decl**	**$p30 $p31 = $p32;**	**let $p31 = $p32**
4	expression	expression	$p33.getEntryCount()	$p33.entryCount
5	**localVarDeclStmt**	**var_decl**	**$p40 $p41 = $p42;**	**var $p41 = $p42**
6 7	statement expression	for_in_stat expression	for($p50 $p51 = $p52; $p51 < $p53; $p51++) {...} $p54.getDataSetCount()	for $p51 in $p52 ..< $p53 {...} $p54.dataSetCount
8 9	statement expression	cnst_decl expression	$p60 $p61 = $p62; $p63.get($p64)	let $p61 = $p62 $p63[$p64]

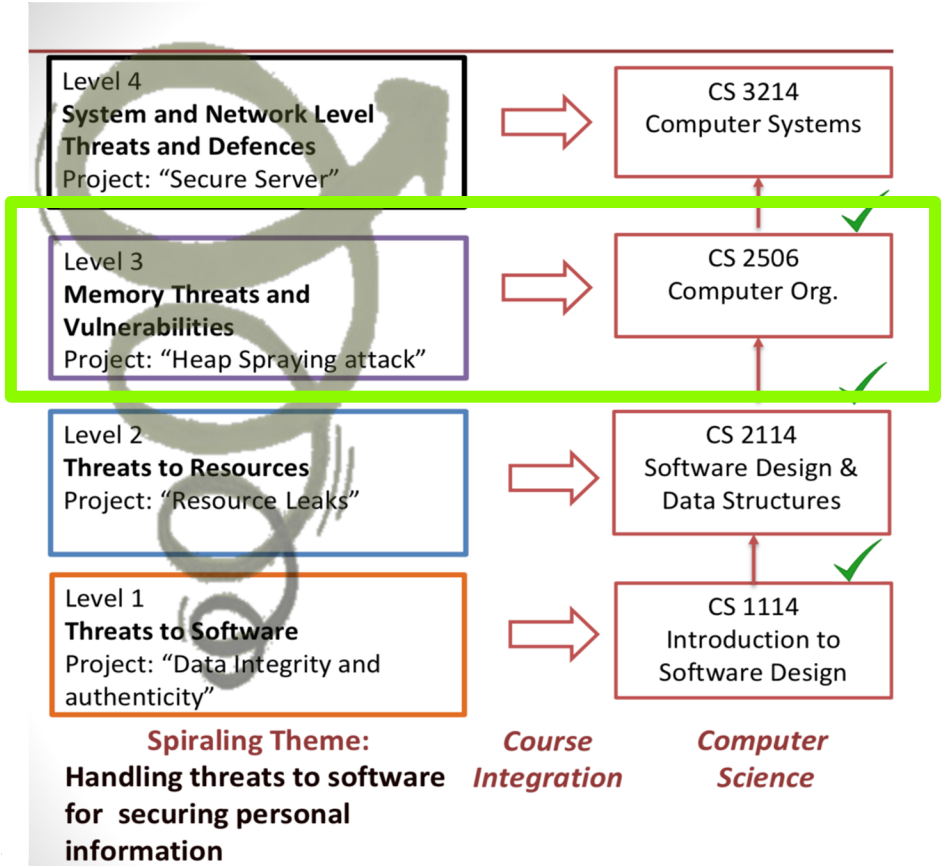


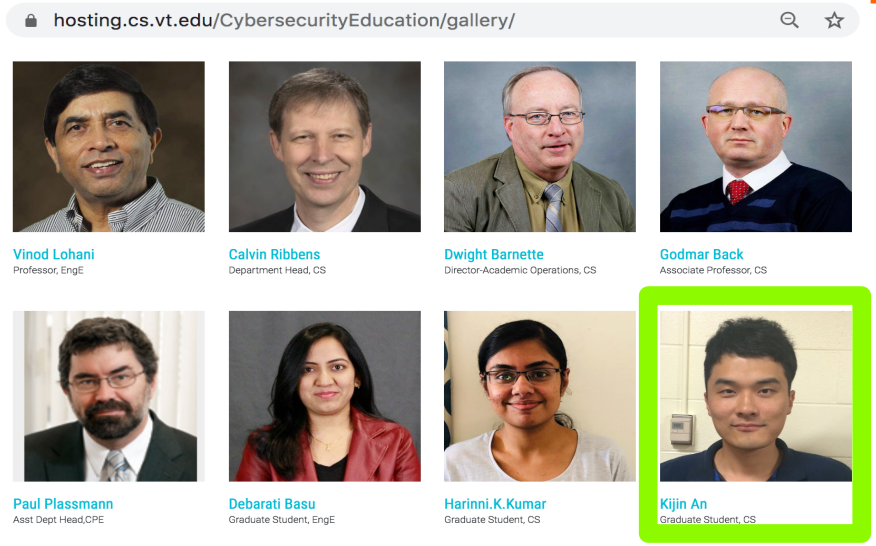
VIRGINIA TECH

Appendix: Understanding Heap Spraying Attacks

User Study (Pre/Post Survey): IRB, 540 undergrads for 3 Semesters

