

# Efficient Microservices Using D

by Vijay Nayar <[madric@gmail.com](mailto:madric@gmail.com)>

DConf 2022: August 1st

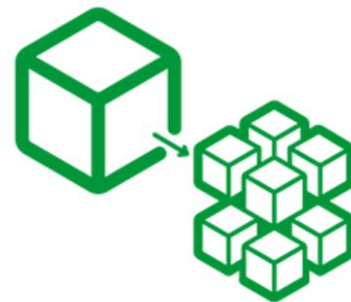
# Overview

- Microservice Concepts
- Unexpected Places for System-Level Languages
- Why D for Microservices?
- Putting it into Practice
  - A Practical Problem
  - System Architecture
  - Building in the Cloud
  - Deploying in the Cloud
- Concluding Remarks

# Microservice Concepts

---

# What is a Microservice?



A brief bit of context.

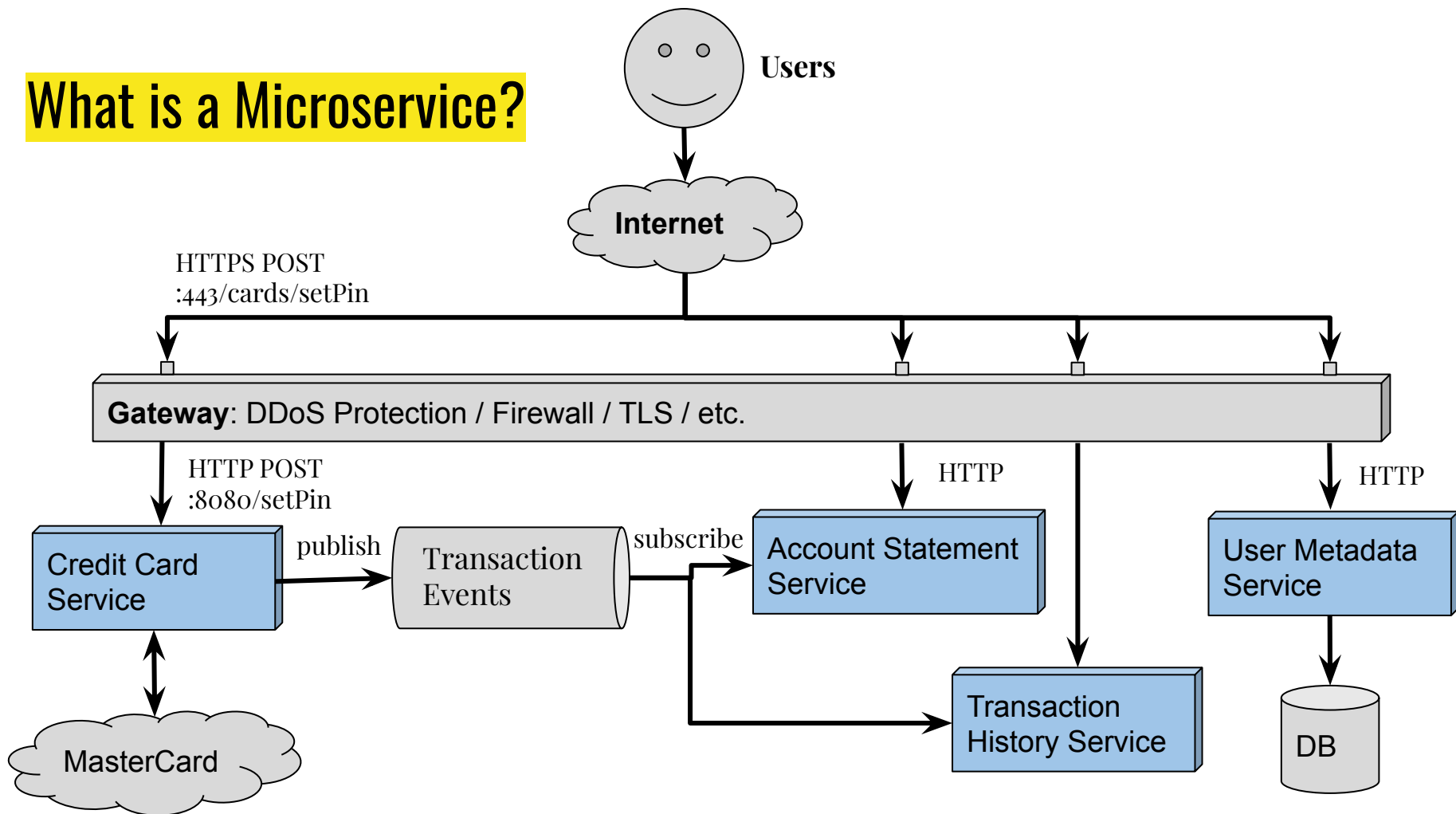
Microservices refer to services that constitute a microservice architecture.

A microservice architecture is a variant of service-oriented architecture, which separates an application into loosely coupled software services.

These services communicate with each other via lightweight protocols:

- Synchronous: REST using HTTP/JSON, gRPC, etc,
- Asynchronous: Apache Kafka, RabbitMQ, AWS SQS, etc.

# What is a Microservice?



## Why bother?

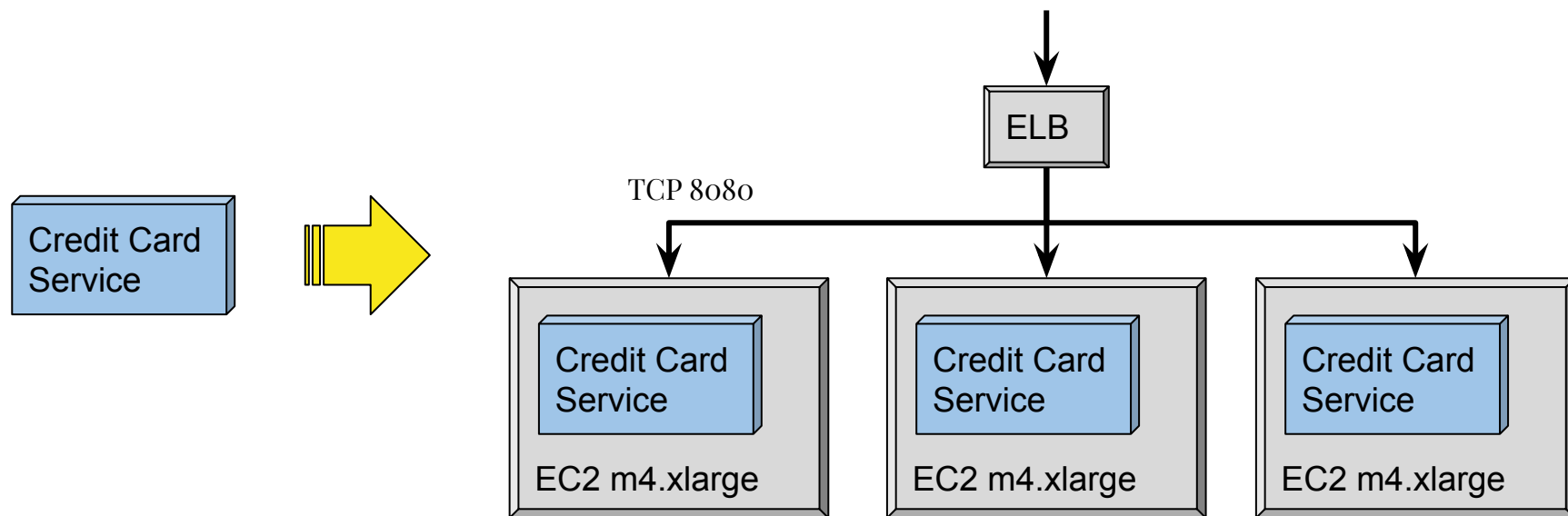
- Allows different services to use different technologies
- De-couple work so that teams of people can work independently
- Define clear contracts that help system development and testing
- Isolate changes related to scaling/performance to where they are needed
- Make it easier to deploy more frequent and smaller changes

Such architectures are now the norm in many industries.



# Microservices and Their Costs

Beyond the interfaces, scaling and deployment are important factors.

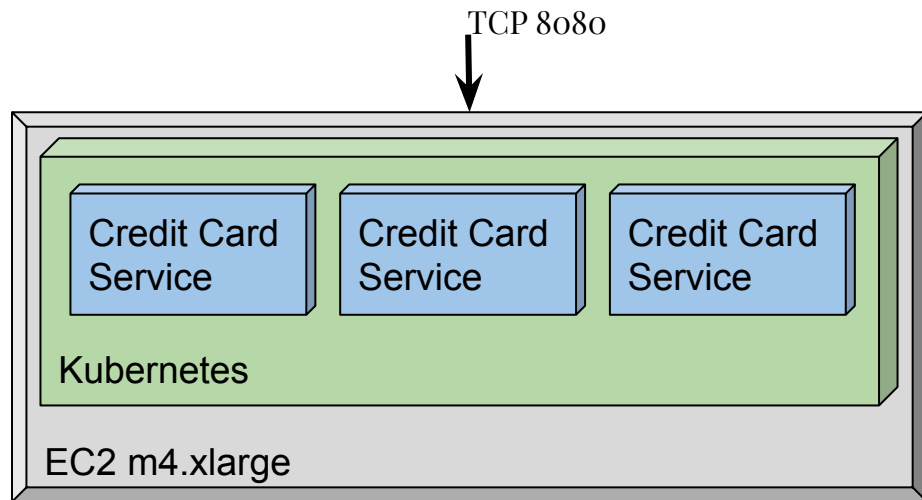


\*Each m4.xlarge costs \$0.20 per hour (on-demand), i.e. \$144.00 per month.

