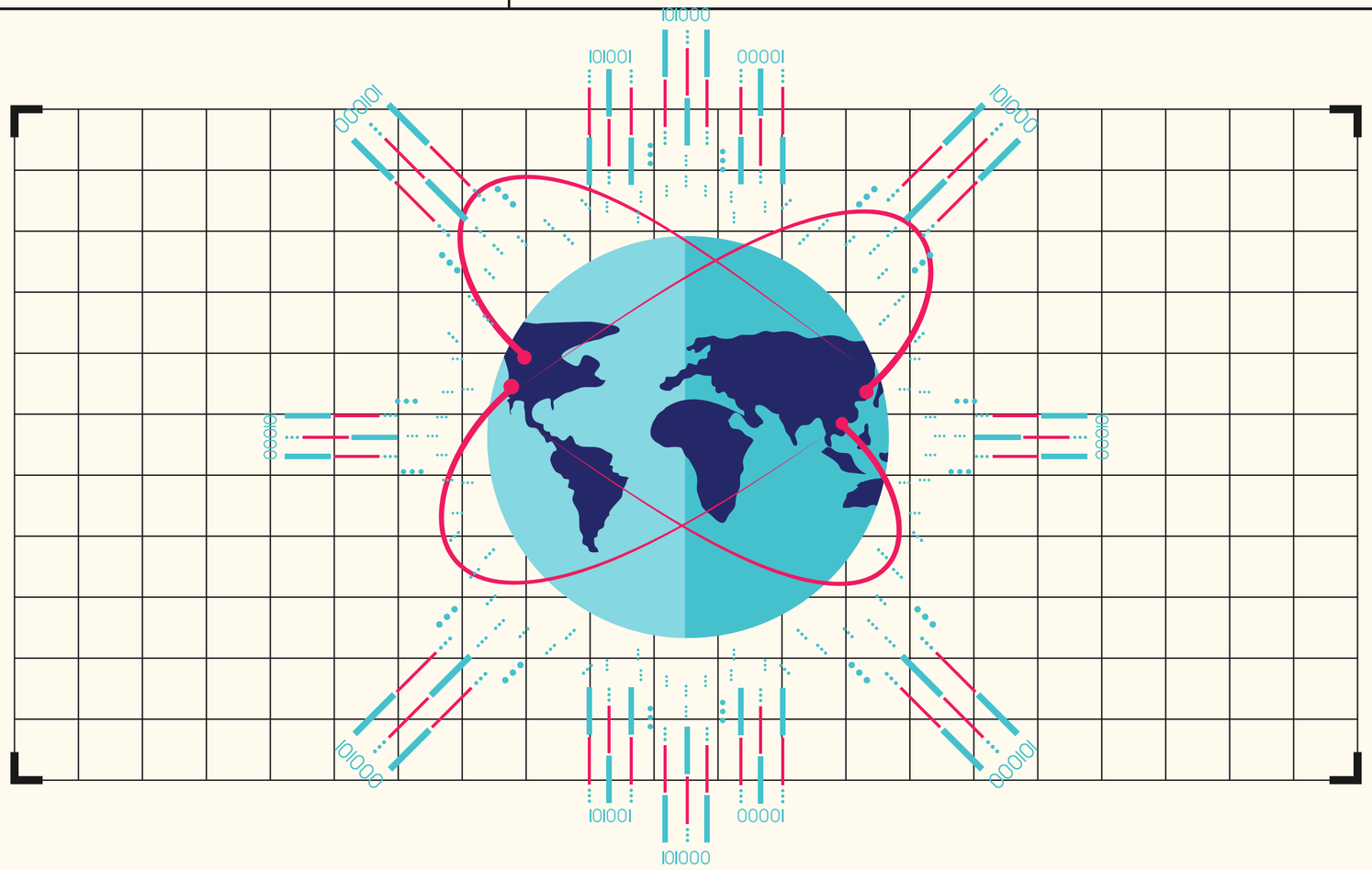
# TWINE: A UNIFIED CLUSTER MANAGEMENT SYSTEM FOR SHARED INFRASTRUCTURE

PRESENTED BY:
SATHWIC KRISHNA JAIN,
LEA RACINE AND
AJAY PRADEEP

# TOPICS ★

THE #1 RULE OF DISTRIBUTED COMPUTING: DON'T DISTRIBUTE YOUR COMPUTING! AT LEAST IF YOU CAN IN ANY WAY AVOID IT.

- Introduction
- Design & Implementation
- Scaling
- Availability and Reliability
- Evaluation
- Experiences & Lessons



“

**I don't need a hard disk in my computer if I can get to the server faster... carrying around these non-connected computers is byzantine by comparison .**

**STEVE JOBS**

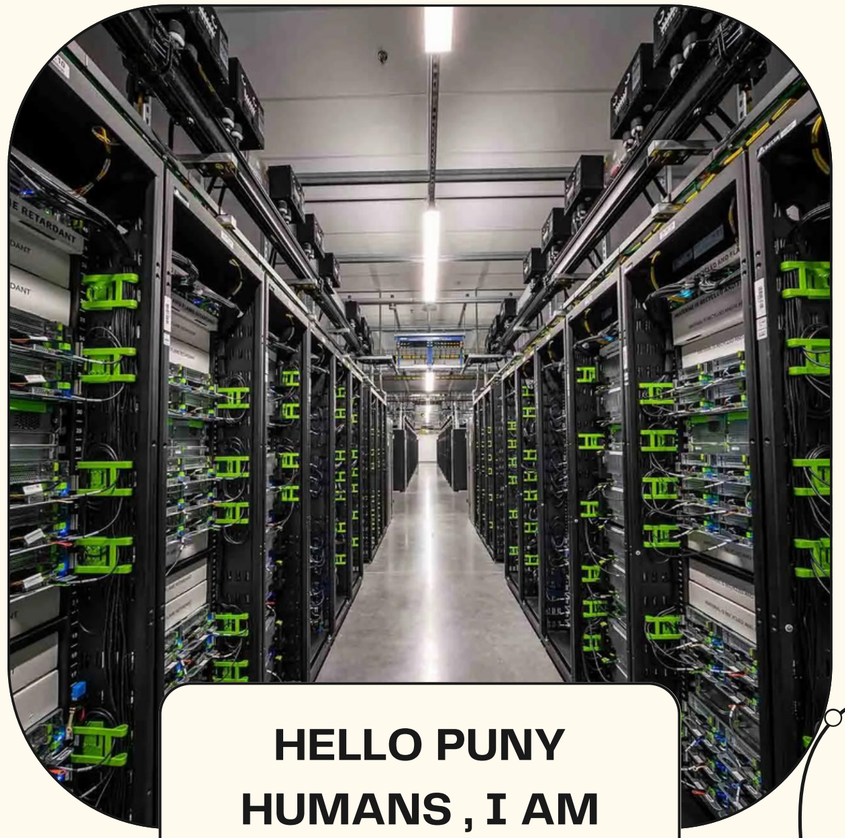
# INTRODUCTION

HEY SIRI, What is Twine and what purpose does it serve ?

I AM A  CLUSTER MANAGER DEVELOPED BY META.  I WAS DEPLOYED IN 2011.

I HAVE EXPANDED FROM ONE DATA CENTER TO 15 GEO – DISTRIBUTED DATA CENTER LOCATIONS.

I ORCHESTRATE CONTAINERS ON MILLIONS OF SERVERS TO RUN NEARLY ALL OF THE META SERVICES.



HELLO PUNY HUMANS , I AM TWINE

# DESIGN DECISIONS

Google : What design decisions did Twine make differently ?

## DECISION 1

DYNAMIC MACHINE PARTITIONING

over

STATIC CLUSTERS

## DECISION 2

CUSTOMISATION IN SHARED INFRASTRUCTURE

over

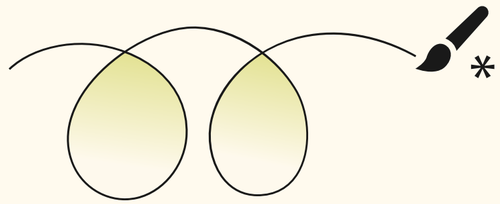
PRIVATE POOLS

## DECISION 3

SMALL MACHINES

over

BIG MACHINES



UPLOAD

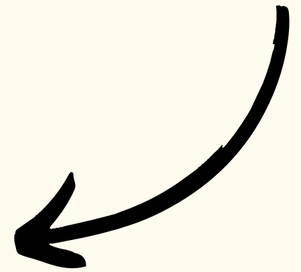
# What if we used Kubernetes ?

An open source cluster management system

kubernetes

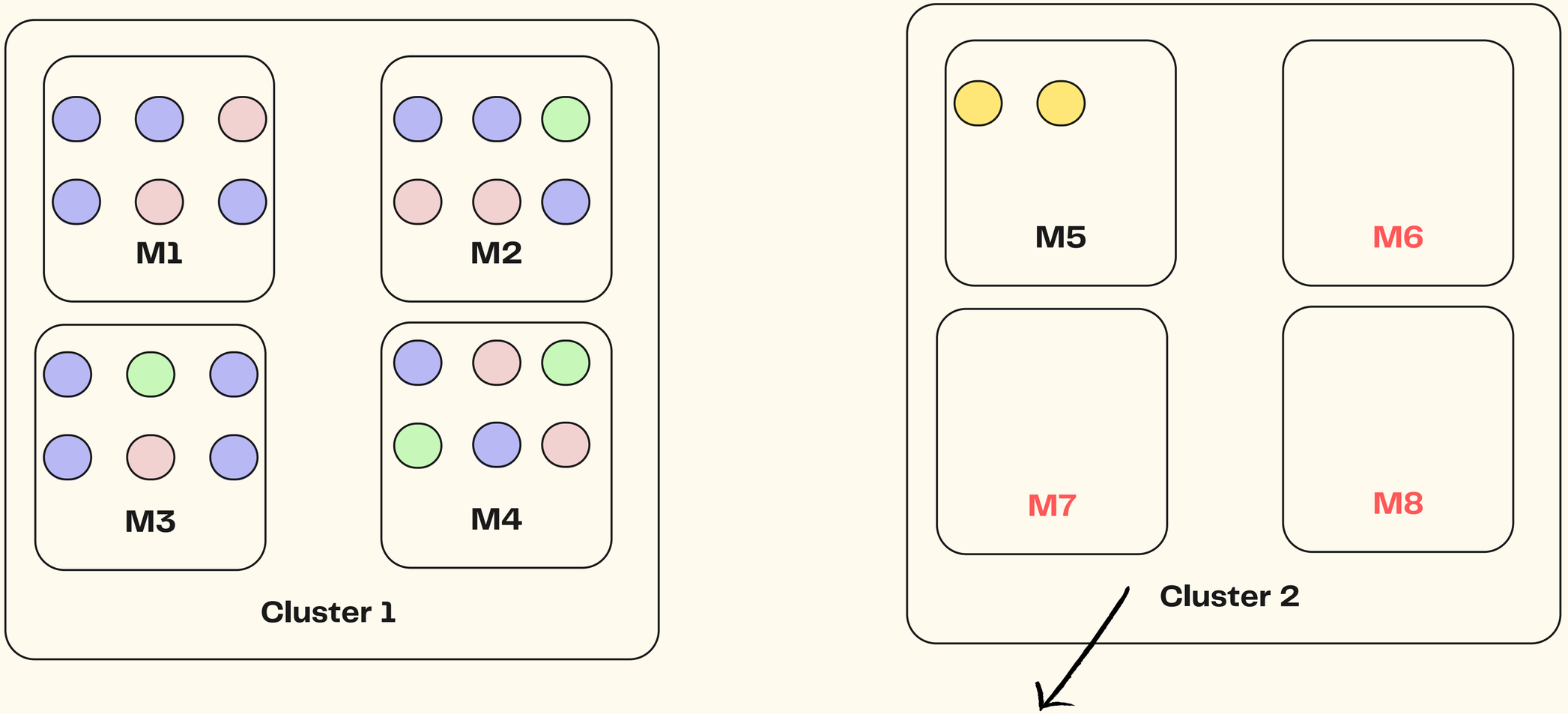
# Building Large clusters

**Kubernetes supports clusters with up to 5000 nodes.**



***The Problem***

**Millions of machines globally, which means thousand of isolated clusters.**


M1
M2
M3
M4
Cluster 1
M5
M6
M7
M8
Cluster 2
STRANDED CAPACITY

**M6,M7,M8 ARE AVAILABLE**
**BUT JOBS IN CLUSTER 1 CANNOT USE THEM.**

**MAKE MACHINE ASSIGNMENT DYNAMIC**

# Taking all this into consideration !

BARD: WHAT IS REQUIRED FOR A GOOD CLUSTER MANAGEMENT SYSTEM

1. A SINGLE CONTROL PLANE TO MANAGE ONE MILLION MACHINES.
2. COLLABORATIVE LIFECYCLE MANAGEMENT
3. HOST – LEVEL CUSTOMIZATION
4. POWER – EFFICIENT MACHINES
5. SHARED INFRASTRUCTURE

# SINGLE CONTROL PLANE

Up next : Collaborative lifecycle management

1

AN ISOLATED CONTROL PLANE PER CLUSTER RESULTS IN STRANDED CAPACITY AND OPERATIONAL BURDEN
BECAUSE WORKLOADS CANNOT EASILY MOVE ACROSS CLUSTERS

2

LARGE-SCALE HARDWARE REFRESH IN A CLUSTER MAY RESULT IN IDLE MACHINES AND OPERATIONAL OVERHEAD.

# COLLABORATIVE LIFECYCLE MANAGEMENT

Up next : HOST LEVEL OPTIMISATION

1

**CLUSTER MANAGEMENT SYSTEMS GENERALLY LACK VISIBILITY INTO HOW AN APPLICATION MANAGES ITS INTERNAL STATE.**

**IT LEADS TO SUBOPTIMAL HANDLING OF HARDWARE AND SOFTWARE LIFECYCLE EVENTS THAT IMPACT APPLICATION AVAILABILITY.**

**TWINE PROVIDES A NOVEL TASKCONTROL API TO ALLOW APPLICATIONS TO COLLABORATE WITH TWINE IN HANDLING TASK LIFECYCLE EVENTS THAT IMPACT AVAILABILITY.**

# HOST - LEVEL CUSTOMIZATION

Up next : POWER EFFICIENT MACHINES

1. HARDWARE AND OS SETTINGS MAY SIGNIFICANTLY IMPACT APPLICATION PERFORMANCE.
2. TWINE LEVERAGES ENTITLEMENTS, OUR QUOTA SYSTEM, TO HANDLE HARDWARE AND OS TUNING.
3. ASSOCIATE EACH ENTITLEMENT WITH A HOST PROFILE, A SET OF HOST CUSTOMIZATIONS THAT THE ENTITLEMENT OWNER CAN TUNE.
4. TWINE DYNAMICALLY ALLOCATES MACHINES TO ENTITLEMENTS AND SWITCHES HOST PROFILES ACCORDINGLY

# POWER - EFFICIENT MACHINES

Up next : SHARED INFRASTRUCTURE

1. MAXIMIZE PERFORMANCE PER WATT, EITHER BY EMPLOYING UNIVERSAL STACKING ON BIG MACHINES OR DEPLOYING POWER-EFFICIENT SMALL MACHINES.

