

Metrics-driven Infrastructure Efficiency

Akash Deshpande, CEO, Sky Blue AI

Overview

This document summarizes the practical experience of metrics-driven optimization of planet-scale infrastructure and the methods that were developed and refined based on this experience over several years.

The goal of Infrastructure Efficiency is to deploy the right amount of infrastructure at the right time and place to support and grow services that run on it without wasteful capital or operational expenditure.

Services make many kinds of demands on the infrastructure, such as scale, lead time, reliability in the face of failures, resiliency to support multiple use cases, responsive and predictable performance, and more. Efficient infrastructure not only serves tactical business objectives such as profitability but also, when served at scale, enables new technologies and user experiences, changing the world in unanticipated ways. For example, caching in content delivery networks reduces both latency and cost, enabling superior user experience at a planetary scale.

The importance of Infrastructure Efficiency as a discipline is rising due to a combination of trends. During high demand growth, excess infrastructure is quickly soaked up and the unit cost (\$/usage) of infrastructure drops rapidly because of increased usage and decreased technology cost. As demand growth and technology improvements taper off, infrastructure unit cost declines are more gradual, requiring a comprehensive, multi-faceted approach.

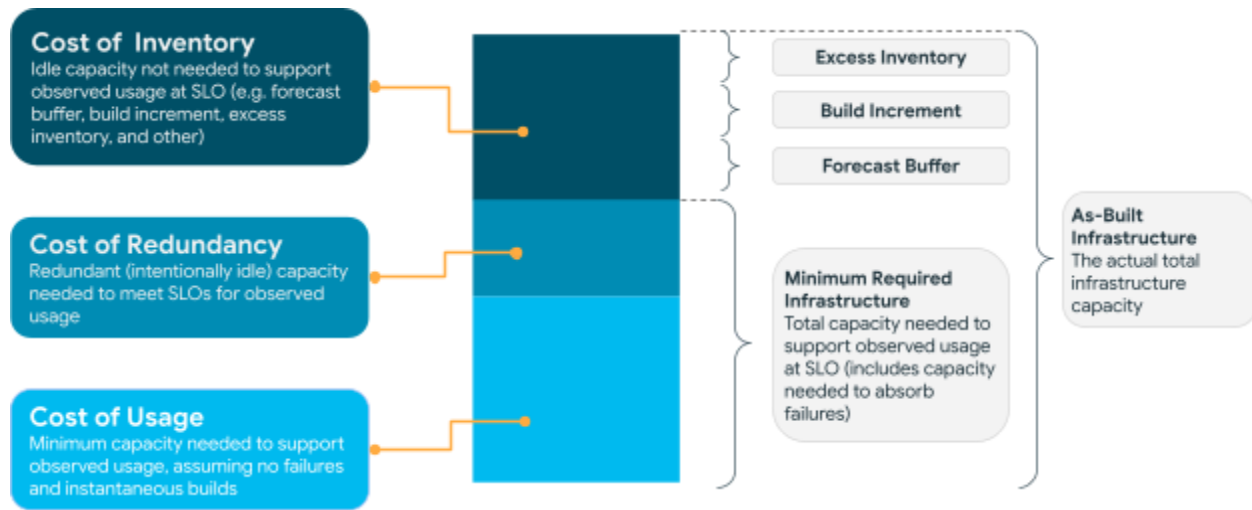
This document presents a metrics-driven approach to Infrastructure Efficiency. Infrastructure Efficiency requires multi-disciplinary skills including software engineering, optimization, data science, machine learning, finance, accounting, and program management. Besides metrics, Infrastructure Efficiency processes include business case reviews for both mandated and efficiency programs, and efficiency projects that cut across functional organizations. Since the incentives of an Infrastructure organization are often aligned to deployment scale (and not efficiency), the Infrastructure Efficiency group is best positioned outside the deploy organization or even outside the overall Infrastructure organization, say in a strategy or financial control function.

While the examples in this document are drawn from IT infrastructure, the concepts apply to any infrastructure such as transportation, utilities, etc.

Infrastructure Efficiency Metrics

The three main infrastructure efficiency metrics are Cost of Usage (CoU), Cost of Redundancy (CoR), and Cost of Inventory (CoI). These are measured in dollars, and they add up to 100% of infrastructure spend during the period of interest (quarter, year, etc.). Each is broken down into sub-metrics for fine-grained tracking and optimization.