# Cutting Costs with Container Orchestration

Low efficiency deployments were enough for first-to-market services in the 2000's and 2010's. In the 2020's, competition and profitability are increasingly important.

Container orchestration systems like [Kubernetes](#), [AWS ECS](#), or [OpenShift](#) help configure and deploy multiple services per computer.



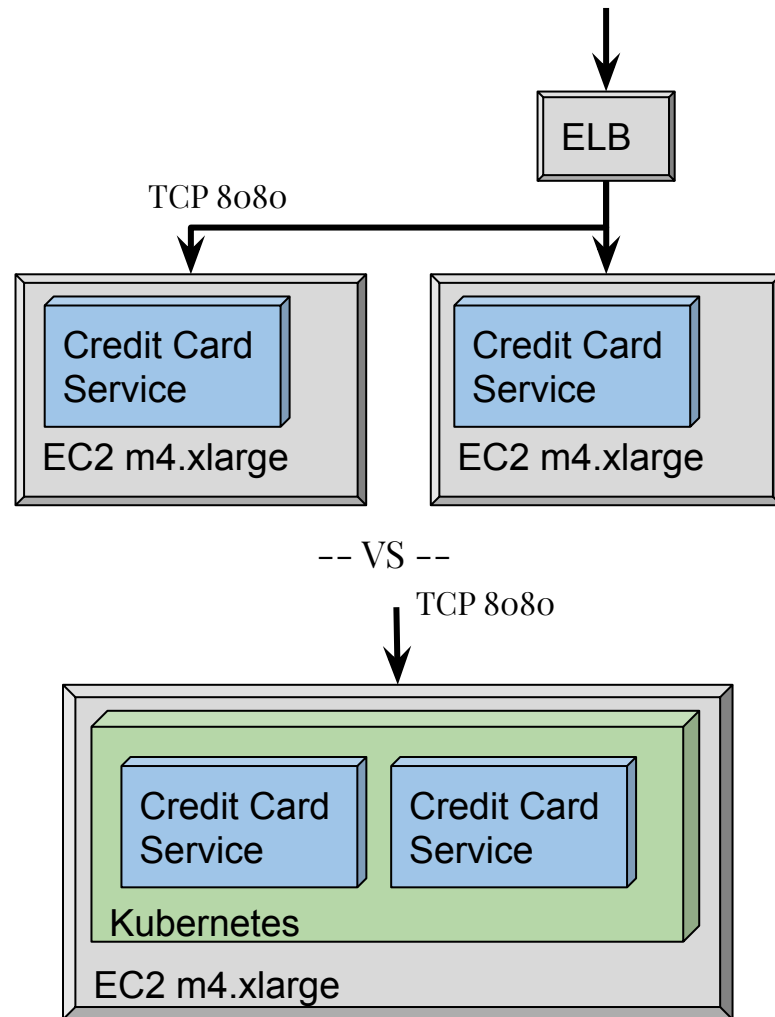


# Microservices and Their Costs

Many types of costs per server:

- Hourly Pricing for CPU/RAM
- Hourly Disk Space per GB
- Network Bandwidth per GB
- Public IP Addresses
- Load Balancer Connections

Container orchestration can help with costs depending on pricing.



# Cutting Costs with Container Orchestration

Savings exist for multiple mostly-idle services that need  $<1$  CPU.

However, savings for fully loaded servers is limited...

E.g. 2 smaller servers cost as much as 1 bigger server.

Even so, there are savings for:

- Load Balancers
- Public IPs
- Low-use services

Instance name ▲	On-Demand hourly rate ▼	vCPU ▼	Memory ▼
c7g.medium	\$0.0361	1	2 GiB
c7g.large	\$0.0723	2	4 GiB
c7g.xlarge	\$0.1445	4	8 GiB
c7g.2xlarge	\$0.289	8	16 GiB

# A Place for System-Level Languages

---

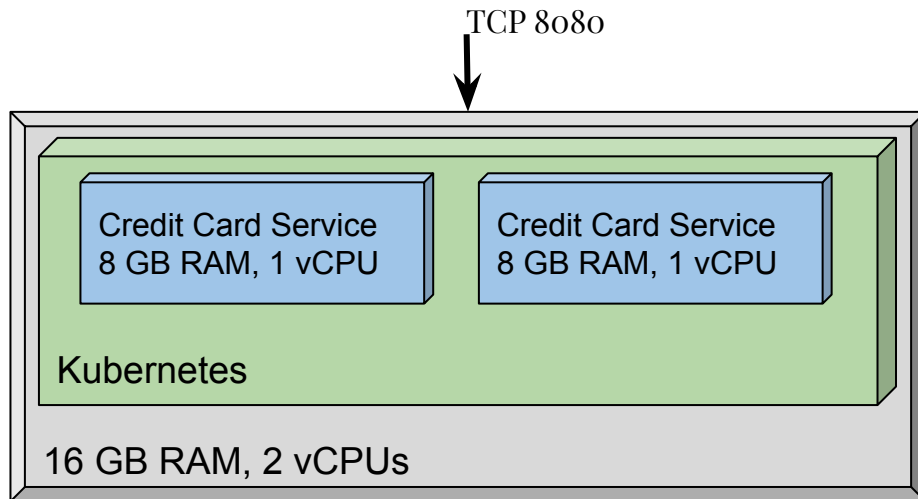
# Get what you pay for, Pay for what you get

Cloud-computing platforms like AWS have become quite efficient.

Two 1-CPU computers cost roughly as much as one 2-CPU computer.

For big companies with 10-100K USD costs per month, resources needs are important.

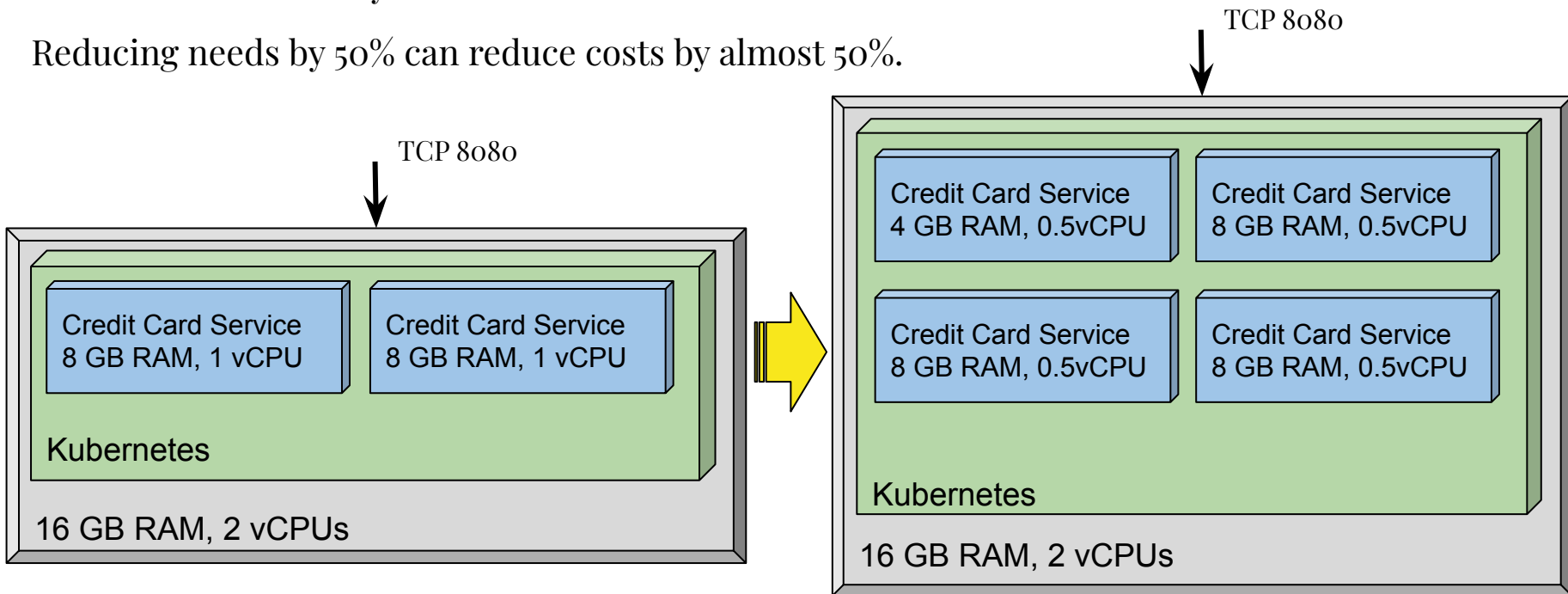
For small companies with a small budget, resources needs are important.



# Waste Not, Want Not

The most effective way to cut costs is to need less.

Reducing needs by 50% can reduce costs by almost 50%.



# Lower Resource Usage with System-Level Languages

Higher level programming languages are easy to work with, but aren't free:

- **Python, JavaScript, Ruby:** An interpreter must exist to execute the code, reducing performance and increasing memory usage.
- **Java, Scala, C#, Erlang:** Code is compiled for use by a virtual machine, which is more efficient than an interpreter, but still not free.