2. NEED TO OPTIMIZE FOR OUR INTERNAL WORK-LOADS.

3. ADOPT SMALL MACHINES WITH A SINGLE CPU AND 64GB RAM

# SHARED INFRASTRUCTURE

Up next : TWINE DESIGN AND IMPLEMENTATION

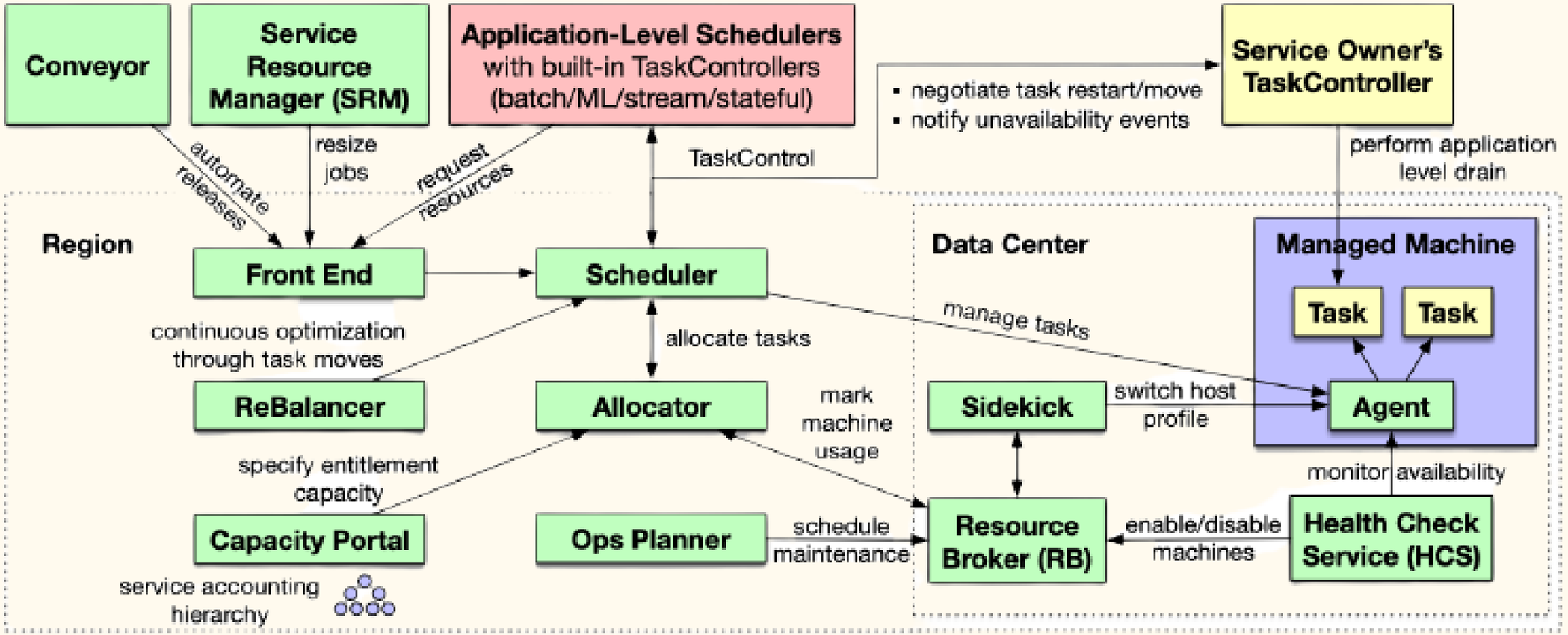
1. MIGRATING WORKLOADS ONTO A SINGLE SHARED COMPUTE POOL, TWSHARED, AND A SINGLE SHARED STORAGE POOL

2. TWSHARED HOSTS THOUSANDS OF SYSTEMS, INCLUDING FRONTEND, BACKEND, ML, STREAM PROCESSING, AND STATEFUL SERVICES.

3. TWSHARED HAS BECOME META'S UBIQUITOUS COMPUTE POOL, AS ALL NEW COMPUTE CAPACITY LANDS ONLY IN TWSHARED.
ACHIEVING NEAR 100% SHARED INFRASTRUCTURE CONSOLIDATION.

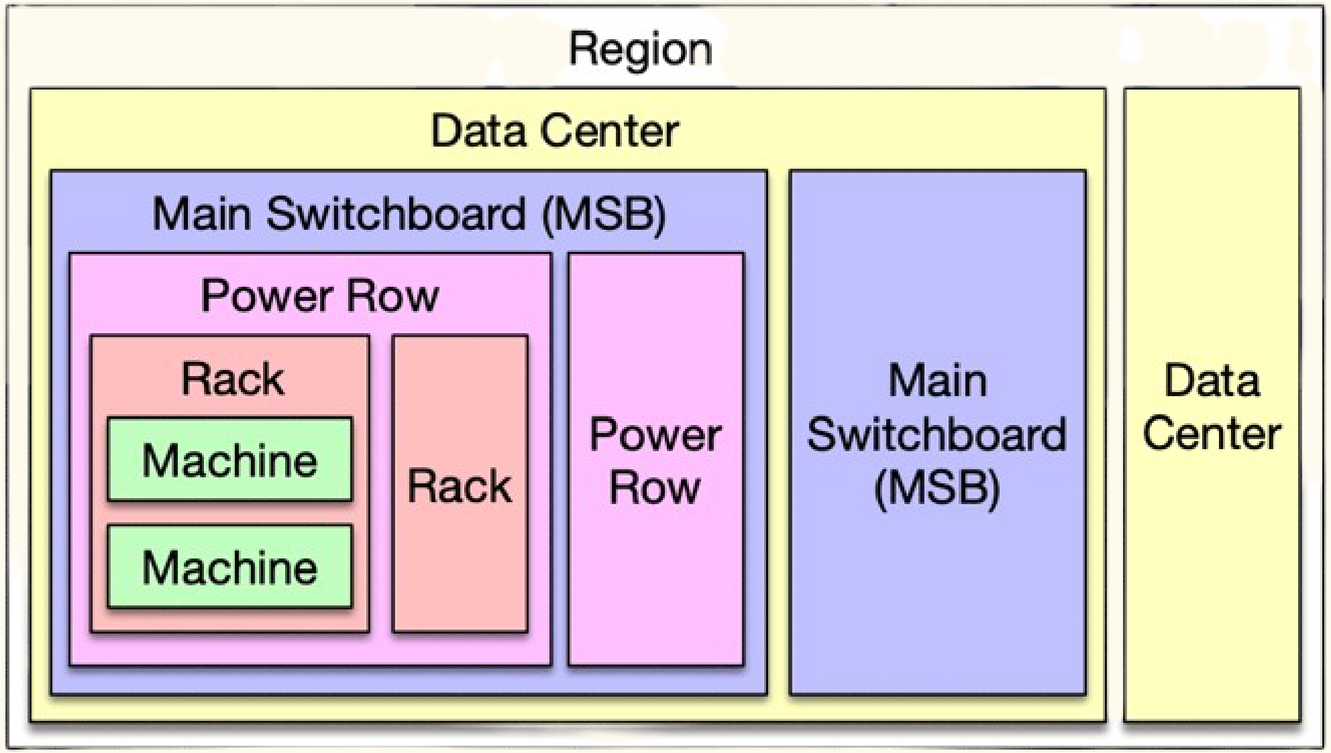
# TWINE DESIGN

DALLE 2 : HOW DOES TWINE LOOK ?



# DATA CENTER TOPOLOGY

UP NEXT : WORKING OF TWINE ECOSYSTEM

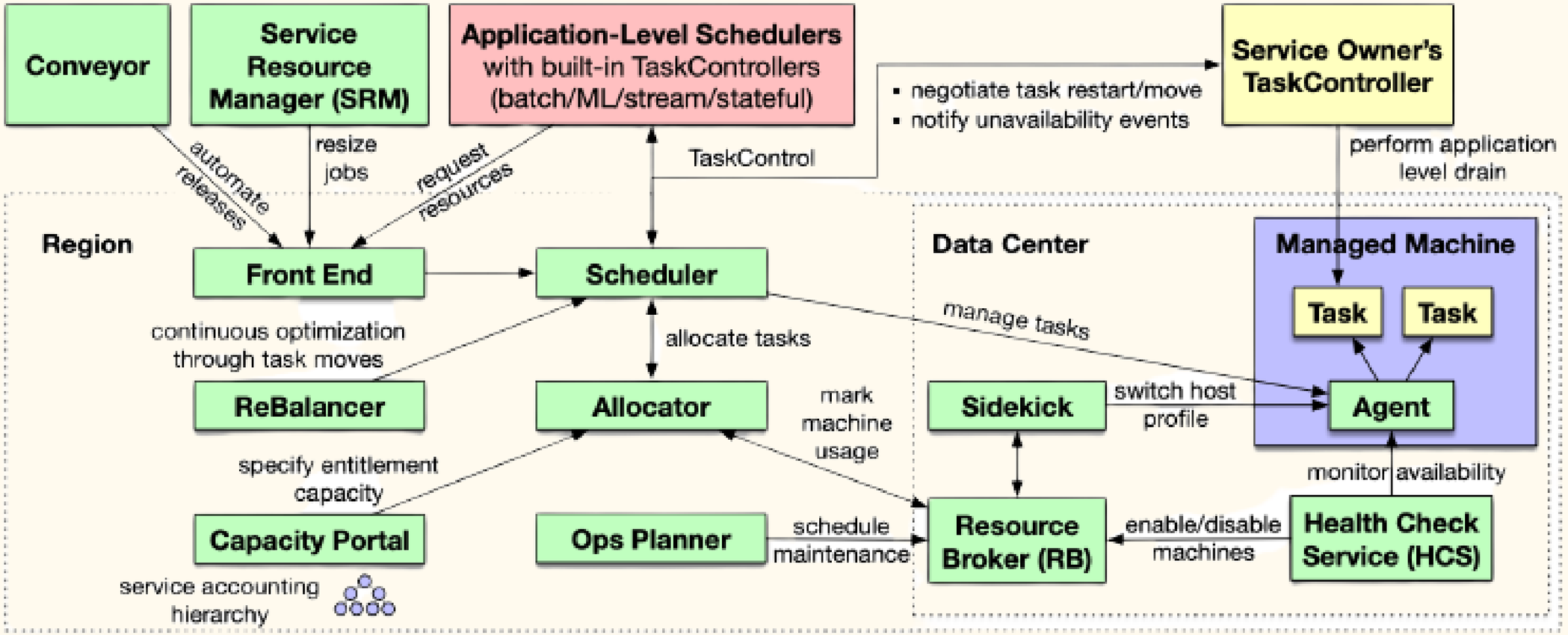



1. A MAIN SWITCHBOARD (MSB) IS THE LARGEST FAULT DOMAIN IN A DC WITH SUFFICIENT POWER AND NETWORK ISOLATION TO FAIL INDEPENDENTLY.

2. A SINGLE TWINE CONTROL PLANE TO MANAGE JOBS ACROSS DCS.

# TWINE DESIGN

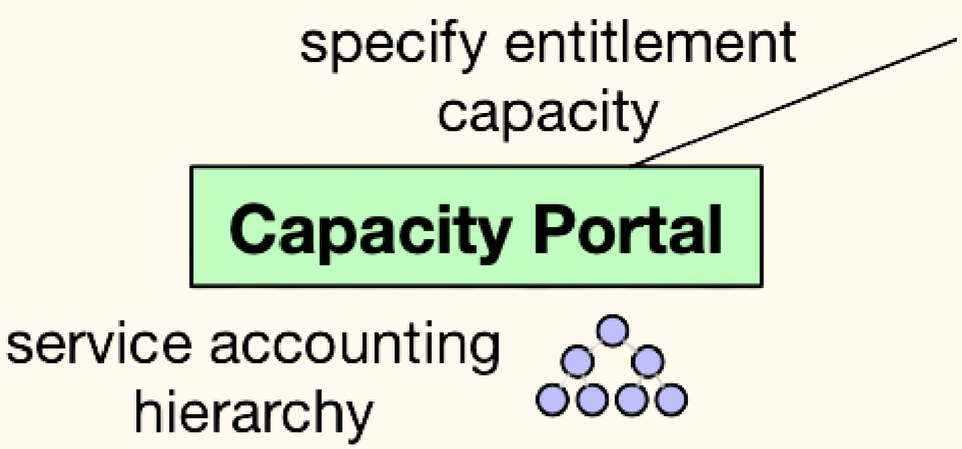
COMING BACK



# Twine Ecosystem

# CAPACITY PORTAL

UP NEXT: SCHEDULER

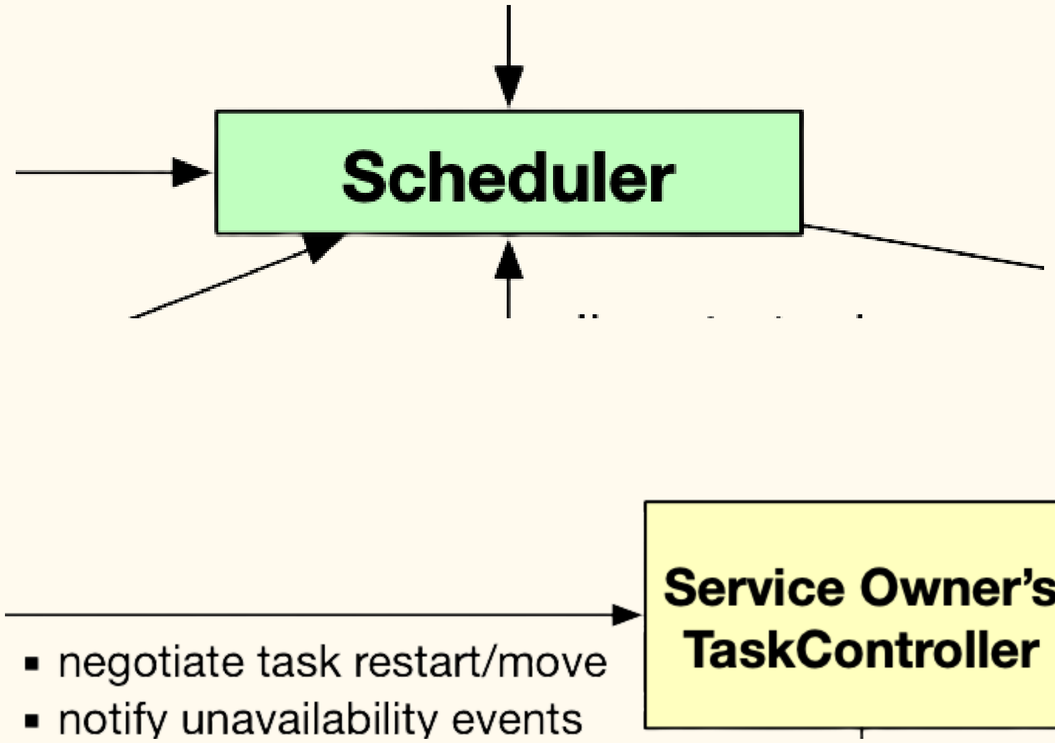



1. **The Capacity Portal allows users to request or modify entitlements**

2. **With a granted entitlement, a user deploys jobs through the *front end*.**

# SCHEDULER

UP NEXT : WAIT I'M NOT DONE




1

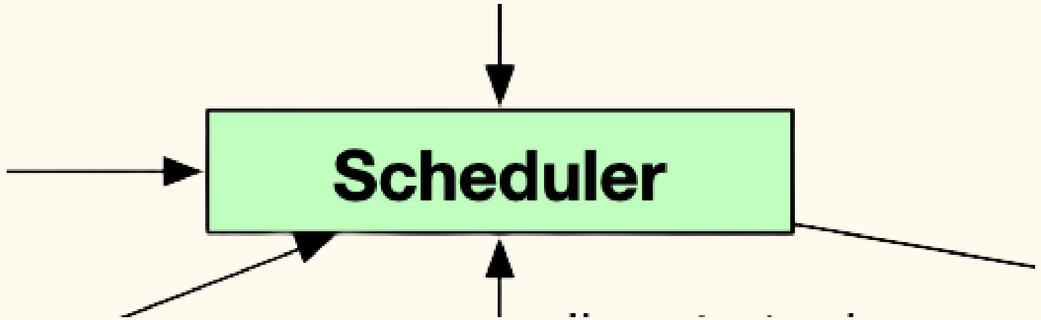
Manages job and task lifecycle, e.g., orchestrating a job's software release.