- NSF-funded project for increasing CyberSecurity-related education in CS and ECE core courses at Virginia Tech
- I developed the Level 3: *Understanding Heap Spraying Attacks*

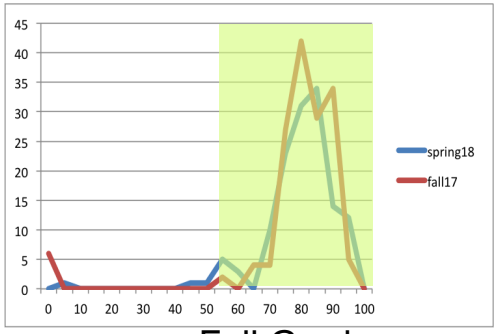




Very Good Feedback from Students

"This assignment was very challenging, but a lot of fun too!"

Students' Achievements



Full Grades

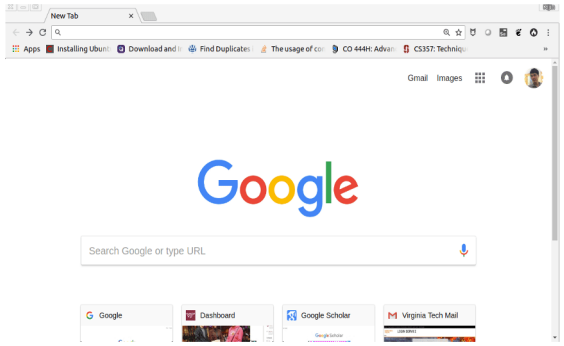
stewes36@vt.edu

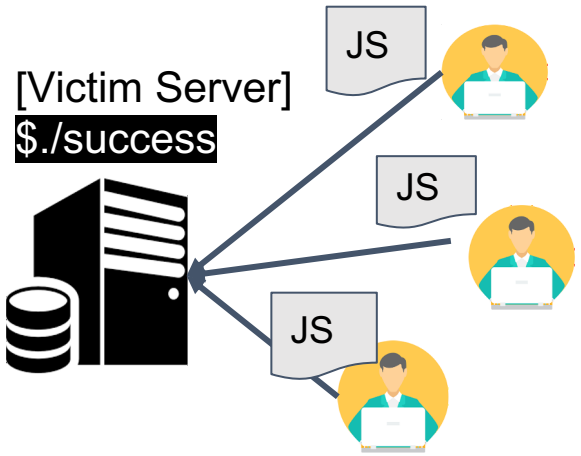
VIRGINIA TECH

Appendix: Understanding Heap Spraying Attacks

User Study (Pre/Post Survey): IRB, 540 undergrads for 3 Semesters

- Next Level of "**Attack Lab**"
- Systems: Victim Server, Grader
  :Executing and Evaluating Programs
- Extension of JavaScript Engine: V8