System-Level languages compile directly to machine code with little overhead:

- **D, C++, C, Rust, Go, Ada, Fortran**

# D vs. Java

A comparison between a minimal web-server using:

- Java / Spring

- Binary Size: 16.8 MB
- Resident Memory: 280 MB
- Virtual Memory: 10,000 MB
- Transactions/second: 5,200 trans/sec

- D / Vibe.d

- Binary Size: 3.8 MB
- Resident Memory: 11 MB
- Virtual Memory: 510 MB
- Transactions/second: 11,000 trans/sec



Sönke Ludwig has done fantastic work, check it out at <https://vibed.org>

# Reasons Not to use System-Level Languages

Typical reasons to not use system-level languages include:

- Higher complexity and development costs.  
(Still true, but powerful libraries reduce this barrier.)
- Lack of libraries for certain domains, e.g. AWS, Kafka, Web, ML, etc.  
(As important as language selection is the libraries for that language.)
- Having to compile separately for multiple platforms.  
(With cloud-infrastructure, this has become far less of an issue.)
- Long compilation times and slower development speeds.  
(Varies considerably from language to language.)



# Why D for Microservices?

---



# Advantages of D for Web Services

- **Lower Costs:** D's compiled high performance reduces hardware needs.
- **Quick Training:** D's C-family syntax speeds up onboarding for developers already familiar with C++, JavaScript, Python, Java, etc.
- **Incorporate Libraries:** Because D's multi-paradigm feature-rich approach, libraries from other programming languages can be mechanically converted.
- **Fast Development:** D, like Java, has garbage collection. The latency is acceptable and it simplifies development.
- **Error Detection:** D's integration of unit tests and pre/post-conditions makes it easier to write tests, encouraging better test coverage.

# A Note on other System-Level Languages

A quick note on other languages:

- **Rust:** ▲ High performance, ▼ unfamiliar syntax, ▼ must re-design libraries.
- **Go:** ▲ Good performance, ▼ unfamiliar syntax, ▼ limited features
- **C++:** ▲ High performance, ▼ difficult development
  
- **D:** ▲ High performance, ▲ familiar syntax, ▲ libraries convert mechanically

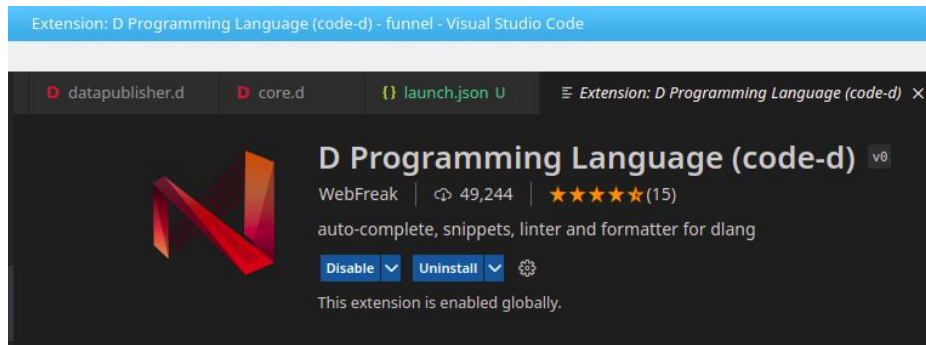
# Challenges of Using D for Web Services (compared to Java)

- **Lack of IDE Integration:** Java has powerful IDE support ([IntelliJ IDEA](#)) and powerful debugging tools.  
(D [debuggers](#) lack features like code execution.)
- **Lack of Profiling/Memory Tools:** Java has powerful inspection tools like [VisualVM](#). (D [profiling](#) is complex, tools often not maintained.)
- **Poor Library Coverage:** Frequently used libraries for Web Services are missing in D.
  - AWS SQS, SNS, Apache Kafka
  - Database ORM (e.g. PostgreSQL)
  - Rule Engines (e.g. [Drools](#))
  - Security (e.g. [Vault](#))

# Overcoming Challenges: IDE Integration

## IDE Integration:

- **VSCode: General-use IDE.**
  - **Extensions for D:** The user [WebFreak](#) has done a good job adding D support.
  - **Syntax Highlighting:** Looks great, responsive, welcoming.
  - **gdb, lldb:** **Partial support**
    - most expressions beyond primitives cannot be evaluated.
    - class members not displayed when debugging class function



datapublisher.d - funnel - Visual Studio Code

File Edit Selection View Go Run Terminal Help

RUN AND DEB... Debug

app.d

launch.json U

VARIABLES

> \_\_capture: <args>

> \_\_r3235: <unknown>

\_\_key3236: 0

> name: Object@\*0x7ffff7547d40

\_\_exception\_object: Could not expand v...

WATCH

CALL STACK

1:Thread 0x7ffff7639600... PAUSED ON STEP

funnel.service.datapublisher.DataPublish...

\_D4vibe4coreQf8setTimerFNb5Qu4time8Durati

\_D4vibe4coreQf11createTimerFNbNDFbNfZvi

\_D4vibe4core4task12TaskFuncInfo\_\_T3setTDI

BREAKPOINTS

app.d mouth/source 42

datapublisher.d mouth/source/fu... 38

datapublisher.d mouth/source/fu... 43

datapublisher.d mouth/source/fu... 45

mouth > source > funnel > service > datapublisher.d > DataPublisher > startTimer

38

39

40

41

42

43

44

45

46

47

48

49

50

setTimer(

750.msecs,

{

foreach (Name name; \_schemaRegistry.getSchemaNames()) {

// The data is now gone, make sure it gets sent.

ubyte[] dataBytes;

ushort count;

\_dataService.getAndClearData(name, count, dataBytes);

if (dataBytes.length == 0)

| continue;

auto bundle = new GenericDataBundle(name, count, dataBytes);

logDebug("Writing to stem: GenericBundle[name=%s, count=%d, dataByte

bundle.name, bundle.count, bundle.data.to!string);

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

Filter (e.g. text, lexclude)

43

2

43

45

ubyte[] dataBytes;

[Switching to thread 10 (Thread 0x7ffffdeffd700 (LWP 23817))](running)

Thread 1 "funnel-mouth" hit Breakpoint 4, funnel.service.datapublisher.DataPublisher.startTimer().\_\_lambda6() (\_\_ca

pture=0x7ffff7563060) at source/funnel/service/datapublisher.d:45

\_dataService.getAndClearData(name, count, dataBytes);

[Switching to thread 10 (Thread 0x7ffffdeffd700 (LWP 23817))](running)

p \_dataService

No symbol "\_dataService" in current context.

No symbol "\_dataService" in current context. (from interpreter-exec --thread 1 --frame 0 console "p \_dataService")

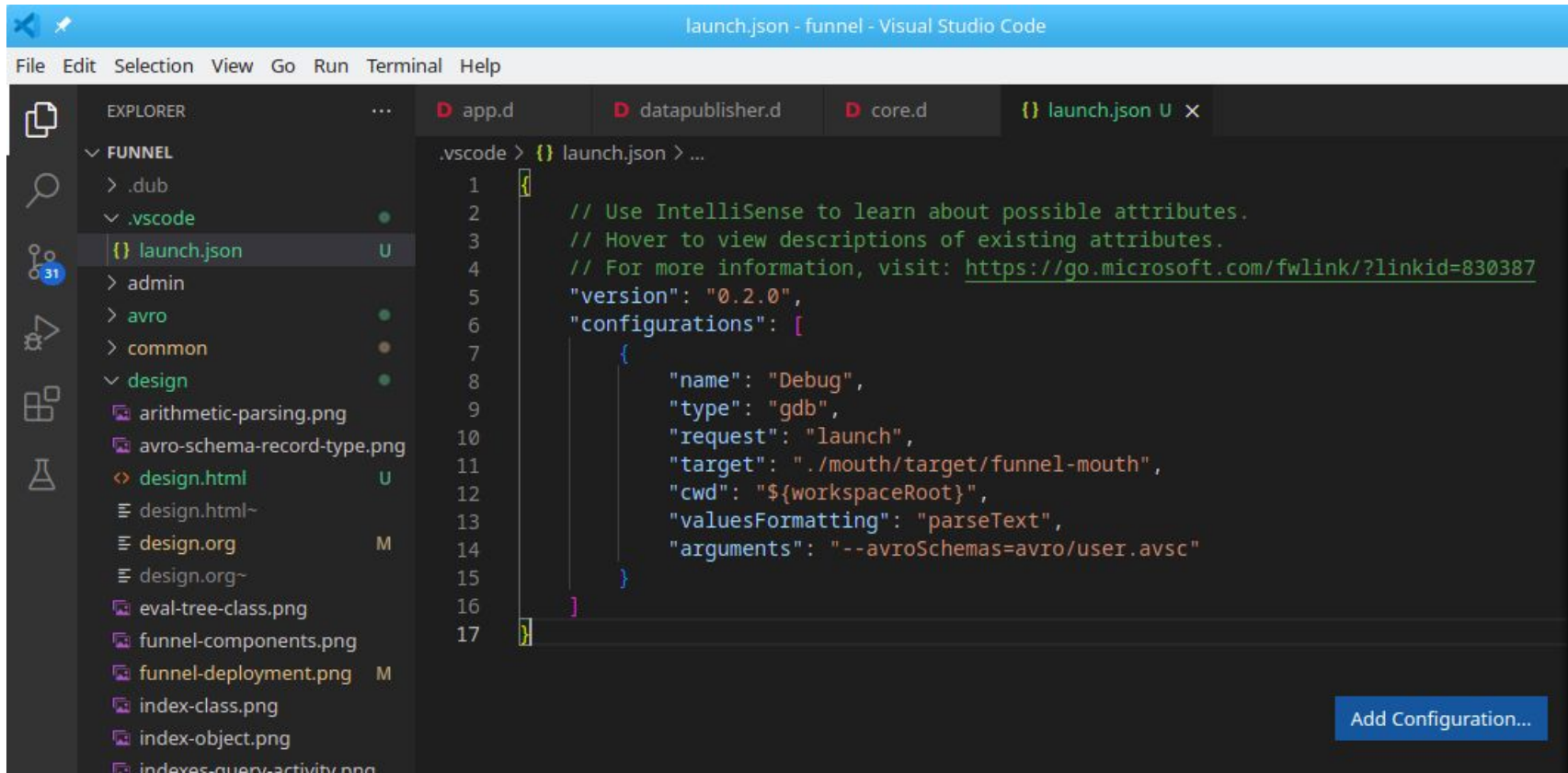
master\*

0 0 0

Debug (funnel) application debug dmd

Ln 46, Col 1 Spaces: 2 UTF-8 LF D

Debug and other configurations must be established by hand without auto-detection.

