Service Level Indicators in SLA

Sharma Vaishali, Sparago Julia, Pryce Ikuseghan, Smith Malcolm

I. Background Overview

In information technology Service Level Indicator (SLI) is a measure of the performance of a system provided by a service provider such as an outsource provider to a customer. SLIs form the basis of Service Level Objectives (SLOs), which in turn form the basis of Service Level Agreements (SLAs), an agreement between the customer and provider. An SLI is thus also called an SLA metric.

SLIs should be formed by identifying the key performance indicators that measure the product performance as they are the direct measure of the system's behavior in every stage of the business or customer satisfaction. In general, there are five different types and metrics of SLIs which help in choosing the good SLIs. Selecting good SLIs serves as the foundation for the SLI development plan. In the present work, all the key SLI types and metrics are highlighted which are useful in building the right SLI specifications and implementation, identifying the coverage gaps, and setting up the SLO targets. The work has been supported with two interesting real examples and review case studies which show how building the right SLIs could help in ensuring the user journey or customer satisfaction. A detailed project workflow has been shown in Fig. 1 which highlights the entire value chain of SLIs development and implementation.

II. TAXONOMY (CLASSIFICATION) STUDY

SLI TYPES

Availability SLI - This refers to the availability of a system that processes interactive requests between a user and a service.. Availability SLI helps to measure the availability of a service and can be implemented to also refer to the time between booting something up and making it accessible to the user (Watts, 2020 and Betsy, 2017; Fig. 2)

Latency SLI - This SLI refers to the proportion of valid requests which are served faster than the threshold. The latency SLI is therefore used to test the timeliness of responses to user requests.

Quality SLI - Quality SLI is that which specifies the quality of the response to the requests made by the user. For example, a service can promise to provide a high percentage of responses to requests without any missing backend response.

Coverage SLI - Coverage refers to the successfulness of data processed. If the service is promising users that their data would be processed and made available to them, a coverage SLI would be used to measure the success rate.

Correctness SLI - Correctness SLI refers to the proportion of valid data which gives off the correct output. For example, if the service user base is expecting that their data has been correctly generated for them and can therefore be validated to provide a statement of that correctness, a Correctness SLI would be appropriate to use.

SLI METRICS

There are essentially 5 ways to measure SLIs that can be seen below (USENIX SLO Workshop). There is no right choice for all situations but understanding the tradeoffs involved may help in deciding which metric to use to measure SLI (Fig. 2):

Server-side Logging which means processing server-side logs of requests or processed data to generate SLI metrics.
Application Server Metrics which is exporting SLI metrics from the code that is serving requests from users or processing their data.

Front-end Infrastructure Metrics which involves utilizing metrics from load-balancing infrastructure to measure SLIs.

Synthetic Clients or Data: Building a client that sends fabricated requests at regular intervals and validates the responses. For data processing pipelines, creating synthetic known-good input data and validating outputs.

Client Instrumentation: Adding observability features to the client the user is interacting with and logging events back to your serving infrastructure that track SLIs. The pros and cons of each metrics is shown in Table 1.

SLI DEVELOPMENT STAGES

A good service level indicator (SLI) is important for building service level objective (SLO). The purpose of SLIs is to provide measurements for the level of reliability promised for the system (Fitzpatrick, 2020). SLI should be kept simple and easy to understand, and only those metrics should be chosen that give business value (Ribenzaft, 2020). It is highly desirable to draft all the service-level metrics such as SLI, SLO and SLA systematically, as this would help in gaining customer trust and improve system reliability and overall performance (Fig 3). SLIs should be formed by identifying the key performance indicators that measure the product performance. After that service-level measures such as SLOs and SLA should be created in order to achieve the achieve better reliability and understand the product's capabilities (Chacko and Merlyn, 2021). Thus, SLI serves as the foundational blocks that help in building SLOs which in turn helps with overall reliability mentioned in the SLAs.