2

The scheduler coordinates with the TaskController to make decisions, e.g., delaying a task restart to rebuild a lost data replica first.

# SCHEDULER

UP NEXT : ALLOCATOR

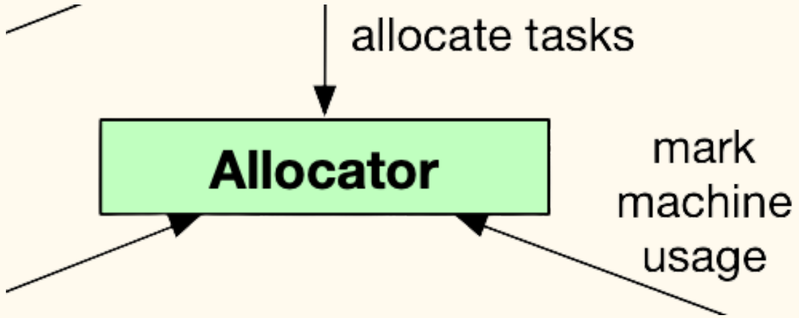



3 The scheduler paces changes to a job’s tasks to avoid application downtime

4 The scheduler has built-in support for commonly used lifecycle policies and offers the TaskControl API to implement more complex policies.

# ALLOCATOR

Up next : ReBalancer




1. Assigns machines to entitlements and assigns tasks to machines.

2. The scheduler calls the allocator to perform a job allocation when a new job starts, an existing job changes size, or a machine fails

3. The allocator uses multiple threads to perform concurrent allocations for different jobs and relies on optimistic concurrency control to resolve conflicts

# REBALANCER

Up next : Resource Broker

continuous optimization
through task moves

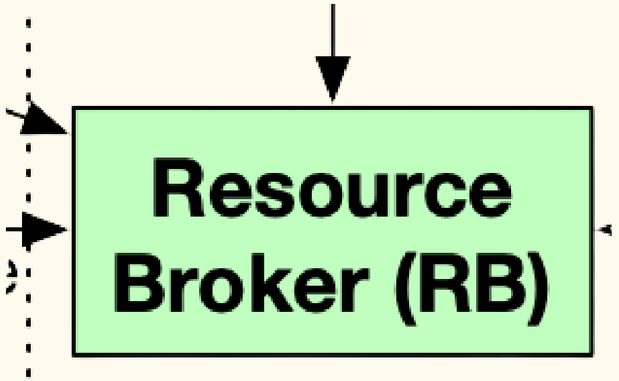
**ReBalancer**

1. **Runs asynchronously and continuously to improve the allocator’s decisions.**

2. **Better balancing the utilization of CPU, power, and network.**

# RESOURCE BROKER

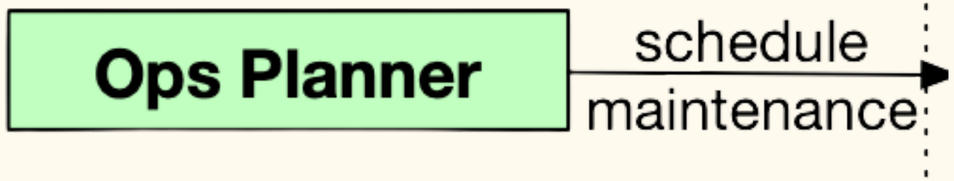
Up next : Ops Planner




**Resource Broker (RB) stores machine information and unavailability events that track hardware failures and planned maintenance.**

# OPS PLANNER

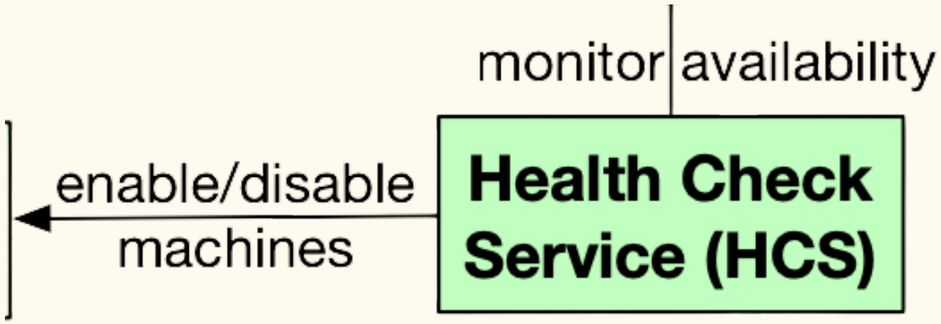
Up next: Health Service Planner




**DC operators schedule planned maintenance through Ops Planner.**

# HEALTH CHECK SERVICES

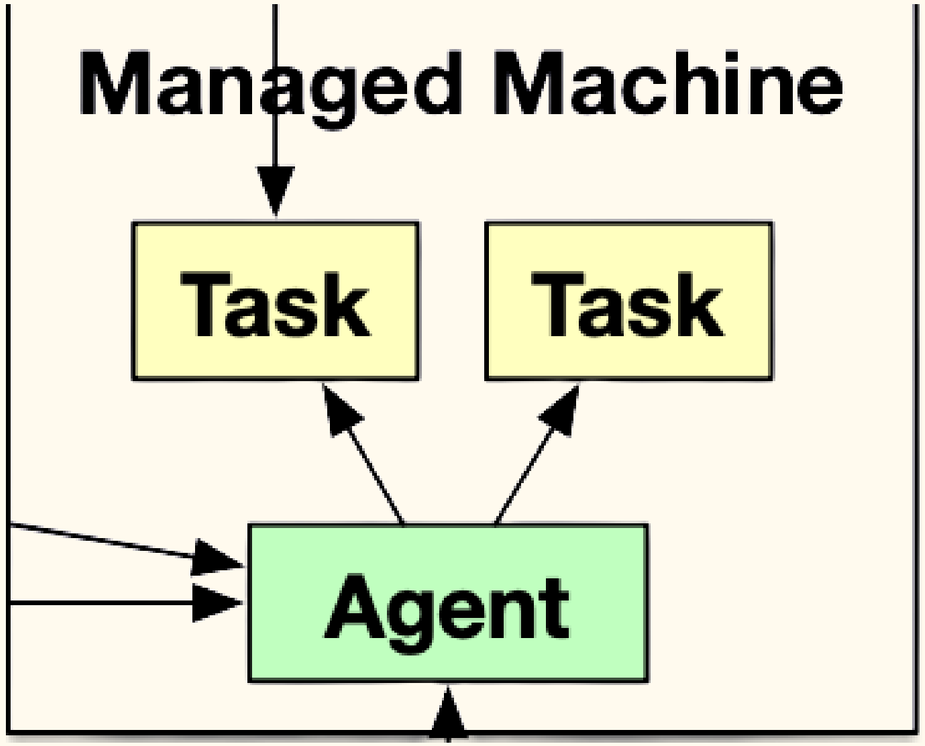
Up next : Agent




**Health Check Service (HCS) monitors machines and updates their status in RB.**

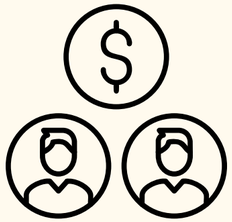
# AGENT

Up next : Sidekick

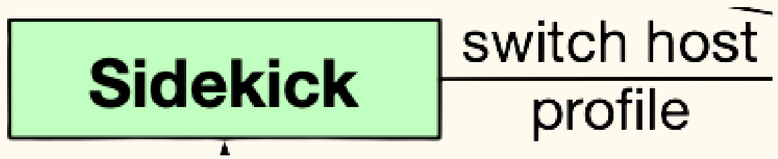



The agent runs on every machine to manage tasks.

# SIDEKICK



Up next : Service Resource Manager



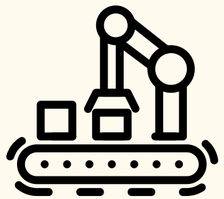

**Sidekick switches host profiles as needed.**

# SERVICE RESOURCE MANAGER

Up next: Conveyor


**Service Resource Manager (SRM) autoscales jobs in response to load changes.**

# CONVEYOR



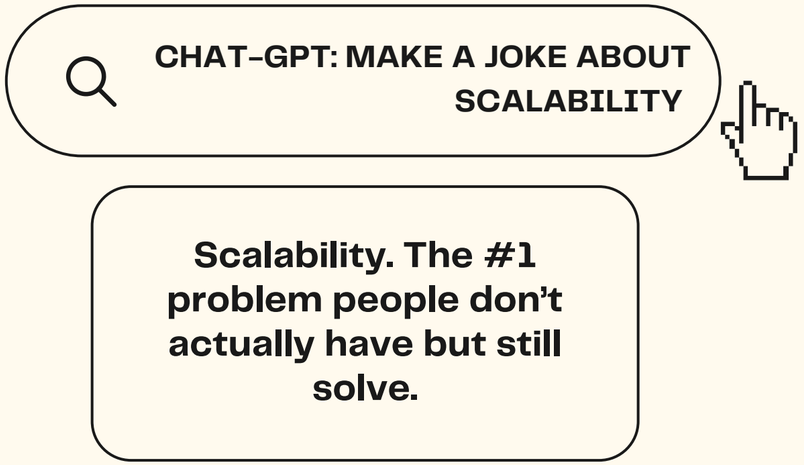
WHEW! I THINK WE'RE DONE,
NO WAIT THERE'S MORE!

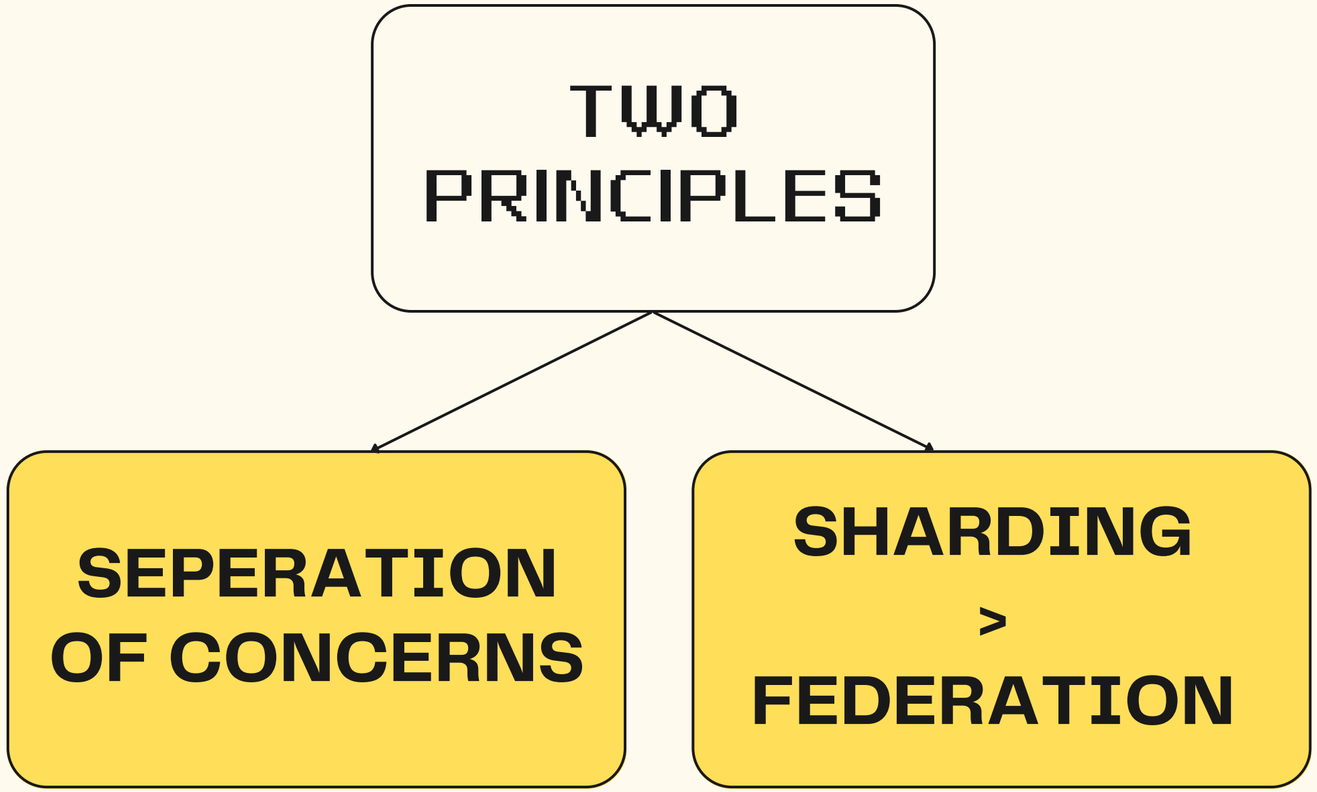
CONVEYOR

automate releases

**Conveyor is Meta's continuous delivery system.**

# SCALING


CHAT-GPT: MAKE A JOKE ABOUT SCALABILITY
Scalability. The #1 problem people don't actually have but still solve.