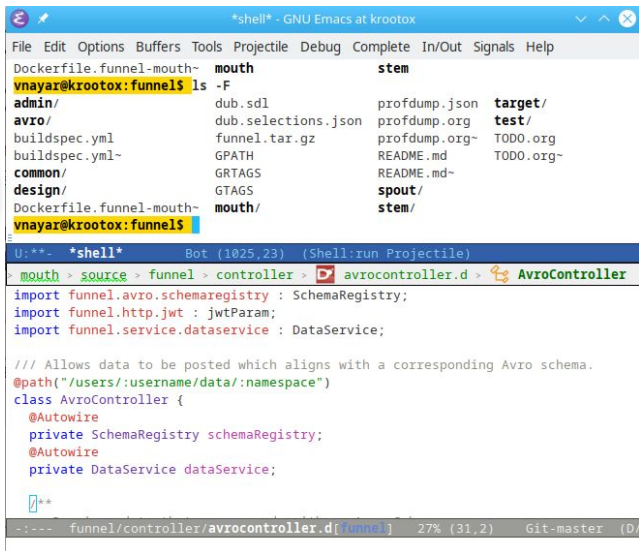
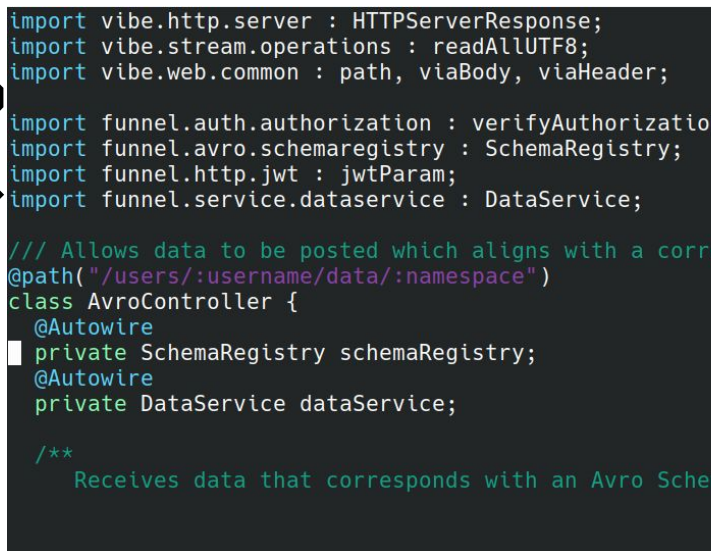


Despite existing configurations in dub, D IDE development is less "plug-and-play" than Java/Maven with IntelliJ or Eclipse.



# Old Fashioned Tools Still Work

If speed and low resource usage are more important than good looks, classic tools still continue to work:

A screenshot of an Emacs shell window titled '\*shell\* - GNU Emacs at krootox'. The window shows a directory listing of a project structure. The prompt is 'vnayar@krootox:funnel\$'. The directory listing shows a tree structure with files like 'Dockerfile', 'funnel-mouth', 'admin/', 'avro/', 'buildspec.yml', 'buildspec.yml-', 'common/', 'design/', and 'Dockerfile.funnel-mouth-'. The prompt changes to 'vnayar@krootox:funnel\$' again. Below the listing, the prompt is 'U:~\*~ \*shell\* Bot (1025,23) (Shell:run Projectile)'. The prompt changes to 'mouth > source > funnel > controller > avrocontroller.d > AvroController'. The prompt changes to 'import funnel.avro.schema.registry : SchemaRegistry;'. The prompt changes to 'import funnel.http.jwt : jwtParam;'. The prompt changes to 'import funnel.service.dataservice : DataService;'. The prompt changes to '/// Allows data to be posted which aligns with a corresponding Avro schema.'. The prompt changes to '@path("/users/:username/data/:namespace")'. The prompt changes to 'class AvroController {'. The prompt changes to '@Autowired'. The prompt changes to 'private SchemaRegistry schemaRegistry;'. The prompt changes to '@Autowired'. The prompt changes to 'private DataService dataservice;'. The prompt changes to '/\*'. The prompt changes to 'funnel/controller/avrocontroller.d[funnel] 27% (31,2) Git-master (D/'.A screenshot of a Vim editor window showing a Java code file. The code is for an 'AvroController' class. The code includes imports for 'vibe.http.server', 'vibe.stream.operations', 'vibe.web.common', 'funnel.auth.authorization', 'funnel.avro.schema.registry', 'funnel.http.jwt', and 'funnel.service.dataservice'. The code also includes a comment '/// Allows data to be posted which aligns with a corresponding Avro schema.' and a path definition '@path("/users/:username/data/:namespace")'. The code defines a class 'AvroController' with two private fields: 'SchemaRegistry schemaRegistry' and 'DataService dataservice', both annotated with '@Autowired'. The code ends with a comment '/\*\*' and a description 'Receives data that corresponds with an Avro Sche'.



# Emacs also works with the "gdb" command (here with "gdb-many-windows")

File Edit Options Buffers Tools Projectile Gud Complete In/Out Signals Help

```
45         _dataService.getAndClearData(name, count, dataBytes);
```

```
(gdb) n
[Switching to thread 7 (Thread 0x7ffffdfff700 (LWP 28178))](running)
```

```
46         if (dataBytes.length == 0)
```

```
(gdb) p name
```

```
$1 = (avro.name.Name *) 0x7ffff7547d40
```

```
(gdb) p name.namespace
```

```
$2 = "funnel"
```

```
(gdb)
```

```
U:***- *gud-funnel-mouth* Bot (115,6) (Debugger:run [and-stepping-range] +1 Projectile)
```

```
mouth - source - funnel - service - datapublisher.d - DataPublisher - startTimer
```

```
enforce(nn_connect(sock, url) >= 0);
```

```
setTimer(
    750.msecs,
    {
        foreach (Name name; _schemaRegistry.getSchemaNames()) {
            // The data is now gone, make sure it gets sent.
            ubyte[] dataBytes;
            ushort count;
            _dataService.getAndClearData(name, count, dataBytes);
            if (dataBytes.length == 0)
                continue;
            auto bundle = new GenericDataBundle(name, count, dataBytes);
            logDebug("Writing to stem: GenericBundle[name=%s, count=%d, dataBytes=%s]",
                bundle.name, bundle.count, bundle.data.toString());
            ubyte[] bundleBytes = bundle.toBytes();
            int bytesSent = nn_send(sock, bundleBytes.ptr, bundleBytes.length, 0);
            if (bytesSent < dataBytes.length) {
                // TODO: Do something if the bytes could not be sent.
            }
        }
    }
);
```

```
U:***- funnel/service/datapublisher.d[funnel] 53% (46,0) Git-master (D// +1 Lens LSP[serve-d:9121] Projectile[funnel])
```

```
0 in funnel.service.datapublisher.DataPublisher.startTimer().__lambda6() of source/funnel/service/
1 in _D4vibe4coreQf8setTimerFNBsQu4time8DurationDFZvbZ9__lambda4MFNBNeZv of ../../.dub/packages/vibe-core-1.22.4/vibe-core/src
2 in _D4vibe4coreQf11createTimerFNBnFDfNBfZvZ1C6opCallMFNBnFSQCdQCbQCe5TimerZ9__lambda2MFNBnF of ../../.dub/packages/vibe-core-1.22.4/vibe-core/src
3 in _D4vibe4core4task12TaskFuncInfo__T3setTDFNBnFSQB8QB8QBt5TimerZvTQtZQBhMFKBiKQB8Z12callDe of ../../.dub/packages/vibe-core-1.22.4/vibe-core/src
4 in vibe.core.task.TaskFuncInfo.call() of ../../.dub/packages/vibe-core-1.22.4/vibe-core/src
5 in vibe.core.task.TaskFiber.run() of ../../.dub/packages/vibe-core-1.22.4/vibe-core/src
6 in core.thread.context.Callable.opCall()
7 in fiber_entryPoint
8 in ??
```

```
U:***- *stack frames of funnel-mouth* All (1,0) (Frames [thread 1] +1 Projectile)
```