The following diagram illustrates the efficiency metrics and their explanations.



Cost of Usage (CoU)

CoU is the cost of the infrastructure that is required under completely ideal conditions where we know exactly what usage will land on the infrastructure, that no part of the infrastructure ever experiences failure, and we can build the infrastructure instantaneously. Under these ideal conditions, what is the minimal (greenfield) infrastructure we need to support the observed usage? CoU is defined as the cost of this minimal ideal infrastructure.

Cost of Redundancy (CoR)

We now make one of the ideal conditions real, viz., parts of the infrastructure can fail from time to time. To meet Service Level Objectives (SLOs) despite these failures, we need to build more infrastructure that is intentionally not used (either as backup systems in active/backup configurations or as headroom in active/active configurations). CoR is defined as the incremental cost of this intentionally idle redundant infrastructure.

Cost of Inventory (CoI)

Finally, we make another ideal condition real, viz., we cannot build the infrastructure instantaneously. Infrastructure lead times can be long and variable. For example, in an IT

environment, data center space and transoceanic fiber can take three to five years, regional and metropolitan infrastructure can take one to two years, and populating campuses and data centers can take several weeks or months. Hence we have to forecast usage at different aggregation levels at different lead times. Forecast Buffer (FB) is the incremental cost of the infrastructure deployed to absorb a certain fractile of the forecasted demand distribution (as prescribed by a newsvendor model). There are certain minimum size requirements when adding capacity, leading to extra cost attributable to Build Increments (BI). Finally, Excess Inventory (EI) may be acquired as a hedge or simply because good deals become available in the market. All of these costs from FB, BI, EI, and other causes are counted in Col.

Managing CoU

Supply Levers for CoU

Supply side can affect how the demand matrix is measured and aggregated, how the cross-correlation properties of usage can help oversubscribe the infrastructure, how the infrastructure cost is allocated towards components that support the demand, how the infrastructure is architected, and how the capacity delivery is planned. This section gives an overview of these levers.

Usage Measurement and Aggregation

Since cost of usage is based on observed demand, how demand is measured and aggregated affects this metric. In the IT scenario, at the most granular level, machine usage arises from tasks running on CPUs, storage usage arises from IO operations, and network usage arises from application bits sent on transport connections between machines. At this granularity, demand is typically spiky, unpredictable, and small (compared to the overall demand). This granular demand is aggregated at various levels such as a cluster of machines or a campus of data centers, or even higher such as a metro or a subcontinent, and time scales by summarizing it using statistical measures such as average or fractiles over periods of different lengths.

Some usage such as user queries and their responses or highly reactive real-time tasks such as critical infrastructure control messages must be carried immediately while other usage such as backups or log writes can be delayed. The demand measurement and aggregation pipeline uses a high percentile (say 95th) for non-delayable usage over short time periods and average (mean) demand over long time periods for delayable usage with appropriate time constants to reflect the immediacy of their respective infrastructure requirements.

Demand aggregation and capacity planning methods must match. For example, constant or variable or deadline-based demand must have a corresponding planning method. While variable demand measurement and planning methods can be more accurate and efficient, they

are more complex to reason about and operationalize. Hence both demand and planning often use constant usage profiles.

Oversubscription

Oversubscription makes use of the statistical properties cross-correlating different usages that make up the demand ensemble to fit more usage into the deployed infrastructure capacity. Simply adding the demand characterizations of two usage patterns gives us a "sum of peaks" of the joint usage pattern. However, their peaks typically don't occur at the same time. What we need is a "peak of sums" of the joint usage pattern. Generalizing to a large number of usage patterns that form the demand ensemble, oversubscription computes safe ratios of sum of peaks to peak of sums for sections of the demand.

Oversubscription ratios can be used in two ways: deflating the demand incident on the infrastructure, thereby inducing a smaller planned capacity, or inflating the built capacity of the infrastructure and accommodating more demand. Since the mapping of usage to infrastructure can change depending on failure conditions, the capacity inflation approach can result in a mismatch leading to unplanned congestion. The demand deflation approach leads to better capacity matching under both normal and failure conditions.