How to choose Good SLIs? Picking the good SLIs is very important phase of SLI development plan. As the complexity of the cloud infrastructure grows, it is recommended to organize the system components into three service types namely, i) Response/Request ii) Storage and iii) Pipeline (Fig 3). Response/Request service type runs on Availability, Latency and Throughput whereas storage depends on Availability, Latency and Durability. Pipeline uses correctness and Freshness as the key SLI types (Azer and Xin, 2020).

SLI Development Stages: There are four stages of developing the SLI and SLOs for a user journey which includes, i) SLI specification, ii) SLI Implementation, iii) Coverage gaps and iv) SLO Targets (USENIX SLO Workshop). The *first step* is to identify the type of SLI that needs to be measured for user journey and establishing high level specifications for these SLIs. This could be possibly the simplistic mapping of user interaction to recommended SLI types. The *second step* is the detailed description of the events to be measured, validity restrictions to the SLI scopes and what makes an event so good. They should be refined in so much detail that anyone could build or configure monitoring infrastructure without asking any further questions. The *third step* is to identify the coverage gaps which means critically examining the interactions of infrastructure with the customer. Any failure modes should be noted carefully and should be resolved before more to next stage. The *final step* is setting up SLO Targets which is usually done if satisfactory SLI implementations are observed. The target value of a service level is measured by SLI. According to Google (Chacko and Merlyn, 2021): SLI ⩽ target ⩽ SLI ⩽ upper bound.

III. Comparative Study

Review Case Study- I

SRE (Site Reliability Engineering) is attracting attention as an approach that attempts to solve many issues related to service and infrastructure reliability with the power of software. Now that IT systems have greatly impacted businesses, efforts to ensure and enhance the "reliability" of various systems are becoming more important (Beyer et al., 2016).

What are the really important indicators for services to achieve stable service operations? It is necessary to think carefully from the user's perspective and understand how to measure and evaluate the index. Therefore, by defining the SLI / SLO / SLA introduced in above sections, one can quantitatively grasp that the service is "normal" and have a common understanding of "normal." As discussed above, SLI (Service Level Indicator) is an index for measuring service quality (Beyer et al., 2016). It is a service level index, a quantitative index that defines the value measured from the user's service. Specifically, SLI includes request latency, error rate, system throughput, availability, and durability (as shown in Fig. 2). As discussed in SLI development stages, an SLO (Service Level Objective) is a set of values or a target value for a service level that an SLI determines. The structure for an SLO is, therefore, lower bound ⩽ SLI ⩽ upper bound, or SLI ⩽ target. For instance, when there is a decision to reverse Shakespeare search "fast," it could be assumed that the SLO average search request latency could be above 100 milliseconds (Beyer et al., 2016). On the other hand, when the average latency requests are below 100 milliseconds, setting such priorities could encourage scripts with low latency behaviors or different types. However, according to Coulter (2020), selecting and maintaining a suitable SLO is challenging. An example has been shown in Fig. 4. People make the system. Naturally, people are creatures that make mistakes. Even if it is not found in the development stage, it is not uncommon to find defects combined with other programs. Since it is basically done in combination with a program developed by another person, bugs will naturally appear. This means that the system is not the end of making. It is meaningless if there are many problems with the product that developers have made. *Black-box testing, and white-box testing* are always performed in the testing process. The difference between the two tests is for users or developers (Beyer et al., 2016). The white box is for developers because the main thing is that various processes are done properly. A black-box test is used to check whether the user's requirements, such as ease of use, are met. White boxes and black boxes have very different roles. But that does not mean SRE developers will do just one or the other. Basically, both are done. The rule of thumb is to do both in most cases, regardless of system type. Even if SRE can meet the user's specifications by performing only the black box test, it is meaningless if bugs and other defects frequently occur (Beyer et al., 2016). Also, even if SRE only performs white-box testing and the system has few bugs, it is not good for users to find it inconvenient. SRE team must evaluate from the perspective of both the user and the developer. The team can adopt the malware detection approaches as discussed by Galal et al., 2016. In recent years, targeted attacks targeting specific companies and individuals have increased, and attacks that bypass previously effective signature-based detection are increasing (Galal et al., 2016). As a result, the detection rate has declined, and it is becoming difficult to protect corporate information assets from malware threats. Security vendors who have traditionally provided signature-based solutions are also aware of the limitations and are increasingly implementing new solutions. However, the basic concept of security measures has changed significantly due to the sophistication and complexity of attack methods compared to the past (Souri and Hosseini, 2018). Previously, the central idea was how to protect companies' information assets, such as confidential information. However, with the advent of targeted attacks, it is now difficult to completely shut out malware, so behavior-based threat countermeasures that assume intrusion are of great importance. Table 2 shows the key inferences of this case study. In summary, an SLO is fundamental to SLI equally as an SLO is essential to an SLI. This is because an SLI is a metric used to ensure compliance with an SLO and connotes the client's level of service. Therefore, if one is developing a new system, both the SLOs and SLIs should be the integral parts of the system's needs.