Locals Registers

__Array_*	__r3235	<complex data type>
uint long long	__key3236	0
avro.name.Name *	name	0x7ffff7547d40
__Array_unsigned char	dataBytes	<complex data type>
ushort	count	0
funnel.avro.genericdatabundle.GenericDataBundle * volatile	bundle	0x7ffff75602c0
__Array_unsigned char	bundleBytes	<complex data type>
int	bytesSent	1442607376

```
U:***- *locals of funnel-mouth* All (1,0) (Locals: funnel.service.datapublisher.DataPublisher.startTimer().__lambda6)
```

```
source/app.d:38 _Dmain [0x555555a2785a]
JWT tokens are only validated if the environment variable JWT_PUBLIC_KEY or JWT_PUBLIC_KEY_URL
is set.
```

```
[main(---) INF] Parsing Avro Schema: ../avro/user.avsc
[main(---) INF] Parsing Avro Schema: ../avro/user.avsc
[main(---) INF] Connecting to stemSubUrl: tcp://127.0.0.1:5000
[main(---) INF] Parsing Avro Schema: ../avro/user.avsc
[main(---) INF] Please open http://127.0.0.1:8080/ in your browser.
[vibe-2(C+MH) INF] Parsing Avro Schema: ../avro/user.avsc
[vibe-2(C+MH) INF] Listening for requests on http://[::]:8080/
[vibe-2(C+MH) INF] Listening for requests on http://0.0.0.0:8080/
[vibe-0(W5nj) INF] Listening for requests on http://[::]:8080/
[vibe-0(W5nj) INF] Listening for requests on http://0.0.0.0:8080/
[vibe-4(Gwlc) INF] Listening for requests on http://[::]:8080/
[vibe-4(Gwlc) INF] Listening for requests on http://0.0.0.0:8080/
[vibe-1(gGuM) INF] Listening for requests on http://[::]:8080/
[vibe-1(gGuM) INF] Listening for requests on http://0.0.0.0:8080/
[vibe-5(Hf39) INF] Listening for requests on http://[::]:8080/
[vibe-5(Hf39) INF] Listening for requests on http://0.0.0.0:8080/
[vibe-6(fW1b) INF] Listening for requests on http://[::]:8080/
```

```
U:***- *input/output of funnel-mouth* 75% (31,65) (Inferior I/O:run +1 Projectile)
```

Num	Type	Disp	Enb	Addr	Hits	What
1	breakpoint	keep	y	<MULTIPLE>	10	in unknown
2	breakpoint	keep	y	<MULTIPLE>	10	in unknown
3	breakpoint	keep	y	0x00005555555a27949	9	in main().__lambda3(shared(poodinis.contai
4	breakpoint	keep	y	0x00005555555a81331	2	in funnel.service.datapublisher.DataPublisher.

```
ner.DependencyContainer), ushort) of source/app.d:47
startTimer().__lambda6() of source/funnel/service/datapublisher.d:45
```

```
U:***- *breakpoints of funnel-mouth* All (1,0) (Breakpoints +1 Projectile)
```

# Overcoming Challenges: Profiling Tools

## Profiling Tools: Valgrind

- Running through Valgrind (D Wiki: [Other Dev Tools](#))
  - Run program through Valgrind, which has a smaller performance impact.  
`$ valgrind --tool=callgrind --collect-jumps=yes  
--callgrind-out-file=callgrind_out  
./mouth/target/funnel-mouth -- --avroSchemas=avro/user.avsc`
  - Demangle D symbols to make them human readable:  
`$ ddemangle callgrind_out > callgrind_out.demangle`
  - View results using kcacheGrind.  
`$ kcacheGrind callgrind_out.demangle`

File View Go Settings Help

Open... &lt; Back &gt; Forward ^ Up % Relative Cycle Detection Relative to Parent &lt;&gt; Shorten Templates Instruction Fetch

Flat Profile

Search: Search Query ELF Object

Self	ELF Object
94.24	funnel-mouth
3.40	libgcc_s.so.1
1.50	libc-2.31.so
0.27	ld-2.31.so
0.74	libstdc++.so.6

Incl.	Self	Called	Function
35.76	0.38	79 872	const pure nothrow @nogc ulong[2] vi...
34.29	0.45	92 673	pure nothrow @nogc @trusted ubyte[1...
33.07	1.73	278 036	pure nothrow @nogc @trusted void std...
31.34	6.21	92 673	pure nothrow @nogc void std.digest.md...
30.33	0.09	79 872	pure nothrow @nogc @safe ubyte[16] st...
30.24	0.30	79 872	pure nothrow @nogc @safe ubyte[16] st...
9.34	1.65	8	@trusted void vibe.http.router.Match...
6.62	0.00	8	nothrow ulong core.internal.gc.impl.c...
6.61	0.00	8	nothrow void core.internal.gc.impl.con...
6.48	4.42	1 482 768	pure nothrow @nogc @safe void std.dig...
6.48	4.42	1 482 768	pure nothrow @nogc @safe void std.dig...
6.21	0.40	15 692	pure @safe immutable(char[]) std.json...
6.16	4.42	1 482 768	pure nothrow @nogc @safe void std.dig...
6.11	0.00	8	nothrow void core.internal.gc.impl.con...
6.09	6.09	2 358	nothrow scope void core.internal.gc.i...
6.01	4.42	1 482 768	pure nothrow @nogc @safe void std.dig...
5.88	0.00	8	nothrow int core.internal.container.tr...
5.88	0.00	8	nothrow int core.internal.container.tr...
5.87	0.00	2 043	nothrow int core.internal.container.tr...
5.87	0.00	2 043	nothrow int core.internal.gc.impl.con...
4.87	0.01	12 784	pure nothrow @nogc @safe ubyte[16] s...
4.85	0.05	12 784	pure nothrow @nogc @safe ubyte[16] s...
4.60	0.72	235 096	pure @safe char std.json.parseJSON(i...
4.16	0.01	2 043	nothrow int core.internal.container.tr...
3.79	3.79	5 931 072	pure nothrow @nogc @safe uint core.bi...

Looks like validating JWT tokens is costly!

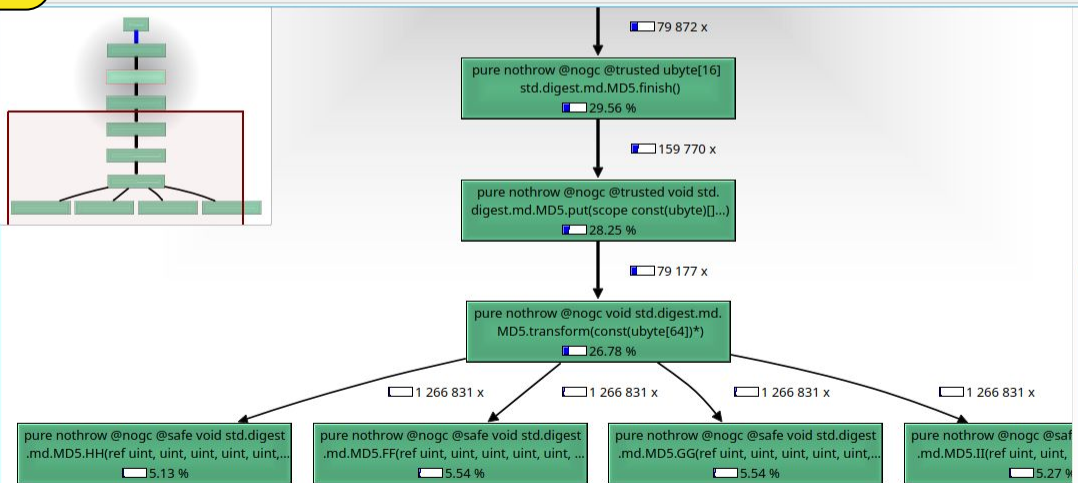
pure nothrow @nogc @safe ubyte[16] std.digest.md.md5Of(void[]).md5Of(void[])

Types Callers All Callers Callee Map Source Code

Event Type Incl. Self Short Formula

Instruction Fetch 30.33 0.09 Ir

Cycle Estimation 30.33 0.09 CEst = Ir



Parts Calleees Call Graph All Calleees Caller Map Machine Code

# Overcoming Challenges: Memory Tools

**Memory Profiling:** Compiler option "-profile=gc"

- May also be enabled via dub through "dub build --build=profile-gc"
- Memory allocations are output to "profilegc.log"
- Output is reasonably formatted (albeit with long names), e.g.:

```
bytes allocated, allocations, type, function, file:line
4096000          500      ubyte[] vibe.internal.array.BatchBuffer!(ubyte,
0LU).BatchBuffer.capacity
../../../../dub/packages/vibe-core-1.22.4/vibe-core/source/vibe/internal/array.d:580
662688          20709      std.array.Appender!(immutable(char)[]).Appender.Data
std.array.Appender!string.Appender.this /usr/include/dmd/phobos/std/array.d:3330
304000          1000      ubyte[] std.base64.Base64Impl!('-', '_',
'\x00').decode!string.decode /usr/include/dmd/phobos/std/base64.d:1339
```