Efficient Infrastructure Architecture

Performance and cost of infrastructure are significantly influenced by the architecture chosen to support its use cases. Taking two extreme examples, a full communication mesh between all nodes performs the best with respect to latency but is inefficient with respect to cost, and a single shared bus connecting all the nodes can be cost-efficient under certain sharing regimes but offers poor latency performance.

Optimized Capacity Planning Tools

Capacity planning takes as input the demand ensemble with required performance characteristics, the architecture patterns to construct the infrastructure, the policies for assigning load, the existing infrastructure footprint, and the cost of the components used to expand or contract the infrastructure, and produces a capacity plan that meets the demand at the lowest cost. Various algorithms and policy tradeoffs can be used for planning from augmenting components with observed high usage to mixed integer and linear programming for capacity optimization to ML-based planning.

Demand Levers for CoU

Services can influence cost of usage by regulating how the infrastructure is used in real time, how different categories of use cases are handled in the service designs, and by architecting services taking into account infrastructure performance and cost characteristics.

Real-time Demand Shaping

Real-time demand shaping refers to the fine-grained control over scheduling of jobs and communication in distributed systems. This can be done through a choice of machine and storage locations, and infrastructure components such as load balancers and communication channels. In addition, a service may choose to back off on RPC calls in the face of delays to relieve congestion or cancel and retry calls that are taking too long. The overall goal is to prevent congestion collapse on one hand and promote smooth near-instantaneous API calls on the other. Real-time price signals given to services can promote demand shaping.

Peak Shaving

When service functions are bursty but not urgent (log writes, for example), the service can delay the function so there is a lower peak to average load ratio.

Efficient Service Architecture

Service architecture has an outsized impact on not only infrastructure usage but also on the general service performance itself. For example, edge caching in content delivery networks reduces long haul bandwidth usage and improves responsiveness to users, leading to both lower cost and higher customer satisfaction. Similarly, properly tiered and positioned load balancers, security proxies, consensus mechanisms, gang scheduled tasks and other patterns have significant impact on infrastructure usage and performance requirements.

Managing Cost of Redundancy

Supply Levers for CoR

Supply can affect the cost of redundancy by providing infrastructure products whose performance characteristics better match service use case requirements, developing accurate failure models, improving asset availability, reducing repair times for failures, and signaling infrastructure prices to PAs for improved service planning.

Infrastructure Product Definition

The goal of infrastructure products is to encapsulate the complexities of technology into an easy-to-consume API that matches the requirements of services and service planners. Product definition has several aspects such as how the service endpoints are defined, how load is characterized, and how performance is specified. Depending on use cases, endpoints can be a single machine or VM, or groups of them such as a cluster of machines, a campus, a metro or others. Compute characteristics cover properties such as job completion times, duty cycle, fan-out and fan-in, cache hit ratio, etc., and storage characteristics cover properties such as IOPS, seek v. read/write duty cycle, bandwidth, and location. Traffic characteristics cover properties such as bandwidth, packet size distribution, protocol and application mix, time of day

and day of week distribution, stream length distribution, inter-arrival time distribution, and burst size distribution. Performance covers properties such as availability, goodput, latency and jitter, and is expressed in terms of the SLOs for the infrastructure.

Choosing a judicious set of product properties gives us confidence that services will perform well when the infrastructure is optimized to deliver the defined products.

Another benefit of defining product SLOs is that it informs a better split between development and operational activities and hence their team sizes. For example, an availability SLO of x -9s allows a down time of y minutes per month. How should this error budget be spent? At one extreme, the organization can decide to staff up large operations teams that can respond rapidly to each incident. At the other extreme, the development team can create reliable solutions for each probable cause of an outage. In practice, an optimal organization will share the SLO error budget between the development and operations teams so the infrastructure systems are sufficiently reliable, and for the remaining SLO-threatening incidents, the operations team has automated mechanisms to detect and mitigate rapidly and then analyze and fix root causes soon after.

Failure Modeling

Failure models inform infrastructure planning for redundant capacity required to meet SLOs despite failures. Failures are rare events and their distribution needs to be discovered by long trails of observations. Small changes in failure characteristics can cause large changes in planned capacity or large misses in SLOs. Hence, developing accurate failure models for components and systems, and combining them according to the architectural dependency graph is important for infrastructure optimization.