Special Case Study-II (Real-Life Example)

SLAs are used in order to address the question of which web service is right for a given consumer. SLAs define the mutual agreement between a service and that service’s provider, in order to guarantee the level of transaction that must be provided. Failing to meet SLAs could result in financial consequences for service providers, and therefore, they must understand what they can promise and what their infrastructure is capable of delivering. In order to create an approach/simulation to decide a given SLA, modeling service level indicators (SLIs) is necessary to generate expectations as to a service's performance level. For the HP case study (Jin et al., 2002), they used various service level indicators in order to inform their optimal Service Level Agreements with users. The HP case study includes two model abstractions - both a business process and a web service (shown in Fig. 5). The business process defines the sequence of activities, and can be made of decision points, joining points, and loops. For their web service, the business process captures the integration logic from one provider to another. Because most enterprises tend to outsource those functionalities that others can provide effectively; it is important to determine which providers to use for the outsourcing. In order to pick a web service to create new business functions, service level indicators help to distinguish the differences in service quality. For a customer to choose these web services, the SLI's are compared across multiple providers to decide the optimal service. In order to discern the possible service level premises that web service vendors can provide, combinations of SLI values are checked by a simulation and synthesized into experimental data. For example, the ratio of qualified purchases and quality of service could define "what if" scenarios that then describe the service levels of purchases. From here, altering scenarios from transactions were demonstrated. Provider managers can then understand SLI combinations, as well as their impact on the business process. Through designing simulations within this process, provider managers can then put their SLO in proper SLI ranges, therefore providing the most optimal yield per response time and service quality. For example, when the qualified purchase rate in this case is moved up from 70% to 80%, the duration of transactions also improves about 8%. Meanwhile, as the purchase rate reaches 98%, the transaction duration would decrease by 20%. Through this process, one can decipher how reduced business process duration would raise transactions, while also reducing customer response time and completion time. This can help businesses such as HP balance the cost of asking for higher qualified purchase rates, and the benefit of having that rate. In order to have a web service be discoverable by other web services, they must be registered. The registration consists of two phases made first of static attributes and then of dynamic attributes - static including things such as URL, service type, and protocols, while dynamic consists of reliability, SLA violation rate, and cost. Dynamic attributes focus more on real-time needs of consumers, and often customers will give out their priority of SLIs. With these available SLIs, customers can then evaluate the possibilities of combinations of SLIs and see charts which showcase the impact of business processes and distribution. From here, they can then select those which are most needed for the function of their business.

SLI metrics used by HP: The main SLI this case study is looking at is an indicator that can examine the service level of a "purchase unit". These purchase units are measured in time and take into account a few different metrics such as duration, quality of service, and if the unit is a qualified purchase or not. To properly measure this SLI HP created what if scenarios to see what kind of outputs they would get. They generated 1000 transactions with random intervals and created a what if scenario that shows business managers what may need to be updated depending on certain situations. If one process during a purchase is taking an extended period of time this will need to be updated to help throughput. If we look at the section written above titled "metrics to measure SLI" we can see a few categories that could classify this SLI type HP is describing. This would fall into the Synthetic clients or data SLI metric. HP is building fabricated requests at regular intervals in order to create synthetic data that allows managers to have known good input data and validate outputs. Although this is a good way to gain knowledge there are some downsides to doing this. One downside is that this approach approximates user experience with synthetic requests rather than actually gaining a real user experience. Although tests are run in the thousands this system cannot perfectly imitate the user experience. Another major downside to this metric is that high reliability data requires frequent probing and testing for accurate measures. This constant probe is not only time consuming, but probe traffic can drown real time user traffic and even slow down their experience. Table 2 shows the key inferences of this case study.