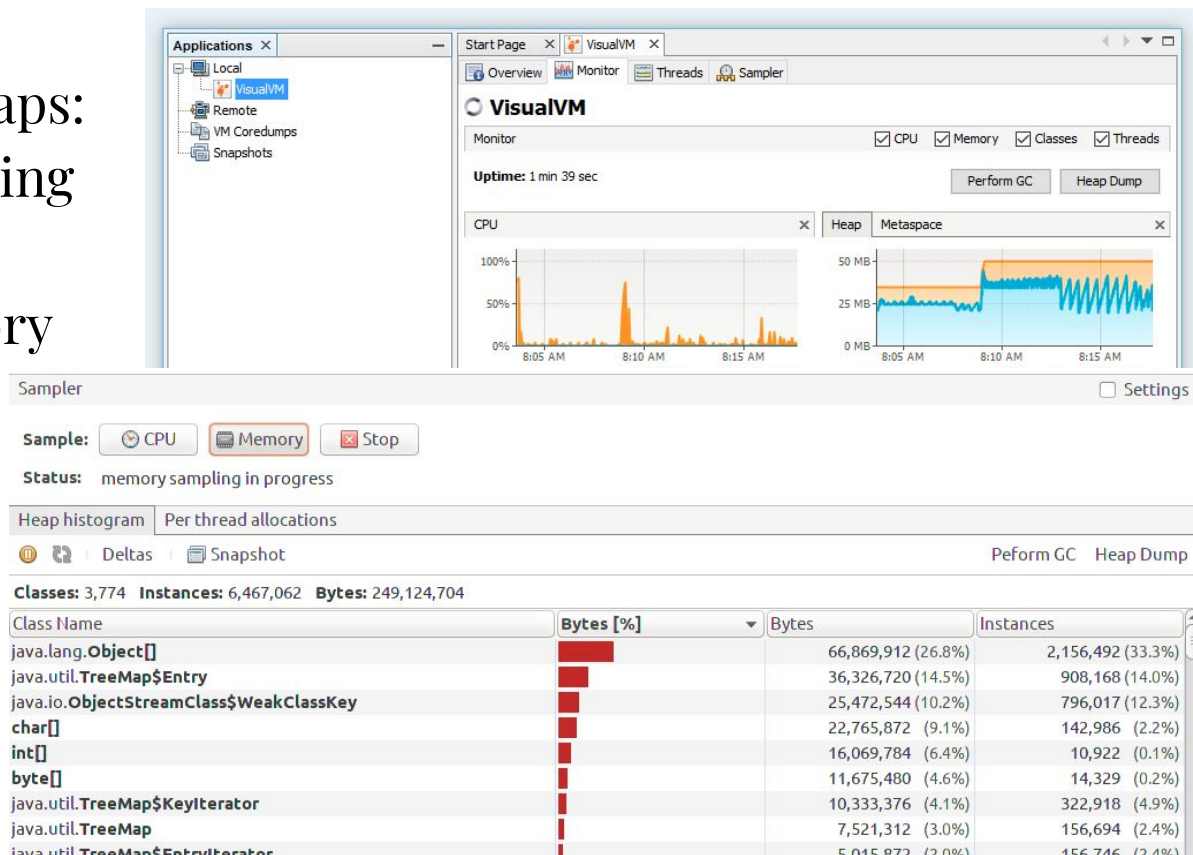


# Overcoming Challenges: Memory Tools

Available tools still have gaps:

- No visual aid for tracking memory allocations.
- No ability to do memory sampling

Languages like Java have such tools available ==>



# Overcoming Challenges: Profiling Tools to Avoid (for now)

## Profiling Tools: dmd -profile

- Theoretically this is built in: Compile using "-profile"
  - Compile using the "-profile" switch or the dub build profile, e.g.  
`$ dub build --build=profile`
- Difficult to use in practice.
  - Compilation frequently fails, e.g.  
`/usr/include/dmd/phobos/std/concurrency.d(2160,17): Warning: statement is not reachable`
  - Compilation using "-profile" produces hard to read results.
  - Performance is impacted considerably.
  - The [output format](#) is undocumented and contains no labels.

# Overcoming Challenges: Library Coverage

**Library Coverage:** Common Web Service libraries are missing in D...  
...but D is an amazing language for importing libraries.

D's multi-paradigm approach makes it extremely compatible with most other programming languages.

	D	Python	C++	Java	Go
Garbage Collection	✓	✓	✗	✓	✓
Pointer Manipulation	✓	✗	✓	✗	✗
Object Oriented	✓	✓	✓	✓	✗
Generic Programming	✓	✓	✓	✓	✓
Fast Prototyping	✓	✓	✗	~	✓

# Porting Libraries to D

D's familiar syntax makes it easy to quickly convert libraries from other languages with much less effort than usual.

For converting large libraries, a common strategy is:

1. Understand the chain of dependencies in the library, build outward starting from the deepest dependencies (typically utils).
2. Convert the library as literally and mechanically as possible, alterations can have cascading impacts which are hard to track.
3. Be sure to migrate tests and validate work as it moves along.
4. Refactor a single feature at a time.



# Porting Libraries to D

Typical range algorithms don't always work when converting libraries. Iterator-capable containers are often needed to port algorithms.

```
// Position the iterator at the first cell that overlaps or follows  
// "target", i.e. such that range_max() >= target.range_min().  
void SeekTo(const RangeIterator& target) {  
    it_.Seek(target.range_min());  
    // If the current cell does not overlap "target", it is possible that the  
    // previous cell is the one we are looking for. This can only happen when  
    // the previous cell contains "target" but has a smaller S2CellId.  
    if (it_.done() || it_.id().range_min() > target.range_max()) {  
        if (it_.Prev() && it_.id().range_max() < target.id()) it_.Next();  
    }  
    Refresh();  
}
```

# Porting Libraries to D

Another example algorithm that requires iterators to be properly converted. In D, such containers are simply not part of the standard library, but can be implemented, e.g. [btree.d](#) and [btree\\_map.d](#).

```
// TODO(ericv): Use a single iterator (iter_) below and save position
// information using pair<S2CellId, const S2ShapeIndexCell*> type.
auto next = new S2ShapeIndex.Iterator(
    _index, S2ShapeIndex.InitialPosition.BEGIN);
auto last = new S2ShapeIndex.Iterator(
    _index, S2ShapeIndex.InitialPosition.END);
last.prev();
if (next.id() != last.id()) {
    // The index has at least two cells. Choose a level such that the entire
    // index can be spanned with at most 6 cells (if the index spans multiple
    // faces) or 4 cells (if the index spans a single face).
    int level = next.id().getCommonAncestorLevel(last.id()) + 1;
    // Visit each potential top-level cell except the last (handled below).
    S2CellId last_id = last.id().parent(level);
```

# Putting it Into Practice

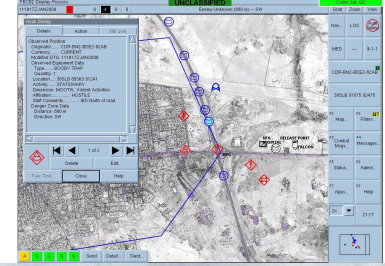
---



# A Practical Problem

What do the following industries have in common?

- Ride Hailing
- Micro Mobility
- IoT and Sensor Monitoring
- Military/Security Situational Awareness

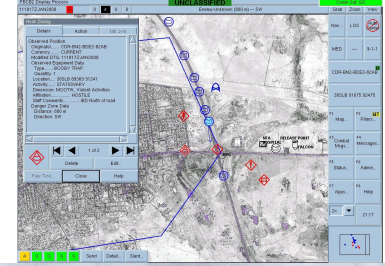


# A Practical Problem

What do the following industries have in common?

- Ride Hailing
- Micro Mobility
- IoT and Sensor Monitoring
- Military/Security Situational Awareness

**They all have very high volumes of data updates and queries in real-time.**



# By the Numbers

- 1,000 Taxis per City
- 50 Cities
- 1 position/status update per 10 seconds

**5,000 updates per second!**

- 10M people per city
- 5% of users look at the app per day
- each user session makes 5 queries

**1,500 queries per second!**

Numbers get big, fast.

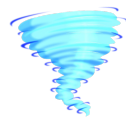


# Need for Speed

Customers need flexible queries... an RDS would fit.

Write performance needs to be extreme... a data-cache would fit.

Traditional solutions get VERY expensive as the system grows and grows, starting from a simple RDS, to multiple read-replicas, to sharding, to cache layers in Redis, etc.



[Funnel-Labs.io](https://funnel-labs.io) was created to tackle this challenge with a ready-made solution.

# A Brief Look

A simple JavaScript  
example client.

<http://demo-query.funnel-labs.io/>

Data format with  
coordinates to query.

demo-query.funnel-labs.io

## Funnel-Labs.io

Demo query playground.

### Settings

The demo uses a [Schema](#) to define formats for uploading and querying.

[View Data Schema](#)

### Data Generation

Create fake data and send it to Funnel-Mouth.

ID Base:

ID Count:

Latitude:

Longitude:

[Generate Data Here](#)

### Query Templates

Set the query from a template value.

[Within 20km of Here](#)

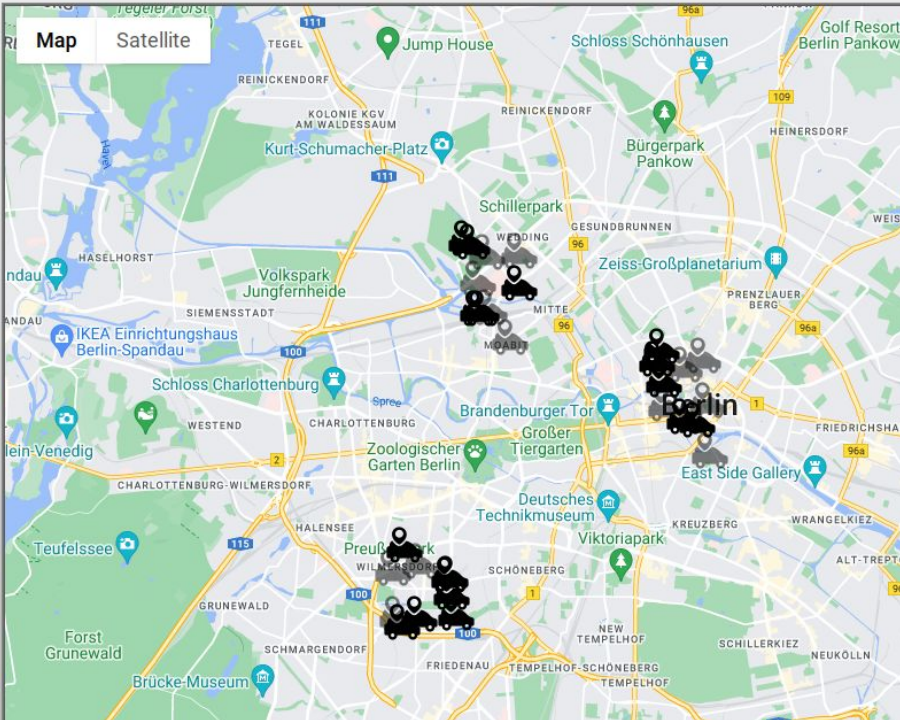
[Complex Query 1](#)

Send the query to Funnel-Spout and view results.