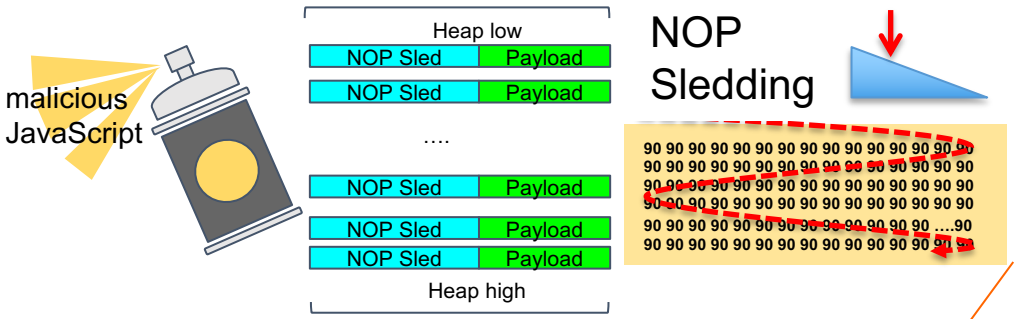


3. Submitting JS Code to Attack Server

1. Crafting Assembly Code for a system call

```
./sample: file format elf64-x86-64
Disassembly of section .text:
0000000000400078 <start>:   "Payload"
400078:    48 c7 c0 01 00 00 00   mov $0x1,%rax
40007f:    48 c7 c7 01 00 00 00   mov $0x1,%rdi
400086:    48 8d 35 19 00 00 00   lea 0x19(%rip)…
40008d:    48 c7 c2 0e 00 00 00   mov $0xe,%rdx
400094:    0f 05                  syscall
400096:    48 c7 c0 3c 00 00 00 00   mov $0x3c,%rax
40009d:    48 c7 c7 00 00 00      mov $0x0,%rdi
4000a4:    0f 05                  syscall
```

2. Heap Spraying with JS Code



VIRGINIA TECH

Appendix: Timelines

2003-2007

서울시립대학교 UNIVERSITY OF SEOUL

ECE B.E (4년)
인공지능 Lab, 이재호 교수

- Undergrad Intern(2년)
- Agent System (Robotics), Software Engineering

2009

Value Creating University POSTECH POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

M.S (2년)
MCNL Lab, 송황준 교수

- **Networking**, **Multimedia** (codec)
- WLAN, Zigbee
- 국제논문 4편

2012

SK telesys

Assistant Manager
시스템 SW 개발
(3년 4개월)

- **전문연구요원**
- WiFi/중계기 시스템 및 원격관리 툴 개발
- WiFi/IPPBX 비지니스 function 개발
- 와이브로, RF/Optic Repeaters

2015

KIST 한국과학기술연구원 Korea Institute of Science and Technology

Researcher (2년 10개월)
Robotics Research,
최종석 책임연구원

- **로봇서비스**를 위한 **클라우드 시스템개발**(ML)
- 50억 정부과제 실무 (개발/관리) 총괄
- 국제논문: Full Paper 3, Short Paper 7

2015.8 2021.5

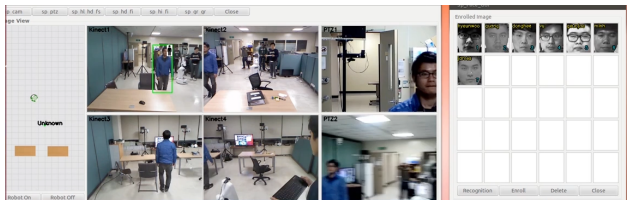
VIRGINIA TECH

Computer Science, PhD,
Software Innovations Lab, Eli Tilevich

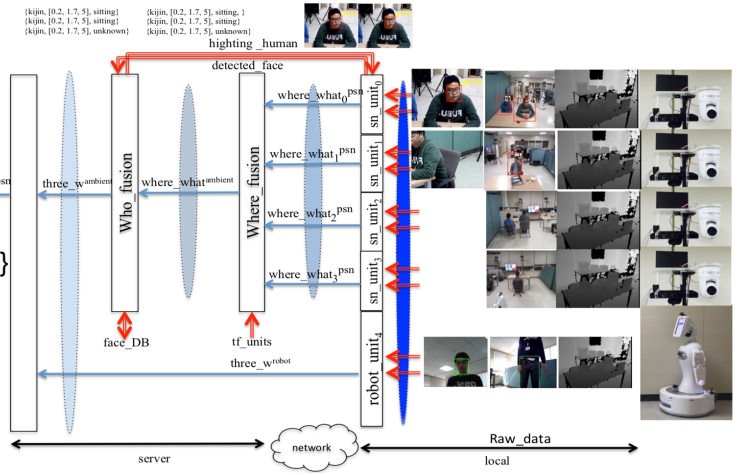
- Software Engineering
- Distributed Systems (ML Apps)
- 국제논문: Full 7, Short Paper 2
- Best Paper 1회, Nomination 2회