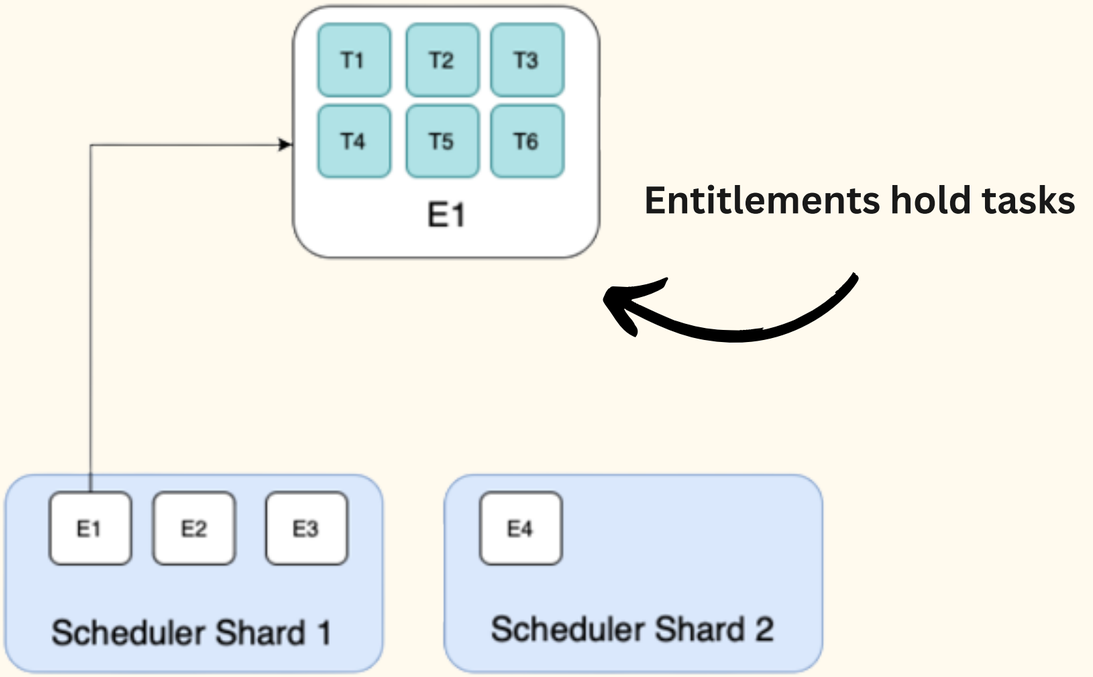

TWO PRINCIPLES
SEPERATION OF CONCERNS
SHARDING > FEDERATION

# SCALING WITH SHARDING

CHAT GPT : WHAT IS SHARDING ?

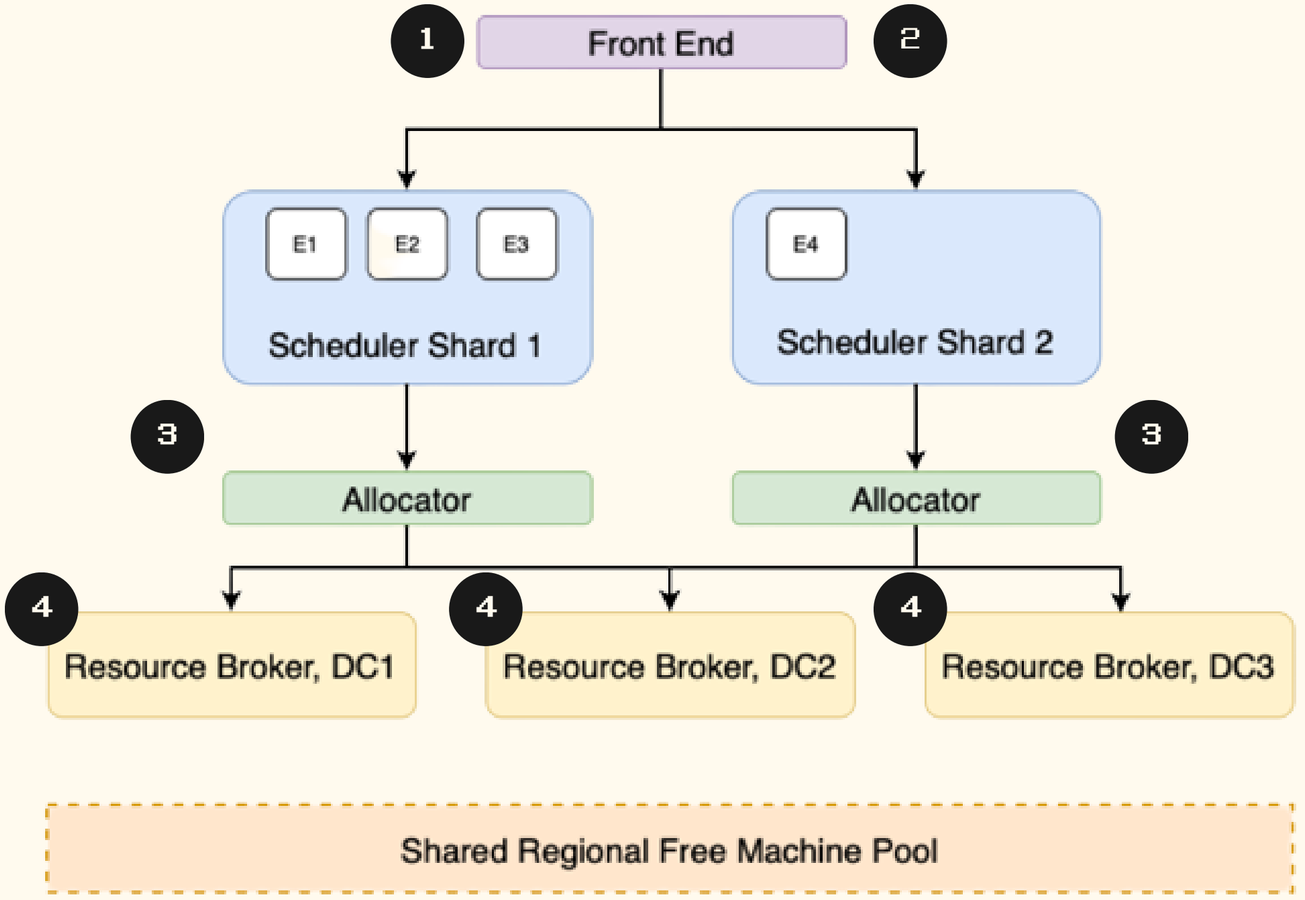
**Dear Human** , It is a technique used in database and distributed systems that involves partitioning data across multiple nodes in a cluster.

1. **Shard twine schedulers by entitlements**
2. **Assignment goes to shards with the least load**
3. **Entitlements can change size and change shards**



1. A MAP IS KEPT OF THE ENTITLEMENT-TO-SHARD ALLOCATIONS
2. REQUESTS ARE FORWARDED TO THE RELEVANT SHARD
3. ALLOCATOR FOR EACH SCHEDULER
4. A RESOURCE BROKER MANAGES MACHINES IN A REGION

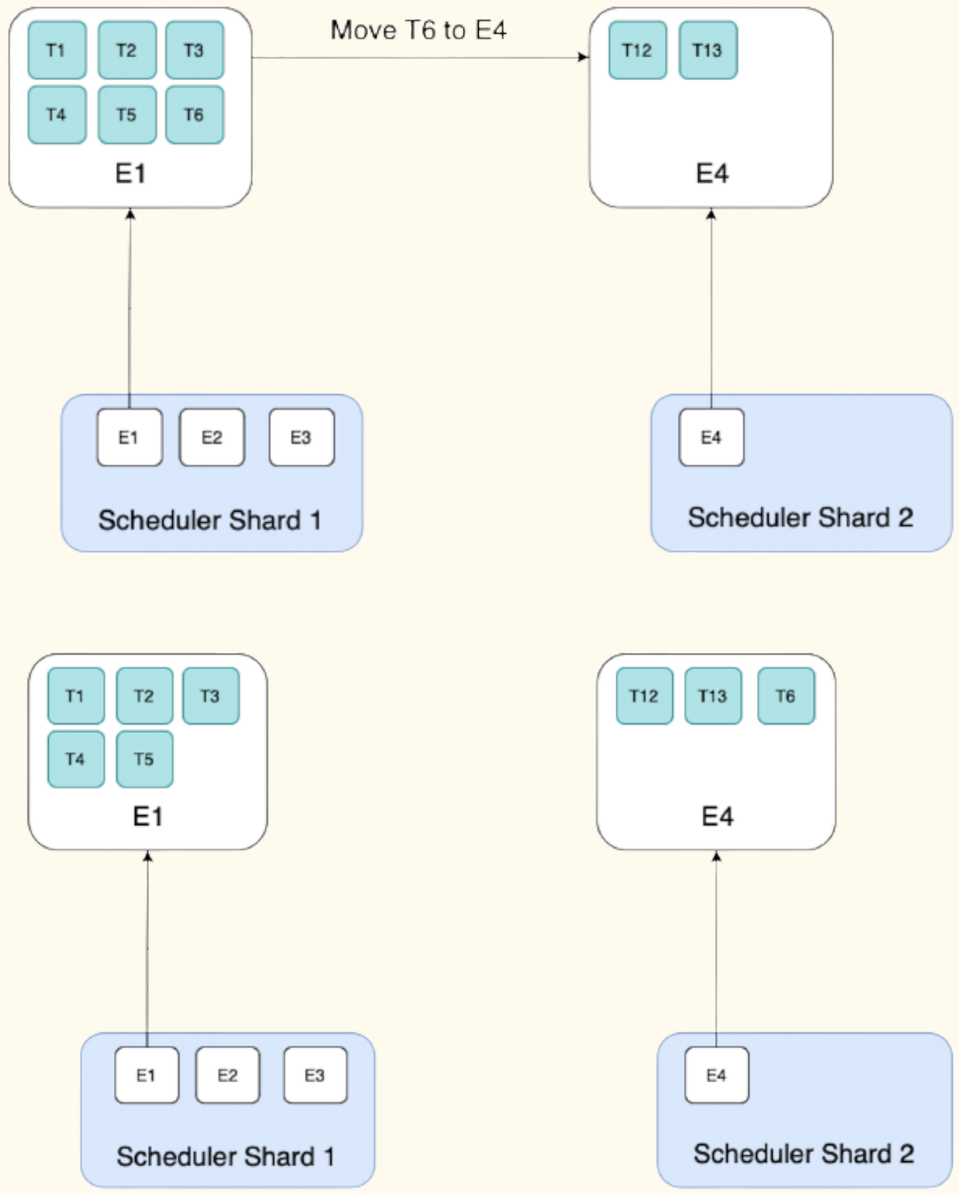
DALLE 2 : DRAW ME A DIAGRAM OF HOW SHARDING WORKS



# What happens if a scheduler shard cannot handle the entitlements?

# SHARDING AND OVERLOADING

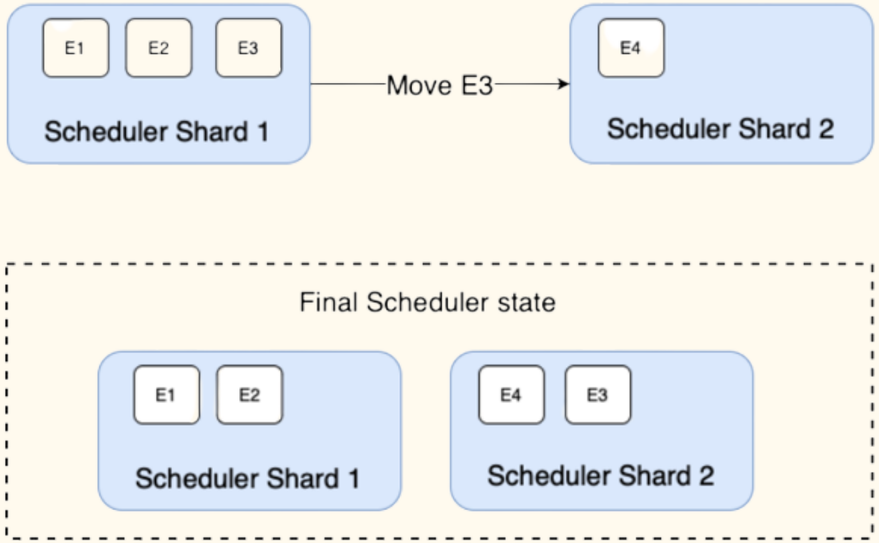
**Individual jobs can also be moved from one entitlement to another**



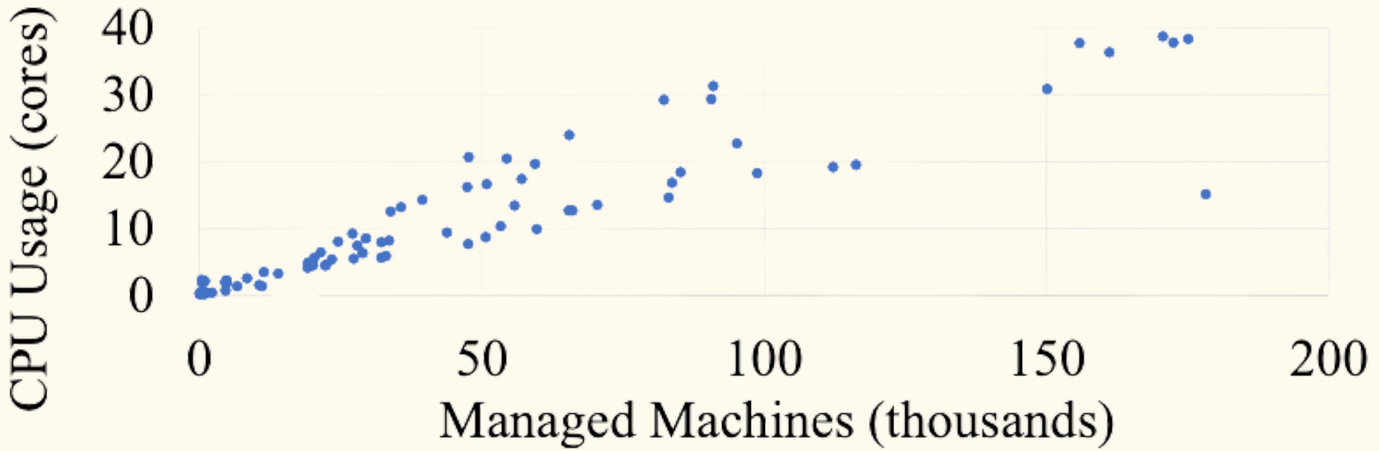
# SHARDING AND OVERLOADING

ALEXA, WHAT HAPPENS IF A SHARD OVERLOADS?