REFERENCES

[1] Azer, M., & Xin Tai, K. (2020, June 22). Cloud Monitoring as a Service | Datadog. Service Level Objectives 101: Establishing Effective SLOs | Datadog. Retrieved from http://www.datadoghq.com/blog/establishing-service-level-objectives/

[2] Betsy, B. (2017). https://sre.google/sre-book/service-level-objectives/. O'Reilly Media, Inc.

[3] Beyer, B., Jones, C., Petoff, J., & Murphy, N. R. (2016). Site reliability engineering: How Google runs production systems. " O'Reilly Media, Inc.".

[4] Chacko, B., & Shelley, M. (2021, January 20). Incident Management Software for SRE and Devops - Squadcast. The Key Differences between SLI, SLO, and SLA in SRE | Squadcast. Retrieved from http://www.squadcast.com/blog/the-key-differences-between-sli-slo-and-sla-in-sre

[5] Coulter, M. (2020). Avoiding Goodhart's Law-Use SLO's as Tools Not Cudgels. In SREcon20 Americas (SREcon20 Americas).

[6] Fitzpatrick, S. (2020, May). Align IT Efforts with Business Outcomes | BizOps. What to Consider When Determining SLOs and SLIs. Retrieved from http://www.bizops.com/blog/what-to-consider-when-determining-slos-slis

[7] Galal, H. S., Mahdy, Y. B., & Atiea, M. A. (2016). Behavior-based features model for malware detection. Journal of Computer Virology and Hacking Techniques, 12(2), 59-67.

[8] Jin, L. J., Machiraju, V., & Sahai, A. (2002). Analysis on service level agreement of web services. HP June, 19, 1-13.

[9] RIBENZAFT, R. (2020, November 02). Instantly Understand Your Microservices | Epsagon. *A Complete Guide to SLAs, SLOs, and SLIs | Epsagon.* Retrieved from http://epsagon.com/development/slas-slos-slis-guide/

[10] Souri, A., & Hosseini, R. (2018). A state-of-the-art survey of malware detection approaches using data mining techniques. Human-centric Computing and Information Sciences, 8(1), 1-22.

[11] USENIX | The Advanced Computing Systems Association. Retrieved from http://www.usenix.org/sites/default/files/conference/protected-files/srecon18emea_slides_fong-jones.pdf

[12] Watts, S. (2020, May 12). BMC Software – Run and Reinvent. A Primer on Service Level Indicator (SLI) Metrics – BMC Blogs. Retrieved from http://www.bmc.com/blogs/service-level-indicator-metrics/