VIRGINIA TECH

Appendix: System Design /Research Experience in KIST

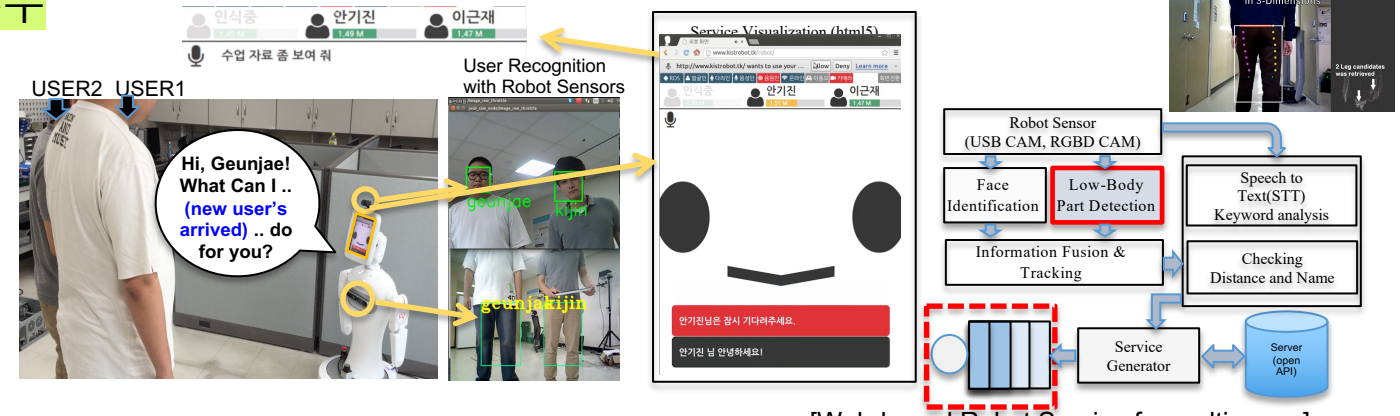


다수인원의
{위치,신원,행동}



[Cloud-based Distributed Systems]:
{Where,Who,What} Fusion for multi-users

- 프로젝트 내용: Cloud-based **Distributed Systems** for real-time Robot Services
- **총실무담당자: 시스템 설계 및 개발. Leading Project 1차년도 ~ 3차년도/총 5년**
  - 1년/2년/3년 평가결과: 모두 우수
    - 종료후 이달의 산업기술상 (PI awarded)
  - 해외컨퍼런스: Full 3, Short 7
  - Web/Machine Learning

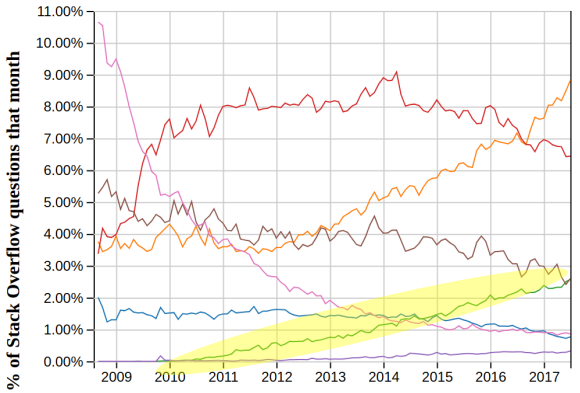


[Web-based Robot Service for multi-users]