**Entitlements moved to other shards without restarting task**



# SCALING AND SHARDING




1 Sharding allows the scheduler to easily scale to one million machines

2 Smaller shards <170K machines to prevent shard failure

3 50K stable region, 20 shards for 1 million machines

# What have we missed?

# What have we assumed?

# SHARDING ISSUES

OK GOOGLE : TELL ME THE ISSUES WITH SHARDING ?

**502 That's an error ! That's all we know. TRY AGAIN !**

**A single job must fit in a single scheduler shard**

**Not practical**

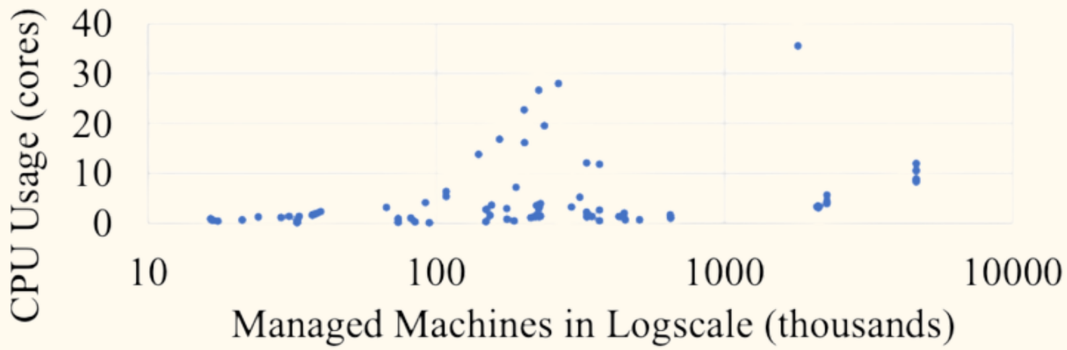


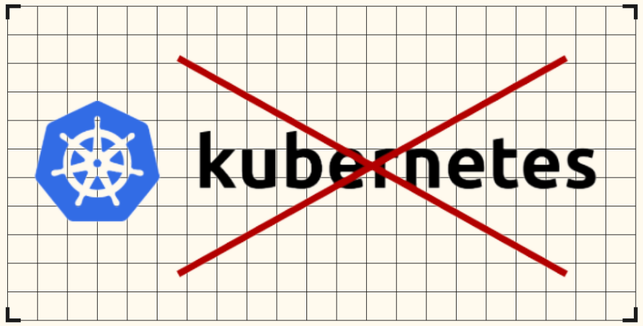

**Figure 12:** P99 CPU usage of production allocators over one week.

**The largest global allocator is scalable because of a high cache hit ratio**

**Does not handle allocations for short-lived batch jobs**

**Does not perform time-consuming optimizations**

# SCALE OUT VIA SEPARATION OF CONCERNS




2

All twine components are scaled and sharded independently

Frontend

Scheduler

Allocator

Resource broker

Health check service

Sidekick

CHATGPT: HOW DO I SHARE LENGHTY INFO IN WAYS, THE AUDIENCE DO NOT SLEEP?

Dear Human ...PROCEEDS TO GIVE A LENGTHY ANSWER..... SLEPT WELL

1

AVOID CENTRALISING COMPONENTS

3

Stateful components have their own external persistent store for metadata

Allows sharing and independent sharding of persistent stores

# SCALE OUT VIA SEPARATION OF CONCERNS

CHATGPT: Make a joke about TWINE

Why did the IT manager use Twine for cluster management?
Because he wanted to keep all his nodes tied together!

HELPS THE ALLOCATOR SCALE : REBALANCER IS TIME-CONSUMING

SEPARATING RESPONSIBILITY BETWEEN TWINE AND APPLICATION-LEVEL

REDUCE HIGH SCHEDULING LOADS

AVOIDS FREQUENT HOST PROFILE CHANGES

APPLICATION-LEVEL HANDLES FINE-GRAINED RESOURCE ALLOCATION AND LIFECYCLE OPERATIONS

# COMPARISON OF SHARDING AND FEDERATION

Up next: LOADING.....

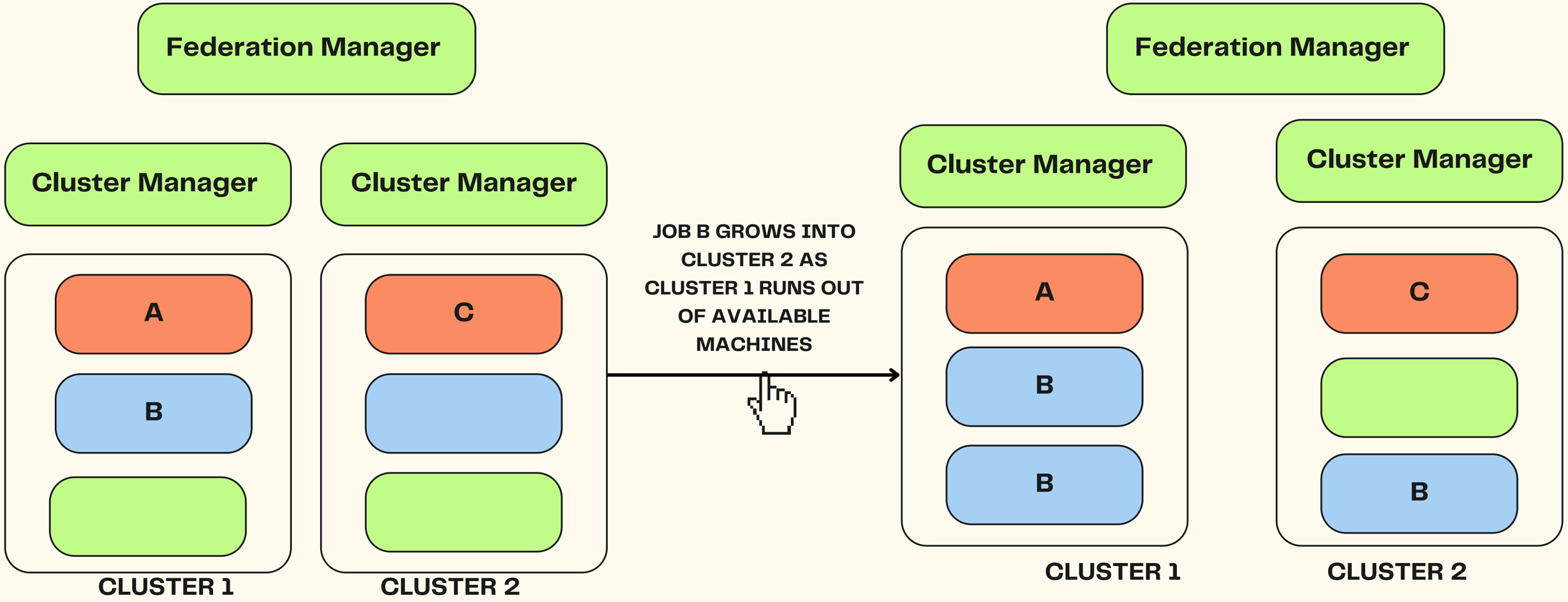
1. WE DO NOT WANT TO KEEP JOB WITHIN A CLUSTER

2. LEADS TO STRANDED CAPACITY
   SOME OVERLOADED, SOME IDLE
   OPERATIONAL BURDEN

3. CAN MOVE JOBS OR MACHINES IN AND OUT OF ENTITLEMENTS RATHER THAN A JOB SPLIT ACROSS MULTIPLE STATIC CLUSTERS (KUBERNETES)

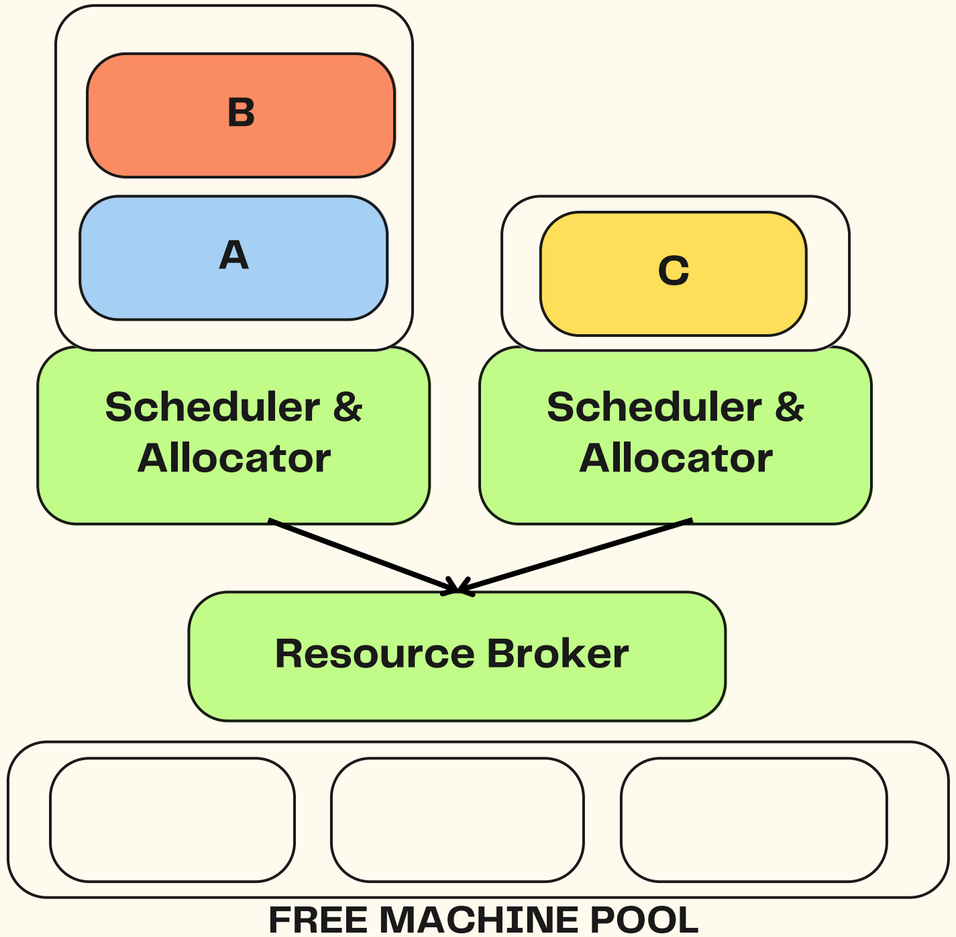
# COMPARISON OF SHARDING AND FEDERATION

**Did you know: that Twine uses a declarative configuration language that allows users to define their cluster infrastructure in a simple, human-readable format?**

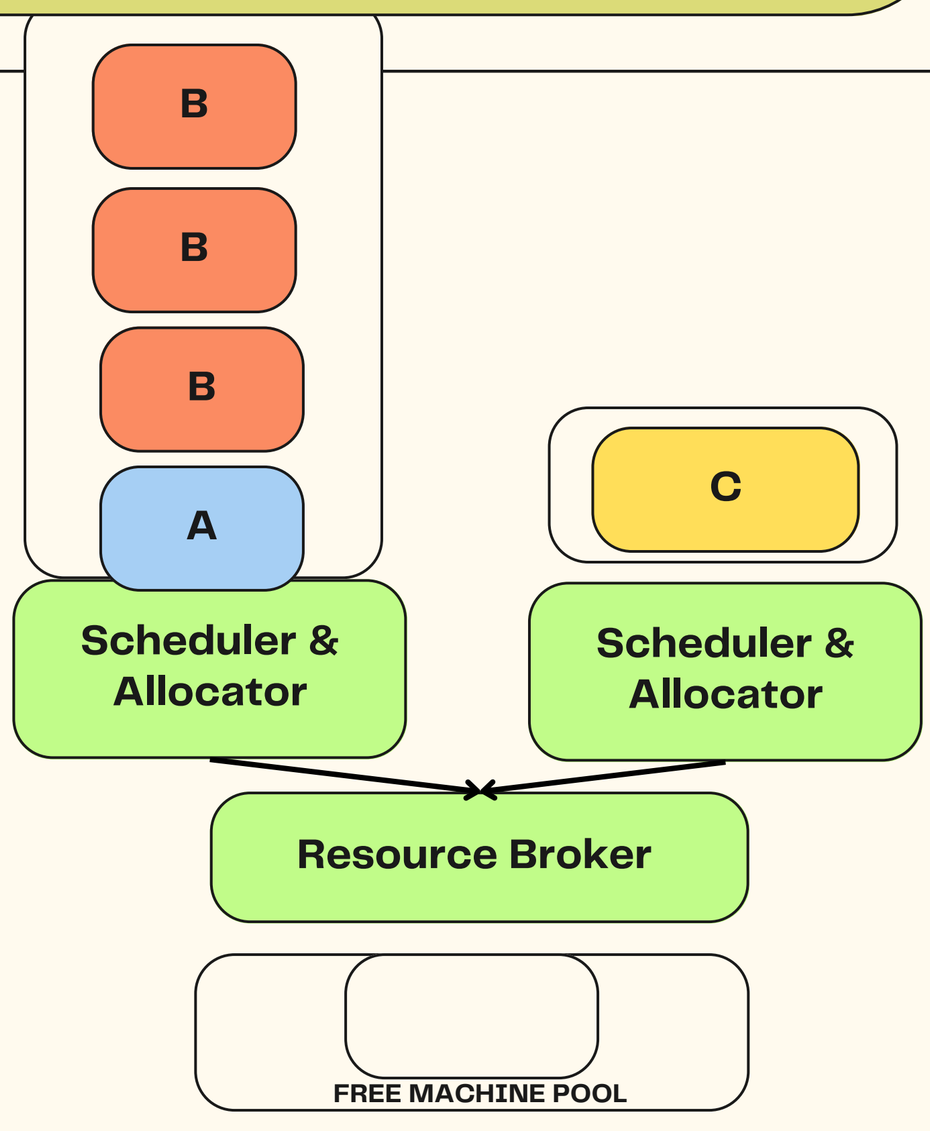


# COMPARISON OF SHARDING AND FEDERATION

Did you know: In addition to Facebook, several other companies use Twine to manage their own large-scale clusters, including Twitter, Airbnb, and Yelp?