Asset Availability (MTBF) Improvement

If components used in the infrastructure fail less frequently or less drastically (i.e., with smaller blast radius), redundant capacity needed for SLO compliance reduces. This does not mean that the goal is to use only failure-free components, which is unrealistic and can be expensive. There is a cost tradeoff between improving asset availability and working around failures with redundancy. Components that have widespread usage should be engineered more for high availability and correctness by design (consider logic gates in semiconductor chips – unreliable gates will induce enormous cost in every system that uses them).

MTTD/MTTM/MTTR Reduction

If failed components can be detected (D), mitigated (M), and recovered (R) back into service faster, redundant capacity needed for SLO compliance will be reduced. In particular, short Detect and Mitigate times are crucial for improved user experience.

Pricing

Prices serve multiple functions in a market. As a lever for infrastructure optimization, prices signal tradeoffs between products to help services choose the most cost-efficient match to their performance requirements. Prices can also signal the potential for stockout or future congestion due to supply bottlenecks or capacity constraints. Real-time prices can signal current congestion bottlenecks.

Demand Levers for CoR

Services can affect the cost of redundancy by choosing appropriate SLOs for their service traffic and creating a demand mix that drives higher utilization of infrastructure capacity.

Service SLOs

Services can affect the cost of redundancy by matching the infrastructure products used to their performance requirements. This includes upgrading or downgrading existing SLOs as well as architecting services to require a different SLO profile.

Service Mix

When the infrastructure is built to support a certain SLO, it can accommodate some level of lower SLO load without incremental cost. The availability of this mix of SLOs for the same cost can inform service architectures and their infrastructure usage.

Managing Cost of Inventory

Supply Levers for Col

Supply can affect the cost of inventory by improving demand forecast accuracy, reducing lead times, choosing optimum build increments, and grooming excess capacity from the infrastructure.

Demand Forecast Accuracy

Demand forecasting is based on three factors: historical usage trends, resource investment by services such as data center facility leases, power contracts, and machine and storage orders, and inorganic business growth through new product launches and service architecture changes. Typically, demand growth is exponential, and forecasting exponentials is notoriously hard. Demand forecasts output the distribution of load that will be seen at some time in the future, aligned with the lead time to deliver different categories of infrastructure. Demand goodness metrics track the overage and underage of the forecast with respect to usage, and need to be back tested since the forecasts can be for many quarters into the future.

Lead Time Reduction

Infrastructure deployment broadly falls into three stages: *assets* such as space and power and subsea and terrestrial fiber, *infrastructure* such as the machine, storage, and network equipment chassis, and *service* elements such as machines, drives, and line cards and optical interfaces. For product capacity delivered at a particular time, assets are deployed three to five years earlier, infrastructure is deployed one to two years earlier, and service elements are deployed within the last quarter or two.

In machine deployment, about 90% of the cost is in the machines and this cost is incurred only in the last few months before delivery. In network deployment, assets add up to about 40% of the cost and infrastructure adds up to another 25%.

Since forecasting accuracy reduces with lead time, the demand distribution is more spread out farther into the future. Covering a certain fractile of that distribution results in a larger capacity envelope that must be deployed. Hence, reducing the lead time of infrastructure deployment can reduce the spend significantly. Some factors such as supply chain management across vendors and warehouses can be influenced and compressed while other factors such as permitting and related administrative processes and the actual laying of the power grid and fibers on ocean floors can be difficult to affect.

Granular Build Increments

Build increment refers to the minimum size of additional capacity that can be deployed. With greater silicon and optical integration, minimum machine, storage, and network sizes keep increasing. Build increments can be thought of as prebuilding by a short amount of time until demand catches up and uses this capacity. Build increment cost can be managed by subdividing or virtualizing the deployed infrastructure and by progressively populating the systems over time. In the scheme of things, build increments do not contribute a significant additional cost and they are tracked only to ensure that this condition continues to hold.

Excess Infrastructure Grooming

Excess capacity may build over time for various reasons. Demand may have moved away and the capacity that was built to support that previous demand may now fall into disuse. A new bottleneck-removing technology that is deployed may attract a lot of usage and the capacity of the old technology can fall into disuse. Excess inventory that was acquired as a hedge may no longer hold any benefit.