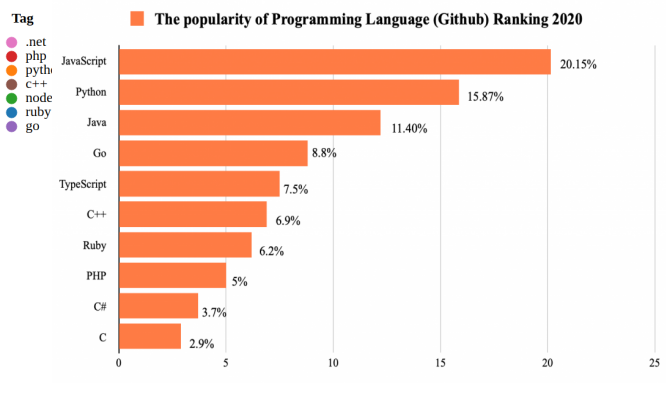
VIRGINIA TECH

Appendix: Applicability of Client Insourcing

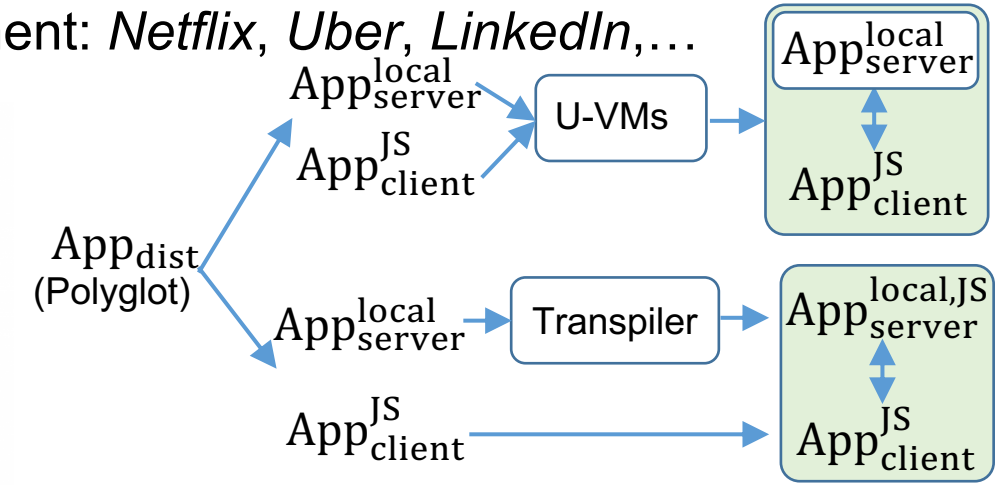
- **Subject**: Full-Stack JavaScript apps (Node.js): Popular in Backend and Open Source
  - Backends follow a lower load time development: *Netflix*, *Uber*, *LinkedIn*,…



[Trend in JS Backend Developments]

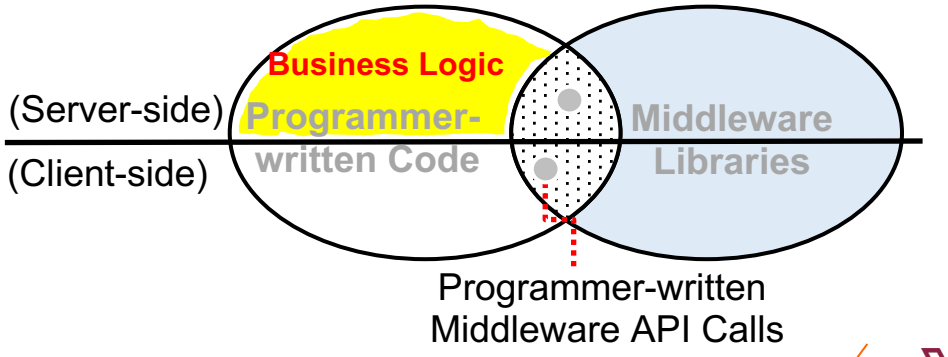


[Trend in Github Repositories]



[Applicability for Polyglot apps]

- **RESTful HTTP Protocols**
- **Insourcing Business (Application) Logic only**
  - What else? *Failure/Exception handling Logics*
- **Server State Isolations/Replications**
  - Database with **SQL**, **Files**, and **global variables**



VIRGINIA TECH