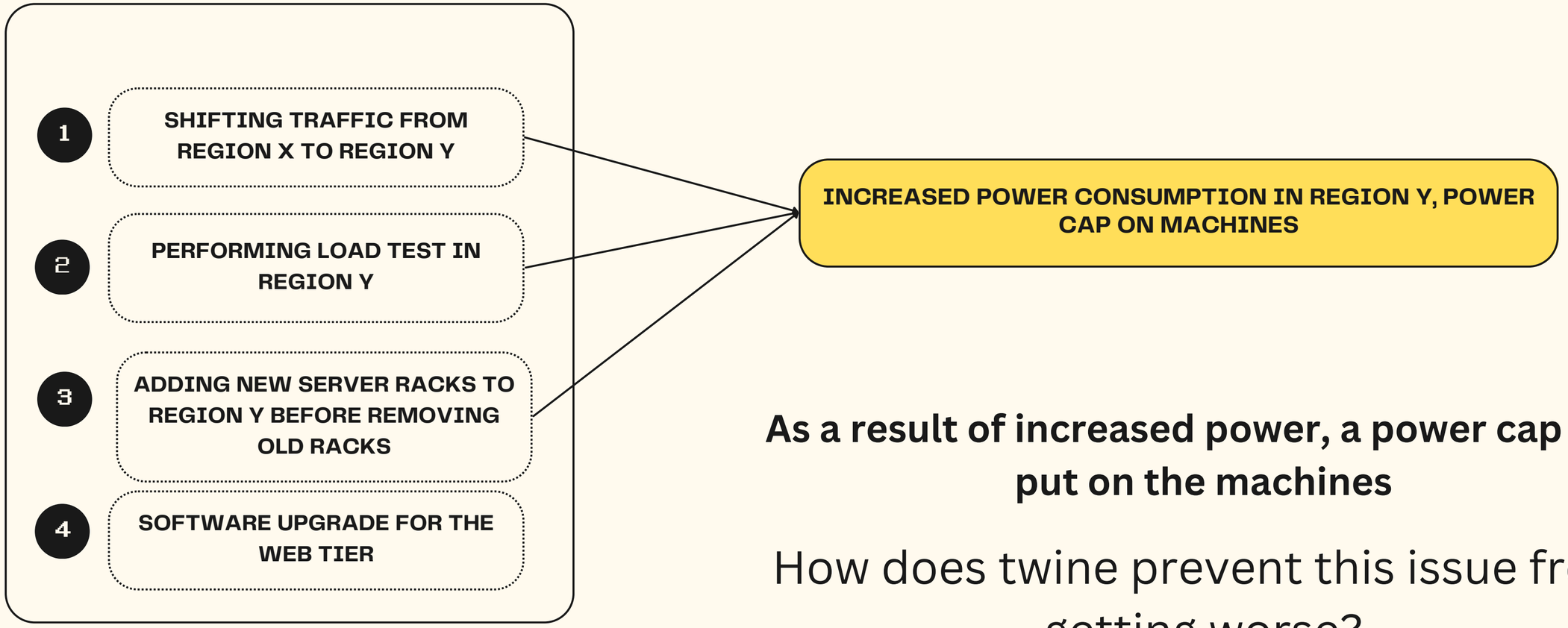

JOB B GROWS AND STAYS IN ENTITLEMENT 1 AS MORE MACHINES ARE ADDED TO ENTITLEMENT 1



# COMPARISON OF SHARDING AND FEDERATION

- Supports complex deployments but at the cost of complexity during scaling
- Federation Manager performs complex coordination
- Metadata and management operations are split among multiple distributed cluster managers

- Simpler as jobs are managed by one scheduler shard only
- Resource broker is a simple interface to manage the shared regional pool of machines




**As a result of increased power, a power cap was put on the machines**

How does twine prevent this issue from getting worse?

# AVAILABILITY & RELIABILITY

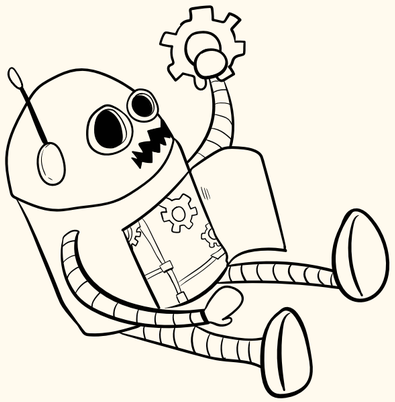
# DESIGN PRINCIPLES

## AVAILABILITY AND RELIABILITY – DESIGN PRINCIPLES

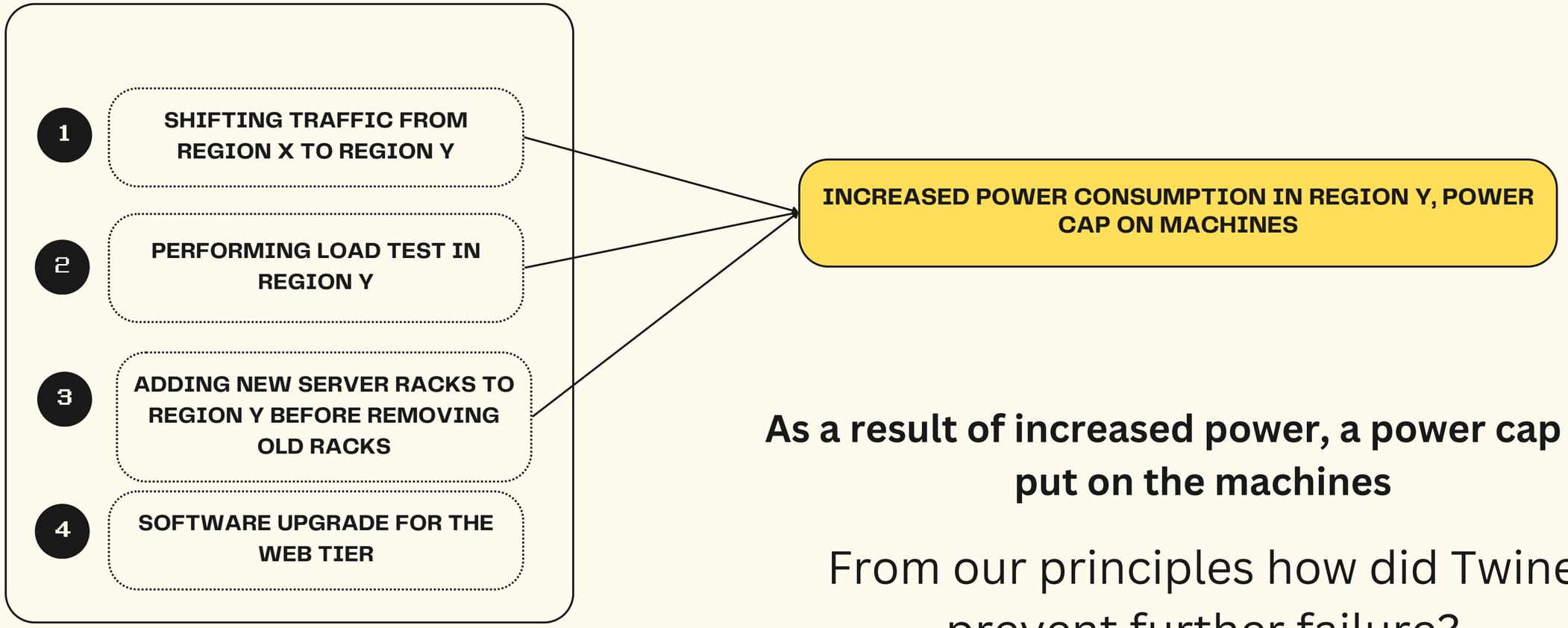
1. ALL COMPONENTS ARE SHARDED
2. ALL COMPONENTS ARE REPLICATED
3. TASKS KEEP RUNNING
4. RATE LIMIT DESTRUCTIVE OPERATIONS
5. NETWORK REDUDANCY

# AVAILABILITY AND RELIABILITY

## OPERATIONAL PRINCIPLES



1. TWINE MANAGES ITSELF
2. TWINE MANAGES ITS DEPENDENCIES
3. GRADUAL BUT FREQUENT SOFTWARE RELEASE
4. RECURRING LARGE-SCALE FAILURE TEST




**As a result of increased power, a power cap was put on the machines**

From our principles how did Twine prevent further failure?

# AVAILABILITY & RELIABILITY

1. SHIFTING TRAFFIC FROM REGION X TO REGION Y
2. PERFORMING LOAD TEST IN REGION Y
3. ADDING NEW SERVER RACKS TO REGION Y BEFORE REMOVING OLD RACKS
4. SOFTWARE UPGRADE FOR THE WEB TIER

**INCREASED POWER CONSUMPTION IN REGION Y, POWER CAP ON MACHINES**

As a result of increased power, a power cap was put on the machines

**SCHEDULER RATE-LIMITED TASKS MOVING AWAY FROM THE MACHINE**

**HALTING THE SOFTWARE UPGRADE**

# AVAILABILITY & RELIABILITY

# Evaluation

- How does TaskControl deal with complex scenarios impacting availability?
- How effective is autoscaling for production?
- How effective are host profiles in improving performance?
- What is the overhead of switching host profiles?
- How cost effective are small machines in replacing big machines?

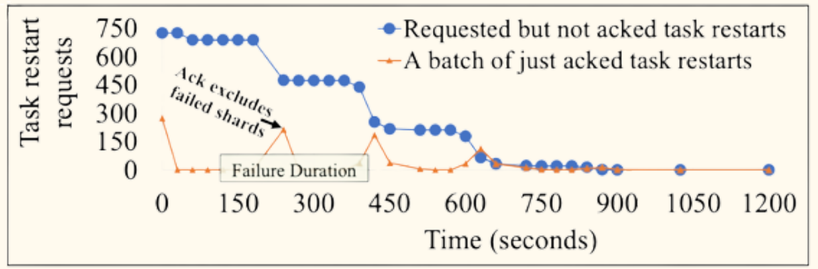
# TASK CONTROL AND AVAILABILITY

UP NEXT: AUTOSCALING

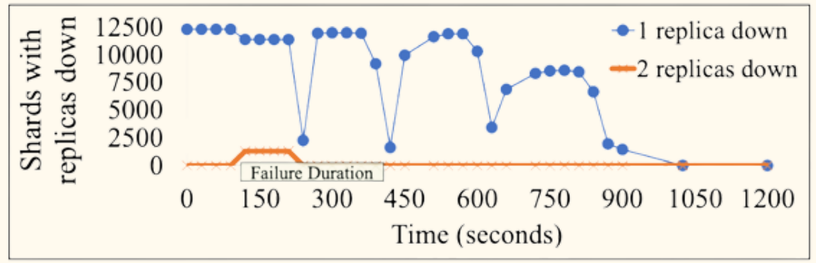
1. SOFTWARE RELEASE AND MACHINE FAILURES CONCURRENTLY

2. USES A CACHING SERVICE MANAGED BY SHARD MANAGER

3. 15000 SHARDS PER PARTITION
   EACH SHARD HAS 3 REPLICAS
   REPLICAS HOSTED BY 1000 TWINE TASKS




(a) Acknowledged and pending task-restart requests.

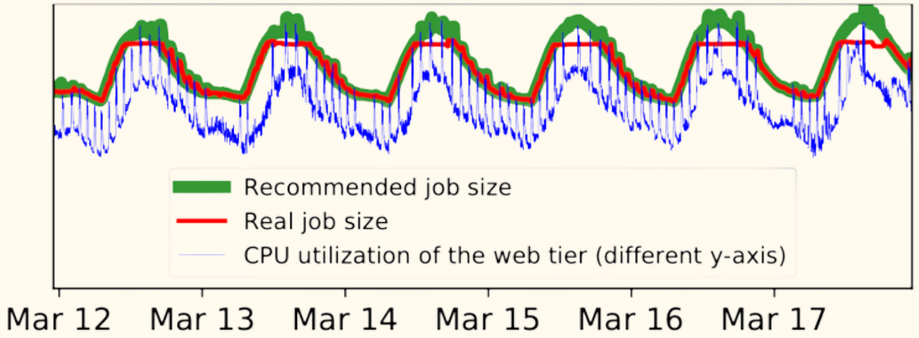


# AUTOSCALING

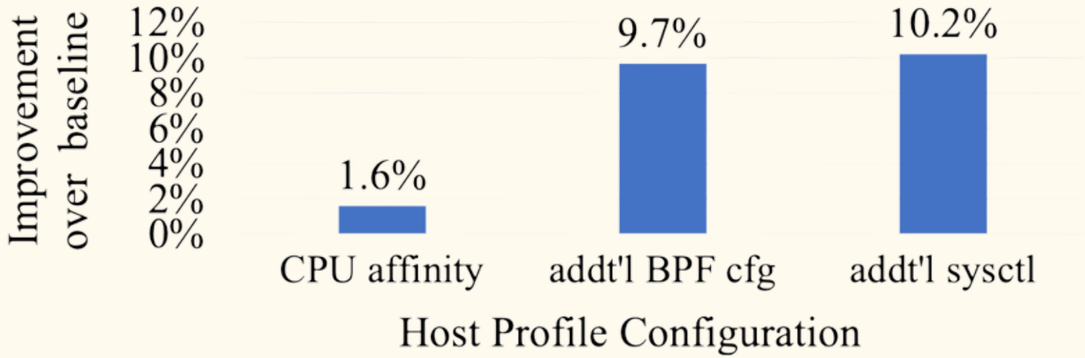
Up next : Host Profile and Performance

1. 800 SERVICES AUTOSCALED
2. 25% OF MACHINES FREED UP DURING OFF-PEAK HOURS
3. CPU UTILISATION CLOSELY FOLLOWS THE RECOMMENDED JOB SIZE
4. COVID CAUSED A CAPACITY STORAGE

TaskController automatically reduce heavy-capacity jobs



# Host Profiles and Performance




1 **HOST PROFILES ALLOW FOR CUSTOMISATION LEADING TO BETTER PERFORMANCE**

E.g avoids unnecessary interrupts and context switches

# HOST PROFILES AND PERFORMANCE

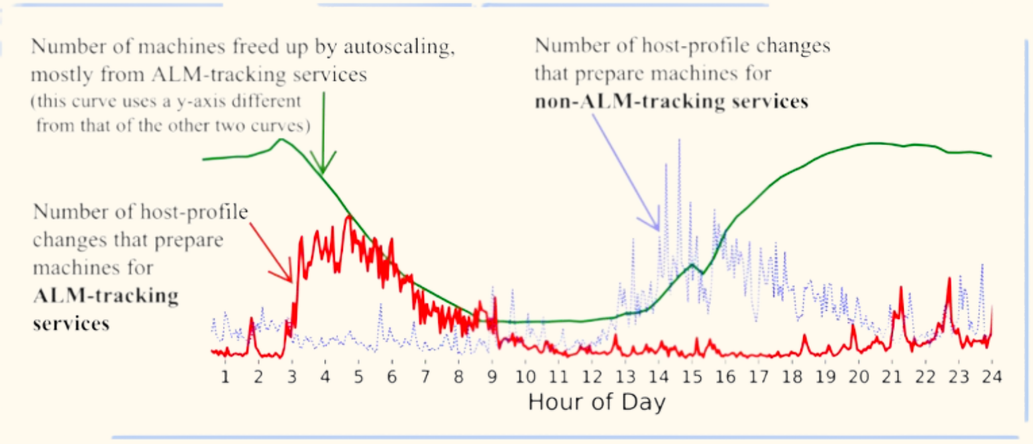
UP NEXT: POWER EFFICIENT SMALL MACHINES

1. **TOO MANY HOST PROFILE SWITCHES REDUCES PERFORMANCE**

E.g avoids unnecessary interrupts and context switches

2. **MEMORY FRAGMENTATION CAUSES ALLOCATION FAILURES**

3. **A FUTURE SOLUTION TO THIS IS A KERNEL IMPROVEMENT**
**THE AVERAGE PROFILE CHANGE IS EVERY 2 DAYS, SO THIS OVERHEAD IS NEGLIGIBLE**