[Run Query](#)

Query:

For query syntax details, see the [docs](#).



The map displays the results of a query for locations within 20km of a specified coordinate in Berlin. Numerous black car icons are scattered across the city, representing the generated data points. The map includes labels for various districts and landmarks, such as the Brandenburg Gate, Zoo, and several parks.

Query results  
shown visually.



# What does it look like to a user?

```
{
  "namespace": "com.example",
  "name": "Staff",
  "type": "record",
  "fields": [
    {
      "name": "loc",
      "type": {
        "type": "record",
        "name": "Location",
        "fields": [
          {"name": "latitude", "type": "double"},
          {"name": "longitude", "type": "double"}
        ]
      }
    },
    {
      "name": "rating",
      "type": "double"
    }
  ],
  "idx_type": "spatial"
},
...
```

1. Authenticate and get a JWT token.
2. Upload a schema.
3. Upload your data as desired.
4. Execute queries on the data.

loc WITHIN 2km of (13.34, 43.21)

rating > 2.5 AND type = ECONOMY

HTTP POST  
<https://auth.funnel-labs.io/auth/realms/funnel/protocol/openid-connect/token>

client\_id=service  
&grant\_type=password  
&username=**ExampleInc**  
&password=**abcd1234**

```
$ curl
https://api.funnel-labs.io/users/ExampleInc/data/com
.example/Staff \
-X POST -d '
{"id": 3,
 "job": "WAITER",
 "loc": {"latitude": 1.0, "longitude": 2.0},
 "rating": 1.3,
 "name": "Bob"}' \
-H 'Content-Type: application/json' \
-H "Authorization: Bearer $ACCESS_TOKEN"
```

OK

# Basic Structure of Funnel System

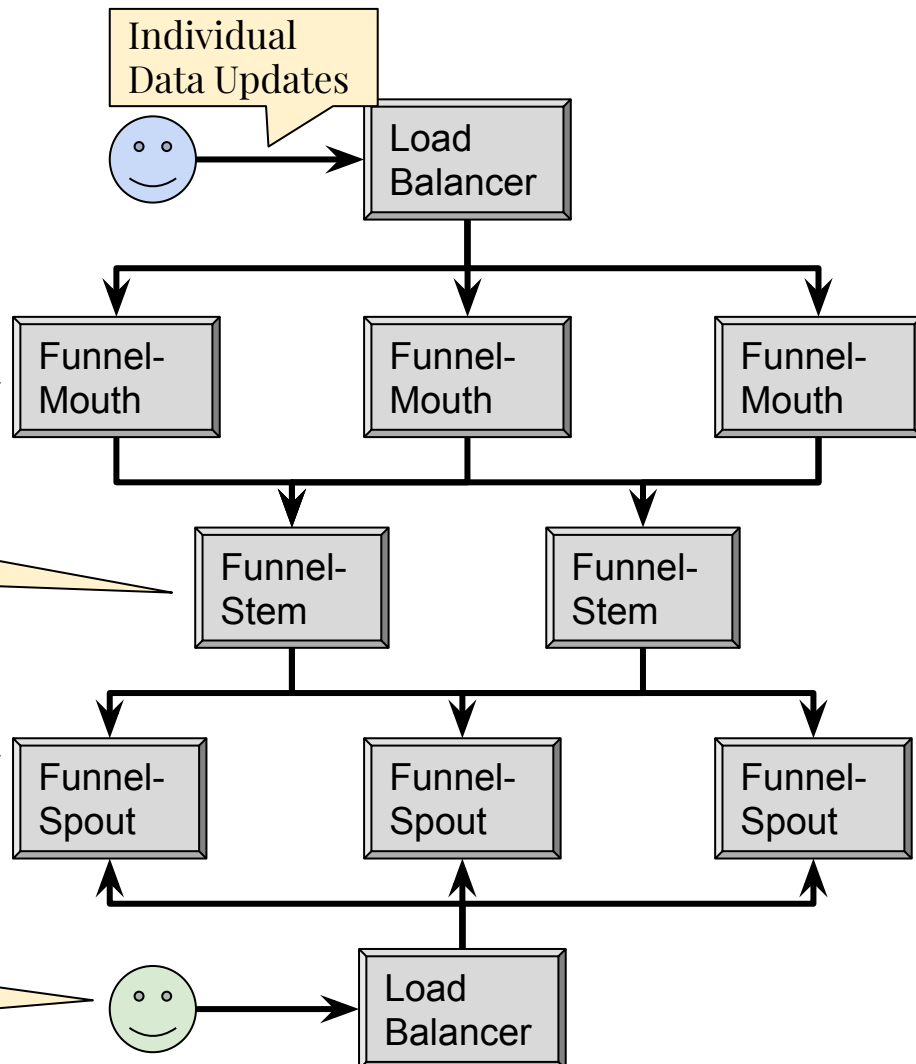
The Funnel System was built to solve this problem as cheaply as possible.

Data collected into batches to reduce network load.

Data replicated to spouts using pub/sub in batches.

Data indexed in RAM for queries.

Arbitrary queries pour in.



# Dependencies

- [Vibe.d](#): Sönke Ludwig – Web Server and concurrency
- [Pegged](#): Philippe Sigaud – Grammar based parser generator
  
- [S2 Geometry](#): Converted to D from C++ as [s2geometry-d](#)
  - A team of 1 converted this 70K loc library more completely than Go, Rust, Java, etc.
- [Avro](#): Converted to D from Java as [avro-d](#)
  - Converted in only a few weeks of work.

**Using D, small teams can match the productivity of bigger organizations.**

# Build Pipeline

---

# Docker Containers: Build Container

The build container is responsible for compiling and uploading to a Docker registry.  
**Alpine Linux lets you pack the build environment into less than 50MB, therefore faster builds.**

*# Create an Alpine Linux image w/ docker & ldc2.*

**FROM** docker:dind

**RUN** apk update

*# Add tools and dependencies.*

**RUN** apk add --no-cache gcc

**RUN** apk add --no-cache ldc

*# The dub build tool is used to build a D repository.*

**RUN** apk add --no-cache -X

<https://dl-cdn.alpinelinux.org/alpine/edge/testing>  
dub

*# Add any C dependencies needed.*

**RUN** apk add --no-cache openssl-dev

**RUN** apk add --no-cache musl-dev

**RUN** apk add --no-cache zlib-dev

*# Install AWS CLI tools for use with ECR*

**RUN** apk add --no-cache aws-cli

**RUN** aws --version

*# Create a build user with reduced privileges.*

**ARG** USER=default

**ENV** HOME /home/\$USER

**RUN** apk add --update sudo

**RUN** adduser -D \$USER \  
    && echo "\$USER ALL=(ALL) NOPASSWD: ALL" >  
    /etc/sudoers.d/\$USER \  
    && chmod 0440 /etc/sudoers.d/\$USER

**USER** \$USER

**WORKDIR** \$HOME

# Build Pipeline

Many options exist, but minimizing engineering cost, server setup, maintenance, etc. can pay off in spades.

A classic [Jenkins](#) server can be set up, but has costs whether in use or not.

Alternatively, cloud-services like [AWS CodePipeline](#) only charge for build-time.

Add a [buildspec.yml](#) file to your project which runs in your build docker container, and you're off to the races.

# Dockerized Build/Deploy Environments

When building using cloud-based resources like AWS CodePipeline, they do not know what kind of environment you need to build and run your application.

Generally, you will need 3 different types of docker containers:

- **Build Environment:** Has build tools like a compiler, library headers, and can upload built images to Docker.
- **Runtime Base:** A common environment for running D programs, with D runtime libraries.
- **Project Runtime:** The container for a specific project, with the compiled program and any additional files.

# Build Pipeline: buildspec.yml

With AWS CodeBuild, the buildspec.yml runs in your build environment, and builds a deployable image.

```
version: 0.2
env:
  variables:
    # Multiple threads is only supported in ldc2.
    DFLAGS: "--threads=4"
phases:
  install:
    commands:
      - nohup /usr/local/bin/dockerd --host=unix:///var/run/docker.sock
      --host=tcp://127.0.0.1:2375 &
      - timeout 15 sh -c "until docker info; do echo .; sleep 1; done"
  pre_build:
    commands:
      # Log into docker and the container registry.
      - AWS_ACCOUNT_ID=$(echo $CODEBUILD_BUILD_ARN | cut -d ':' -f5 )
      - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c
1-7)
      - IMAGE_NAME=funnel-mouth
      - IMAGE_TAG=${COMMIT_HASH:=latest}
      - ECR_REPO=$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION |
      docker login --username AWS --password-stdin $ECR_REPO
```

```
build:
  commands:
    - echo Build started on `date`
    - cd mouth
    - dub build -b release
    - dub test -b release
    - echo Creating docker image...
    - docker build --build-arg IMAGE_BASE=$ECR_REPO/runtime:latest -t
      $IMAGE_NAME:$IMAGE_TAG .
    - docker tag $IMAGE_NAME:$IMAGE_TAG $ECR_REPO/$IMAGE_NAME:$IMAGE_TAG
    - docker tag $IMAGE_NAME:$IMAGE_TAG $ECR_REPO/$IMAGE_NAME:latest
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push $ECR_REPO/$IMAGE_NAME:$IMAGE_TAG
      - docker push $ECR_REPO/$IMAGE_NAME:latest
```



# Docker Containers: Example Project Container

Projects have their own Dockerfile, defining what files and arguments are needed.