Discovering excess requires testing the capacity against both past usage and future demand, and when it is persistently unused in both, it can be groomed out.

Demand Levers for Col

Services can affect the cost of inventory by improving demand forecast accuracy for their services, dialing in their risk level, and trading or returning quota they have acquired.

Demand Forecast Accuracy

Trending historical demand misses inorganic factors such as new product launches, opening new geographical markets, and changing service architectures. Services bring this additional layer of information to demand forecasting. However, focusing on the granular service-level forecasts loses the statistical multiplexing effect that gives better stability and accuracy to aggregate demand forecasts.

Service Risk Profile

The services' risk profiles refer to the fractiles of their demand forecast distributions that should be covered by the capacity plan. This can be reasoned using the newsvendor model to balance the lost revenue due to stockout and the cost of excess inventory.

Another aspect of the risk profile is the level of quota utilization services wish to drive. A rapidly growing business with expanding market share may wish to drive lower quota utilization to provide headroom for growth while a profit-conscious business may wish to drive higher quota utilization to control costs.

Quota Trading and Returns

As different services grow and shrink in different regions, there can be opportunities for them to trade their holdings of infrastructure products and rebalance their quotas. This can be simplified from the $O(N^2)$ exchange between services to an $O(N)$ exchange by creating a central broker (market maker) that provides liquidity through quota returns and resales.

Supporting Processes

Infrastructure efficiency requires three supporting process pillars in addition to efficiency and optimization in each functional area: regular computation of accurate efficiency metrics, business case reviews of major initiatives based on their impact on the efficiency metrics, and cross-functional efficiency programs.

Efficiency Metrics Program

Efficiency metrics should be computed daily and reviewed weekly within the Infrastructure Efficiency team, monthly with Infrastructure leads, and quarterly with executives. While these metrics are not expected to change a lot from day to day, running the pipelines frequently surfaces software and data issues early, incorporates updates to usage and capacity delivery continuously, and provides early indications of anomalies and insights, leaving time for their

analysis. Reviews should surface and explain anomalous changes in the metrics, and connect them to root causes and remedies. Computing efficiency metrics will need additional tools and pipelines, such as accurate allocation of cost of product to cost of infrastructure. For example, the same compute resource may offer both 99% uptime and 90% uptime for jobs running on it, and the same link may carry flows from A to B and from X to Y, and the cost of the resource must be allocated accurately to the products used by services. For another example, given a demand, computing the most efficient greenfield capacity plan for it may require a different tool from the production planning tool which increments from an existing infrastructure footprint and incorporates many operational rules of thumb.

Business Case Reviews

Programs above a certain threshold of investment must have approved business cases to surface and approve the impact on efficiency metrics. Programs broadly fall into two categories: mandates and optimizations. Mandates must demonstrate that the chosen approach is the better way forward in comparison to at least one other reasonable approach, and optimizations must demonstrate that they are better than the current mode of operation. Here "better" is based on financial metrics such as IRR (much) higher than the corporate discount rate and positive NPV of savings. Impact delivered by programs needs to be tracked for at least a year to ensure that the assumptions pan out and the BCR is valid. Tracking beyond a year is generally difficult since the counterfactuals become difficult to obtain.

Cross-functional Efficiency Programs

Cross-functional efficiency programs are typically discovered during root cause analysis of metrics anomalies, or by surveying multiple infrastructure teams to discover the cross-org dependencies that limit optimization. Since standard optimization techniques are likely already used in each functional area, cross-functional efficiency programs often rely on machine learning to discover efficiencies. An example of cross-functional efficiency is the answer to a question such as "what third party contracted infrastructure has seen significant underutilization over the past year". In this example, the "join" across at least three very different databases (contracts, inventory, usage) can be problematic since the "keys" that identify the infrastructure don't correlate across them. ML is used to discover the best joins across databases. Another example is the discovery of optimal configuration across multiple independent systems to optimize service use cases. Metrics-based insights and cross-team discussions can surface dozens of promising efficiency ideas. These need to be categorized based on size of impact on the metrics, time period over which the impact is delivered, and the development cost, with prioritization of obvious low hanging fruit as well as a few strategic projects.