# POWER-EFFICIENT SMALL MACHINES

**OK GOOGLE: HOW MANY SMALL MACHINES CAN MATCH THE PROCESSING OF A LARGE ONE?**

**Lowest TCO:**

- **Shard manager can achieve a low relative cost with shard migration**
- **Small machines have to lead to better CPU use with the exact cost**

**Overall 83% RTCO lead to 17% fleet-wide savings**

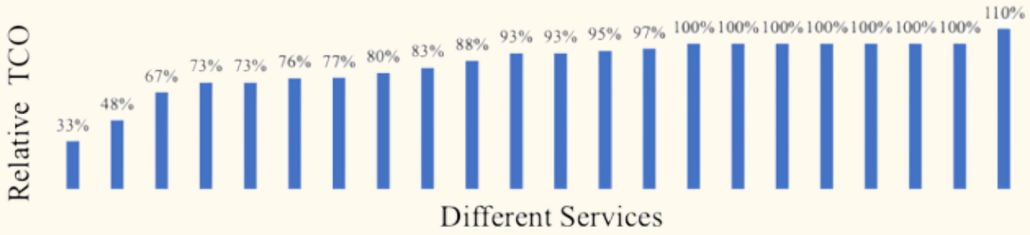
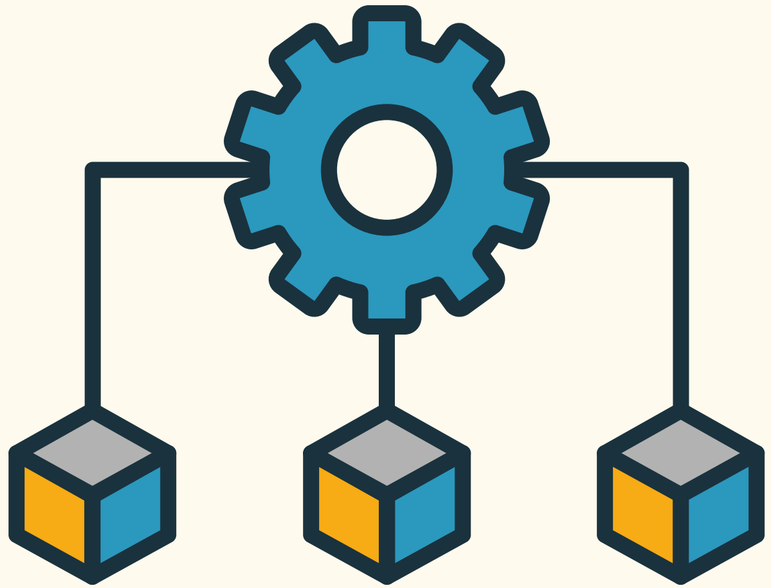



**Figure 19:** The relative total cost of ownership of services running on small machines vs. on big machines. Smaller numbers mean bigger savings.

# Experience with Shared Infrastructure



# EXPERIENCE <-> SHARED INFRASTUCTURE

UP NEXT: AUTO-SCALING

## COST OPTIMIZATION WITH TWSHARED

**PROVIDES ECONOMIES OF SCALE BY REDUCING HARDWARE AND DEVELOPMENT COSTS AND INCREASING THE UTILIZATION OF UNDERLYING RESOURCES**

1. **CAPACITY BUFFER CONSOLIDATION** → **COMBINING BUFFERS USED IN FOR INDIVIDUAL PURPOSES INTO A CENTRALIZED BUFFER.**

2. **TURBO BOOSTING** → **TURBO BOOSTING ALL THE CORES AND USING REBALANCER TO MAXIMISE UTILIZATION.**

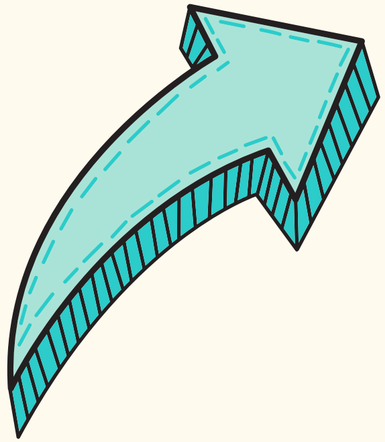
3

# AUTOSCALING

EASILY FREEING UP OVER-PROVISIONED CAPACITY DURING OFF-PEAK HOURS PROVIDES IT WITH THE OPPORTUNISTIC CAPACITY FOR OTHER WORKLOADS TO INCREASE MAXIMUM UTILIZATION.

# So what was hold-up ?

**MAXIMUM UTILIZATION IS STILL NOT ACHIEVED BECAUSE TWINE HAS NOT YET IMPLEMENTED SERVICE-LEVEL OBJECTIVES(SLOS).**



# JOURNEY TO SHARED INFRASTRUCTURE

**NO LARGE COMPANY ACHIEVED 100% SHARED INFRASTRUCTURE CONSOLIDATION**

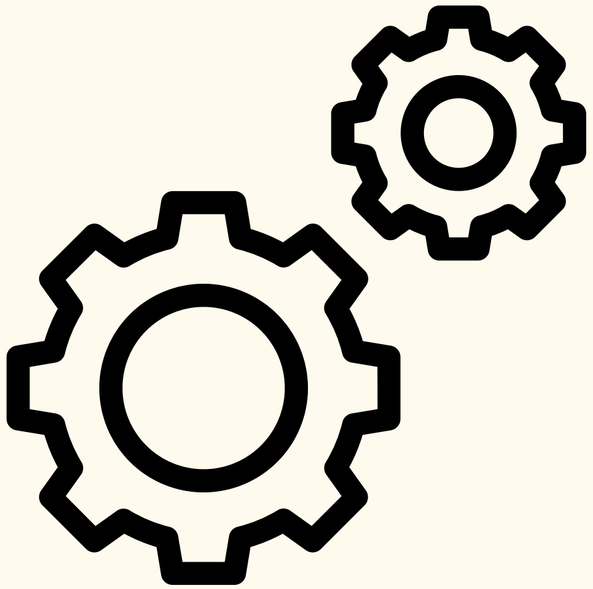
**STRATEGIES TOWARDS MIGRATING ALL COMPUTE LOADS TO TWSHARED AND PUBLICIZING THE SAME**

**BUILDING TWINE TO SUPPORT A LARGE COMPUTE POOL**

**USING SCALABILITY, ENTITLEMENTS, HOST PROFILES AND TASK CONTROL**

1

# BUILDING TWINE TO SUPPORT A LARGE COMPUTE POOL



**TASKCONTROL AND HOST PROFILE ENSURE THAT TWSHARED CAN SUPPORT**

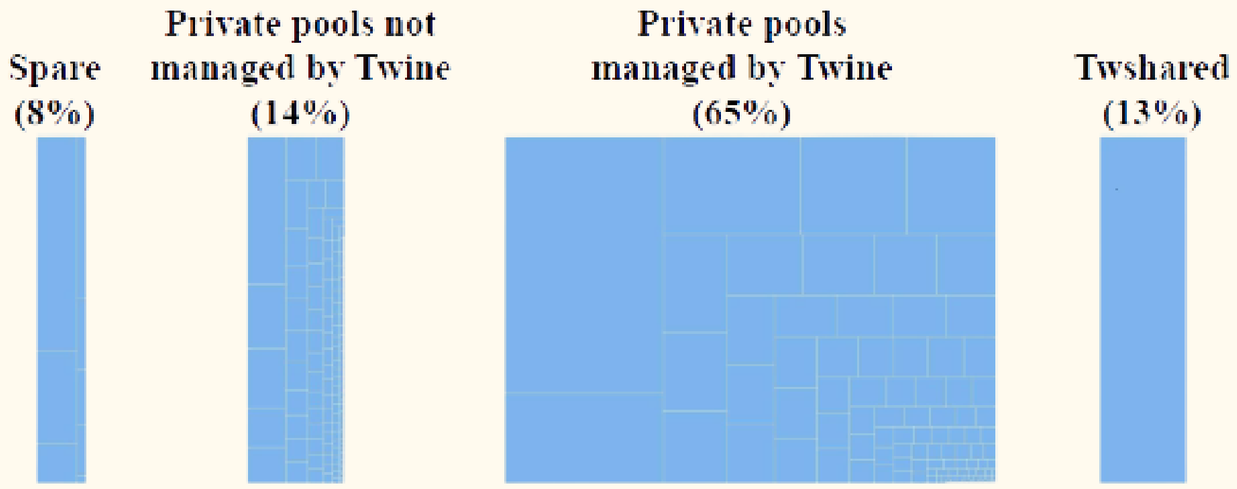
1 **GENERAL NEEDS OF THOUSANDS OF SERVICES**

2 **SPECIALIZED NEEDS OF A SMALLER TAXING SET.**

# Publicize the growth and health of twshared

**PROVIDED THE PUBLIC WITH THE REAL–TIME BREAKDOWN OF THE FLEET AND GROWTH OF TWSHARED.**

**The breakdown evolved into a percentage figure of [13%, 5%, 26% 56%] respectively by 2020, showing successful ongoing migration of the majority of services to twshared.**




**Breakdown of the fleet as in August 2018**

3

# MOTIVATING THE COMPANY FOR MIGRATION

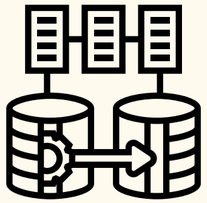
1. INITIALLY TARGETING THE WEB TIER, THE LARGEST PRIVATE POOL.

2. DIRECTLY SERVING THE USERS, THUS OUTAGES ARE NOTICED QUICKLY.

3. SMOOTH , MONITORED MIGRATION MOTIVATED THE REST OF THE TEAM TO FOLLOW.

4

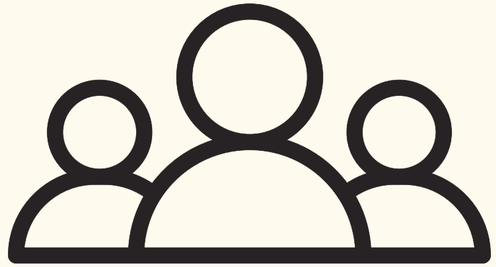
# MANDATORY MIGRATION



ESTABLISHED THAT ALL NEW COMPUTE CAPACITY WILL LAND ONLY IN TWSHARED.

THESE STEPS ALONG WITH TWINE'S CAPABILITY OF SUPPORTING CUSTOMIZATION THROUGH TASKCONTROL AND HOST PROFILING MADE TWSHARED, THE COMPANY'S UBIQUITOUS COMPUTE POOL.

# THE CASE OF PG_X



MIGRATION OF A PG_X WITH DIVERSE FIELDS OF SERVICES USING UP TENS OF THOUSAND OF COMPUTE POWER IN PRIVATE POOL

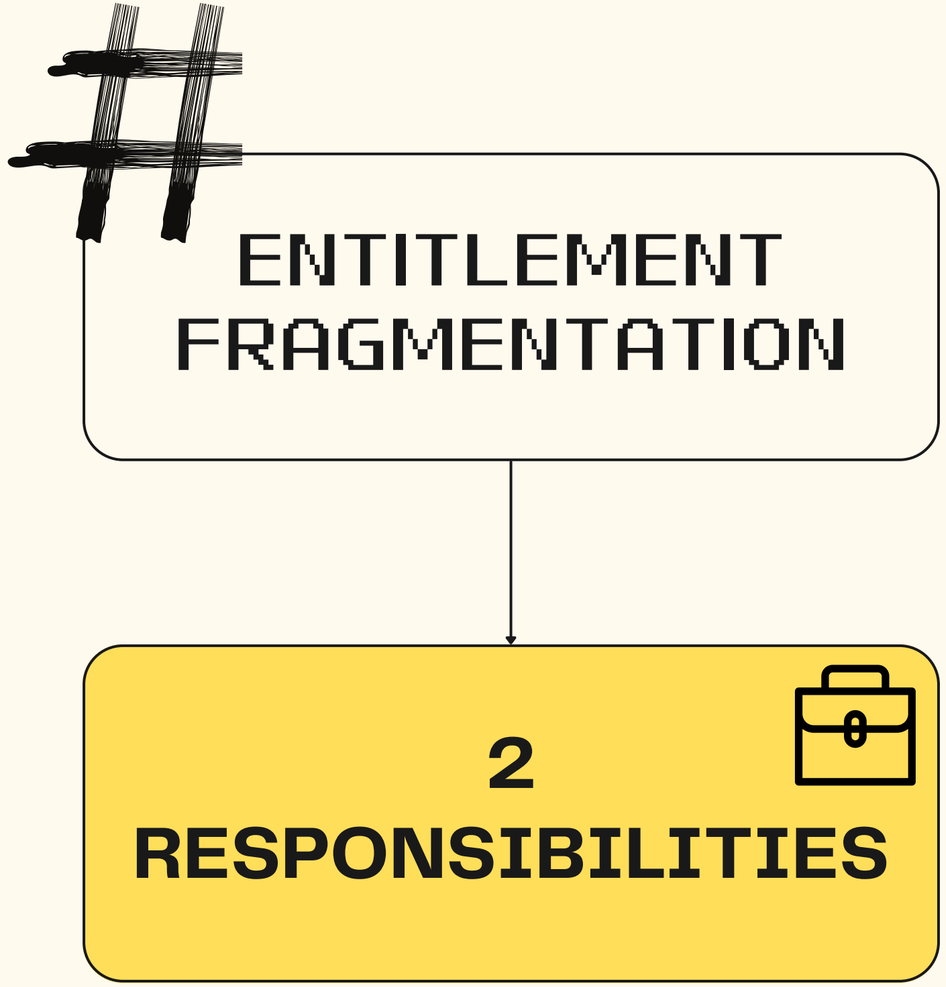
shows off

TWSHARED WAS ABLE TO ACCOMMODATE WORKLOAD SPECIFIC REQUIREMENTS

PG_X'S SUCCESS IN USING OPPORTUNISTIC CAPACITY AT A LARGE SCALE

MOTIVATION FOR MIGRATION AND SLO DEVELOPMENT

# Lessons Learned



# Fleet partitioning

01

PARTITIONING MILLIONS OF MACHINES INTO SMALLER UNITS, EFFECTIVELY MANAGED BY SCHEDULER SHARDS.