*# Container definition for the funnel-mouth project.*

**ARG** IMAGE\_BASE=runtime:latest

**FROM** \$IMAGE\_BASE

*## Environment Variables*

*# LISTEN\_PORT - The port where HTTP requests are received.*

*# STEM\_SUB\_HOST - The hostname of a mouth-stem server.*

**ENV** LISTEN\_PORT=8080

**ENV** STEM\_SUB\_HOST=127.0.0.1

**RUN** mkdir service

**WORKDIR** service

**COPY** target/funnel-mouth .

**RUN** sudo chown -R \$USER .

**EXPOSE** \${LISTEN\_PORT}

**ENTRYPOINT** ./funnel-mouth --port=\${LISTEN\_PORT} --stemSubUrl=tcp://\${STEM\_SUB\_HOST}:5000 --vverbose

# Docker Containers: Runtime Container

A small tidy runtime environment enables faster deployments.

*# Light-weight runtime environment.*

**FROM** alpine:3.15

*# Numerous shared libraries needed in the environment.*

**RUN** apk add ldc-runtime

**RUN** apk add openssl

**RUN** apk add musl

**RUN** apk add zlib

*# Install AWS CLI tools for use with ECR*

**RUN** apk add --no-cache aws-cli

**RUN** aws --version

*# Create a build user with reduced privileges.*

**ARG** USER=default

**ENV** HOME /home/\$USER

**RUN** apk add --update sudo

**RUN** adduser -D \$USER \  
    && echo "\$USER ALL=(ALL) NOPASSWD: ALL" >  
/etc/sudoers.d/\$USER \  
    && chmod 0440 /etc/sudoers.d/\$USER

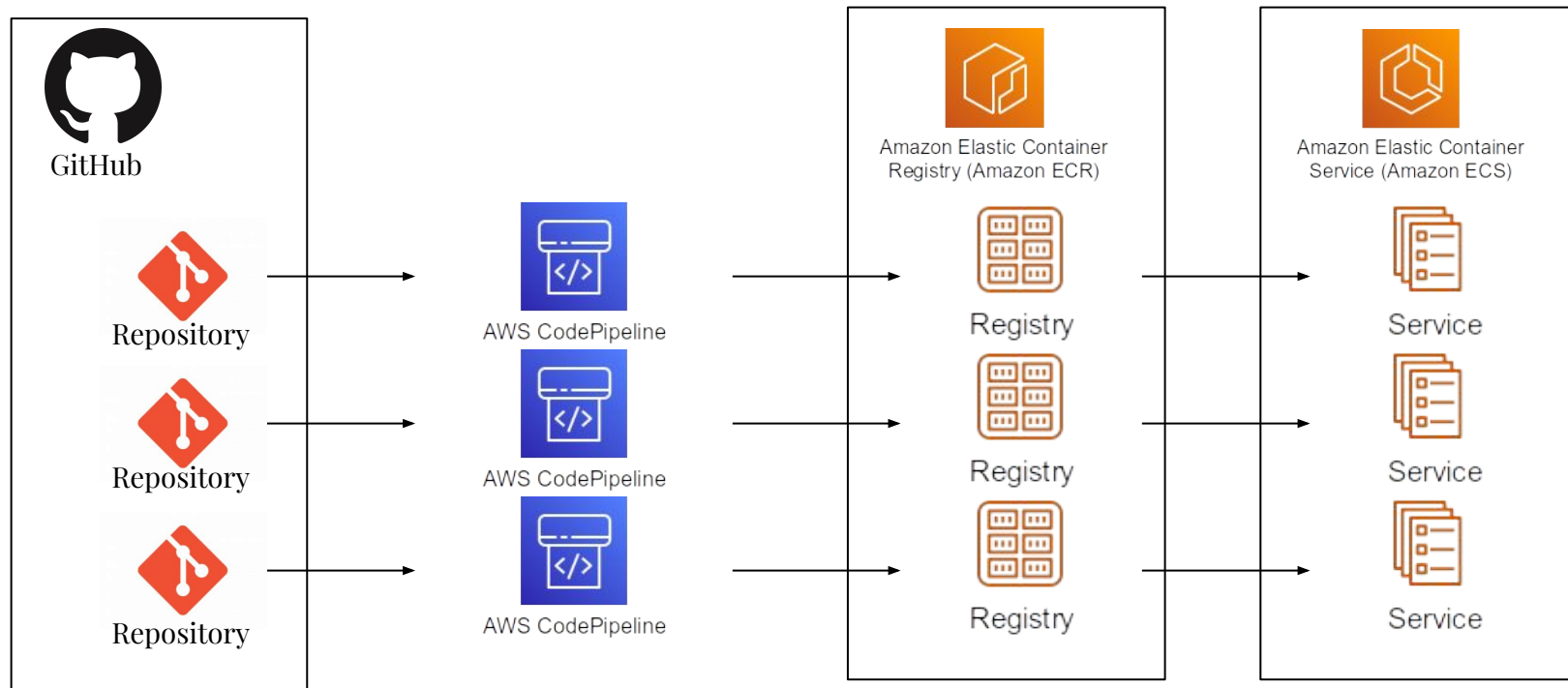
**USER** \$USER

**WORKDIR** \$HOME

# Deployment

---

# Example Deployment Flow



# Infrastructure and Deployment

If your goal is to lower costs, keep costs in mind when choosing how to deploy!

For example, should an AWS setup use [AWS Elastic Kubernetes Service](#) (EKS) or [AWS Elastic Container Service](#) (ECS)?

In terms of price:

- EKS has a [base-price](#) of \$0.10 per hour, i.e. \$72 per month, plus EC2/Fargate.
- ECS has no such base price, only the cost of EC2/Fargate resources.

## Right Tools for the Job

If you plan on deploying a project, the tools being used should match the skills of the people doing the work.

Kubernetes has excellent, albeit lengthy and complicated documentation. Deployment is done via YAML files, which are constructed very carefully.

If you are short on people and do not have dedicated DevOps staff, consider something like AWS Cloud Development Kit (CDK).

CDK consists of programming APIs that generate config files for Kubernetes, ECS, Terraform, etc.

**Program your deployment in a familiar environment with development tools.**

# Create Reuseable Functions for Standard Configurations

```
// Constructs a service that follows all the standard rules described in {@link StdServiceSettings}.
public FargateService constructStdService(Cluster cluster, ApplicationListener applicationListener, StdServiceSettings settings) {
    // A FargateService is a load-balanced set of Tasks, each of which are defined by a TaskDefinition.
    FargateTaskDefinition taskDefinition = FargateTaskDefinition.Builder.create(this, settings.name + "-td")
        .cpu(settings.cpu)
        .memoryLimitMiB(settings.memoryLimitMiB)
        .build();

    // Define and add a project container and to the TaskDefinition.
    ContainerDefinition stdContainerDefinition =
        taskDefinition.addContainer(settings.name, ContainerDefinitionOptions.builder()
            .image(ContainerImage.fromEcrRepository(
                Repository.fromRepositoryName(this, settings.name + "-ci", settings.name),
                "latest"))
            // Multiple containers can be in a host, map the host to the containers.
            .portMappings(List.of(
                PortMapping.builder()
                    .hostPort(8080)
                    .containerPort(8080)
                    .build()))
            // ...
            .build());

    FargateService fargateService = FargateService.Builder.create(this, settings.name + "-service")
        .serviceName(settings.name)
        .cluster(cluster) // Required
        .securityGroups(settings.securityGroup != null ? List.of(settings.securityGroup) : null)
        .desiredCount(settings.desiredCount) // Default is 1
        .taskDefinition(taskDefinition)
```

# Example Deployment Config in Java

```
FargateService mouthService =  
    constructStdService(  
        networkStack.getEcsCluster(), loadBalancerStack.getHttpsListener(),  
        StdServiceSettings.builder()  
            .name("funnel-mouth")  
            .securityGroup(networkStack.getSecurityGroup())  
            .hostname(hostname)  
            .urlPrefixes(List.of("/users/*/data/*"))  
            .httpRequestMethods(List.of("POST"))  
            .environment(Map.of(  
                "JWT_PUBLIC_KEY_URL", "https://auth.funnel-labs.io/auth/realms/funnel",  
                "ADMIN_REST_URL", "http://funnel-admin.local:8080")  
            )  
            .cpu(512)  
            .memoryLimitMiB(1024)  
            .desiredCount(2)  
            .build());
```

The important point is: **code-completion, documentation, functions, and other programming features can lower the cost of setting up infrastructure.**



## Concluding Remarks

- D can deliver significantly higher performance and lower costs than higher-level languages like Java, Python, JavaScript, Ruby, etc.
- D is is very feature rich, making it especially suitable for bringing in libraries from other programming languages.
- D has a good balance of feature to help speed up programming (like garbage collection) without sacrificing much for performance.
- **However**, D suffers from a smaller community and tooling needs some love.

**Thank you!**  
**Questions?**  
**Feedback?**

---