02

TWINE JOBS CAN ONLY STACK IN THE SAME ENTITLEMENT IMPLYING THAT EACH ENTITLEMENT IS SIZED FOR A FEW THOUSAND OF MACHINES.

# Quota management

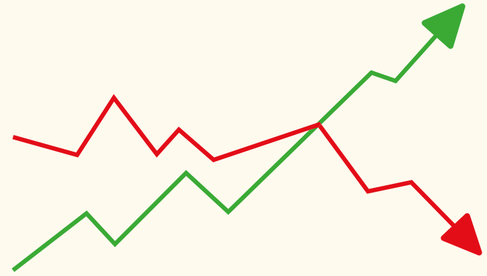
**01**

**LEVERAGING ENTITLEMENT FOR QUOTA MANAGEMENT RESULTS IN SMALL ENTITLEMENTS.**

**02**

**PROTECTING THE QUOTA FROM AN UNEXPECTED GROWTH OF ROGUE SERVICES.**

THE TRADE-OFF BETWEEN THEM LEAD TO THE WORK ON MATERIALIZATION FOR FLEET PARTITION AND A STACKABLE RESERVATION FOR QUOTA MANAGEMENT.



# CONTROLLED CUSTOMIZATION

UP NEXT: SUPPORTING GLOBAL SERVICES

**CUSTOMIZATION WAS THE KEY TO MIGRATING THE SERVICES OVER TO TWSHARED.**

E.g without host profiling and TaskControl web tier and memcache services would not run as expected in twshared

**CURRENTLY, 17 HOST PROFILES AND 16 TASKCONTROLLER SUPPORT THOUSANDS OF SERVICES TO PERFORM TO THEIR MAXIMUM UTILIZATION.**

**INITIAL CUSTOMIZATION ON INDIVIDUAL JOB TASKS THAT ARE IDENTICAL BY DEFAULT WERE ABUSED BY DEVELOPERS.**

This broke fleet – wide customization

# SUPPORTING GLOBAL SERVICES

DEVELOPERS WISH TO RUN THEIR SERVICES GLOBAL WITHOUT WORRYING ABOUT THE REGION AND THE CAPACITY REQUIREMENT

MULTIPLE GLOBAL TWINE DEPLOYMENTS THAT SPREAD A GLOBAL JOB'S TASK ACROSS A SMALL REGION WERE EMPLOYED FOR THIS.

# SUPPORTING GLOBAL SERVICES

UP NEXT: CHALLENGES WITH SMALL MACHINES

1. MACHINES IN A REGION ARE FUNGIBLE DUE TO HIGH NETWORK BANDWIDTH AND LOW LATENCY WITHIN THE REGION.
2. IT IS NOT THE CASE FOR MACHINES DISTRIBUTED ACROSS MULTIPLE REGIONS.
3. DECOMPOSE THE GLOBAL SERVICES INTO THE CAPACITY NEEDS OF SPECIFIC REGIONS AS OPPOSED TO THE GLOBAL ALLOCATORS
4. A FEDERAL SYSTEM TO SUPPORT THE SAME ON-TOP OF GLOBAL-SERVICE WAS DEPLOYED

**MISCALCULATED ABSTRACTION**

# The Story with small machines

01

EFFORT TO RE-ARCHITECT AND REIMPLEMENT MEMORY-CAPACITY-BOUND SERVICES WAS HIGHER THAN ANTICIPATED

02

MOVING FROM STATIC SHARDING TO DYNAMIC SHARDING IMPROVING LOAD BALANCING

03

ACHIEVING 18% POWER EFFICIENCY, FOR SMALLER MACHINES MOTIVATES THE PURSE

04

THE COMPANY IS NOW TRYING TO WORK THE HARDWARE OF SMALLER MACHINES FOR INTERNAL WORKLOAD MANAGEMENT.

# FACTORS THAT LED TO THIS ADOPTION OF SMALLER MACHINES

THE LEGACY LARGE SERVICES WERE OPTIMIZED FOR UTILIZING ENTIRE MACHINES RUNNING IN PRIVATE POOLS

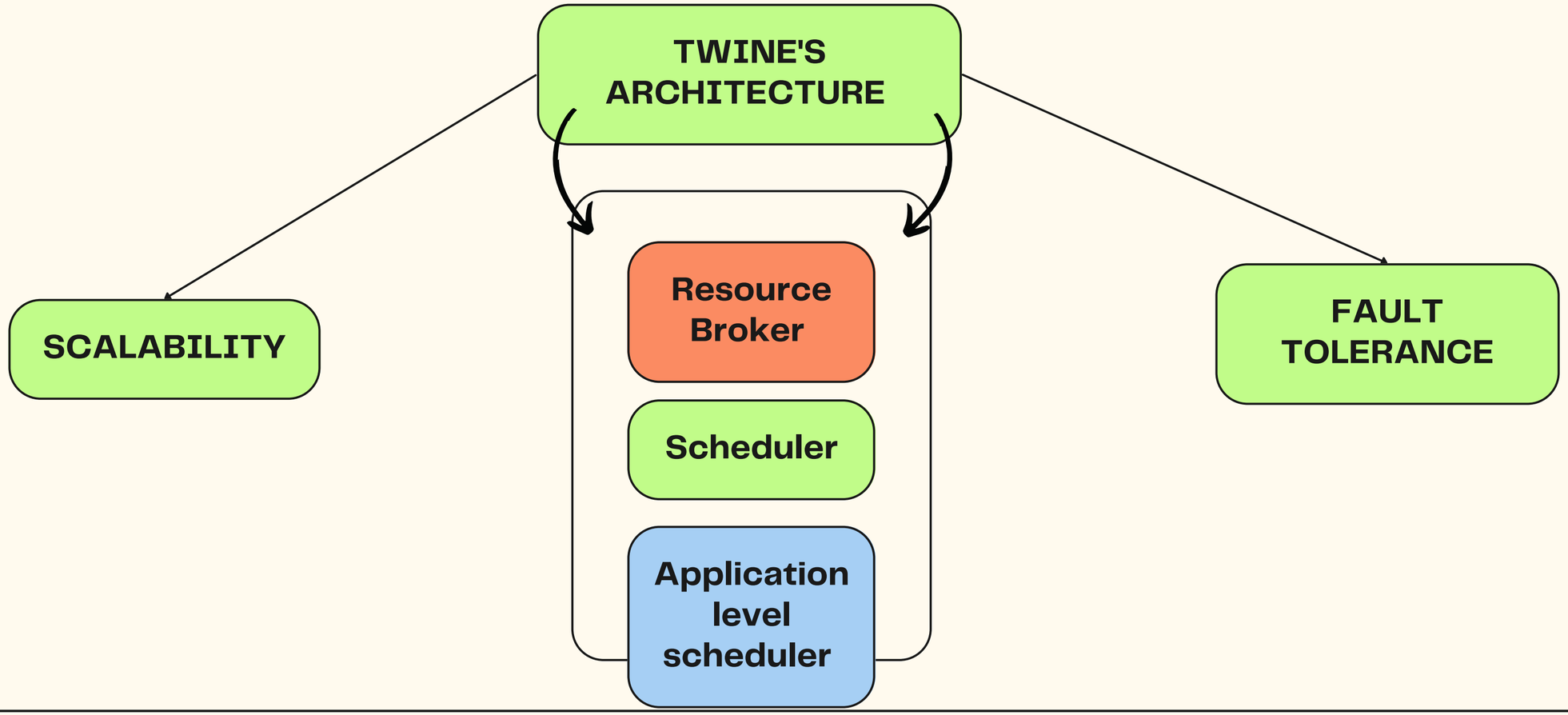
THE STACKING TECHNOLOGY NEEDED TO MATURE AND IMPROVE SUPPORT FOR PERFORMANCE ISOLATION

**Did you know: Twine is available on GitHub under the Apache 2.0 license and is actively maintained by Uber Engineering and the open-source community?**

# RELATED WORKS

Did you know: that Twine supports Kubernetes as well as other container orchestration systems, such as Docker Swarm and Apache Mesos?

## 1 ENTITLEMENTS



# 2 AUTOSCALER

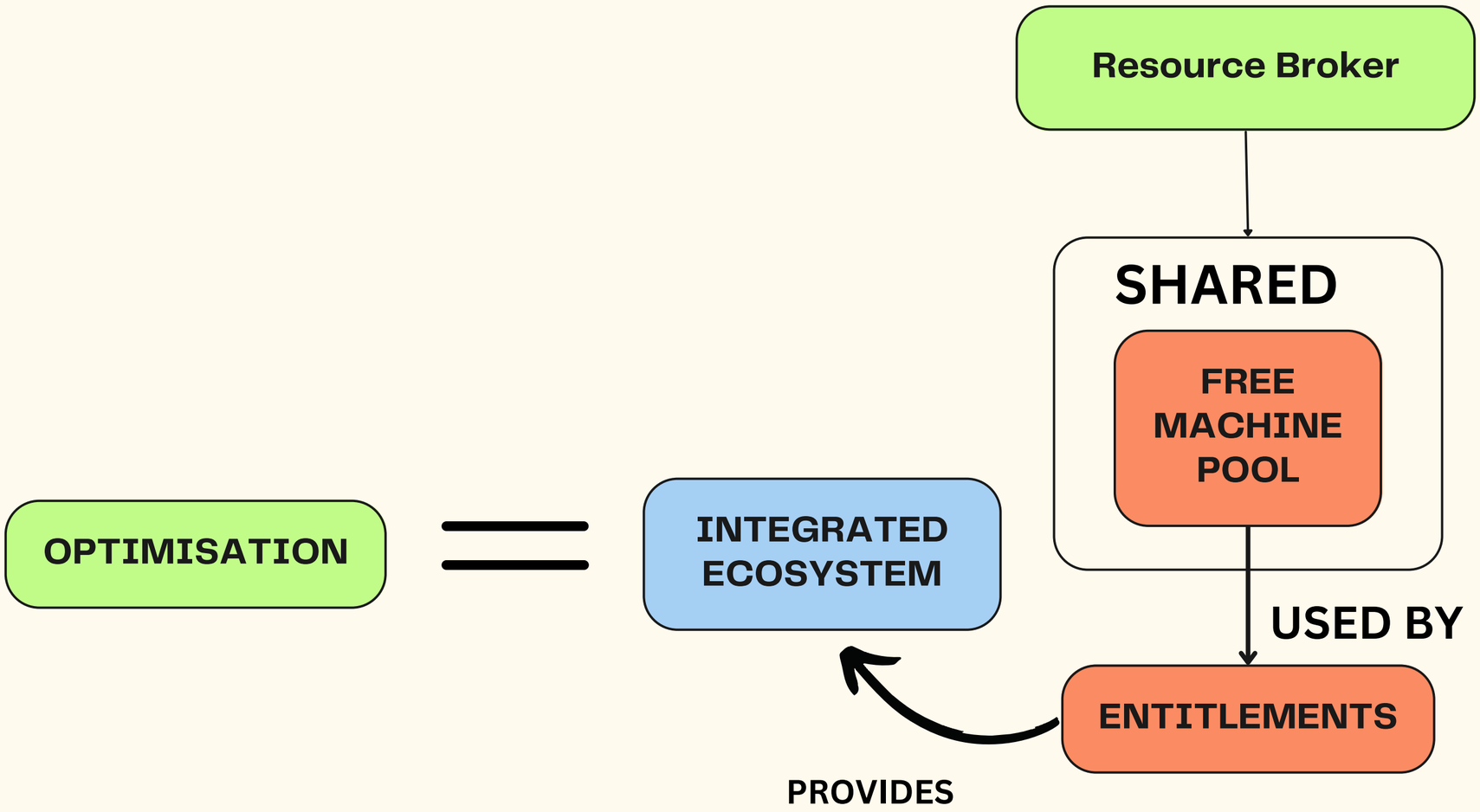
DID YOU KNOW: TWINE IS BUILT ON TOP OF A DISTRIBUTED DATABASE CALLED APACHE CASSANDRA, WHICH PROVIDES HIGH AVAILABILITY AND SCALABILITY FOR MANAGING CLUSTER METADATA.

## KUBERNETES' CLUSTER AUTOSCALER

Is designed to respond to workload growth by provisioning new virtual machines (VMs) in a public cloud environment and adding them to a node pool.

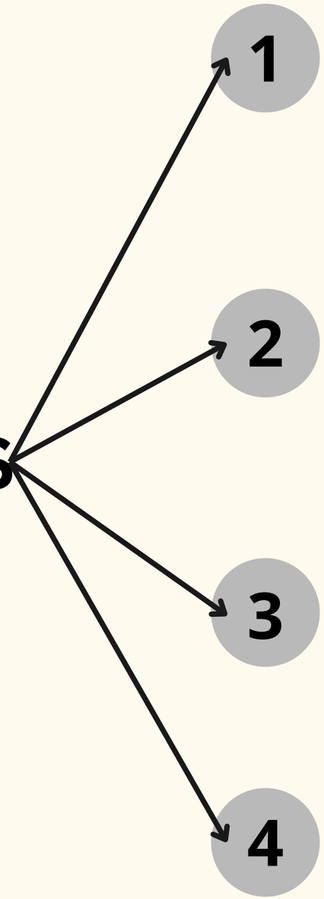
This resizable node pool in Kubernetes corresponds to Twine's entitlements, which are resource offers that can be allocated to different applications or workloads.

# BUT IN TWINE


Resource Broker
SHARED
FREE MACHINE POOL
USED BY
ENTITLEMENTS
PROVIDES
INTEGRATED ECOSYSTEM
OPTIMISATION

# Conclusion

## MAIN CONTRIBUTIONS



1. ABILITY TO MANAGE A LARGE-SCALE SHARED INFRASTRUCTURE OF ONE MILLION MACHINES ACROSS PHYSICAL CLUSTERS.
2. COLLABORATE WITH APPLICATIONS TO MANAGE THEIR LIFECYCLE.
3. SUPPORT HOST CUSTOMIZATION IN A SHARED POOL
4. EMPLOY AUTOSCALING TO IMPROVE MACHINE UTILIZATION.

# LIMITATIONS OF EXISTING SYSTEMS

1. **INFLEXIBILITY**
2. **LACK OF SUPPORT FOR DYNAMIC RESOURCE ALLOCATION**
3. **INABILITY TO MANAGE LARGE-SCALE SHARED INFRASTRUCTURE**

# TWINE = PROMISING SOLUTION

1 **Features**

2 **Performance**

3 **Scalability**

**TWINE CAN SIGNIFICANTLY IMPROVE THE EFFICIENCY AND UTILIZATION OF SHARED RESOURCES WHILE SIMPLIFYING MANAGEMENT FOR ADMINISTRATORS AND USERS.**
**~ META ENGINEERS**


THANK
YOU!

“

# View the slides independently

shorturl.at/fnru6

OR