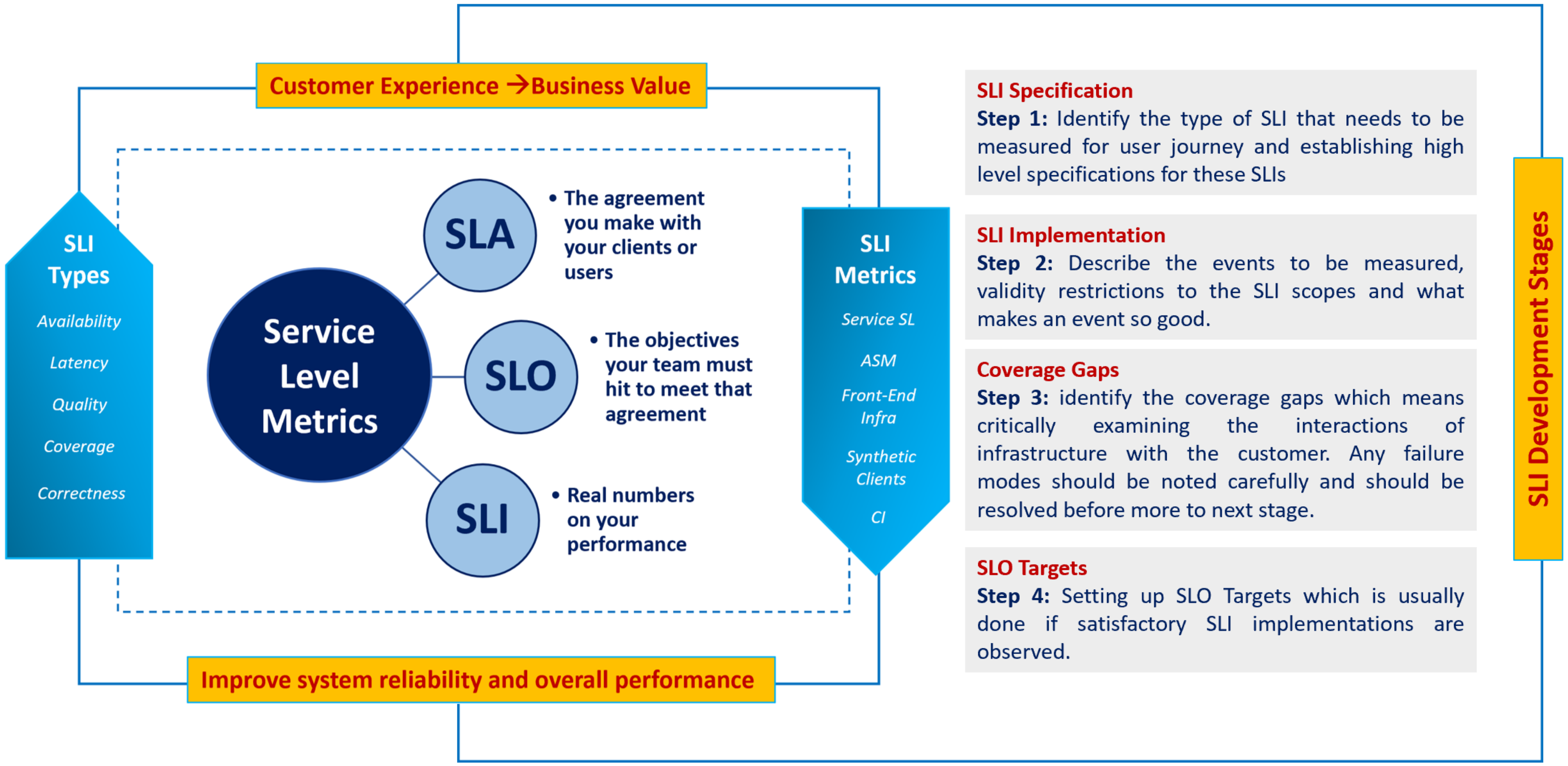

Customer Experience →Business Value
SLI Types
Availability
Latency
Quality
Coverage
Correctness
Service Level Metrics
SLA
• The agreement you make with your clients or users
SLO
• The objectives your team must hit to meet that agreement
SLI
• Real numbers on your performance
SLI Metrics
Service SL
ASM
Front-End Infra
Synthetic Clients
CI
Improve system reliability and overall performance
SLI Specification
Step 1: Identify the type of SLI that needs to be measured for user journey and establishing high level specifications for these SLIs
SLI Implementation
Step 2: Describe the events to be measured, validity restrictions to the SLI scopes and what makes an event so good.
Coverage Gaps
Step 3: identify the coverage gaps which means critically examining the interactions of infrastructure with the customer. Any failure modes should be noted carefully and should be resolved before more to next stage.
SLO Targets
Step 4: Setting up SLO Targets which is usually done if satisfactory SLI implementations are observed.
SLI Development Stages

Fig. 1. Project Workflow describing essential elements of Service Level Metrics - Indicators

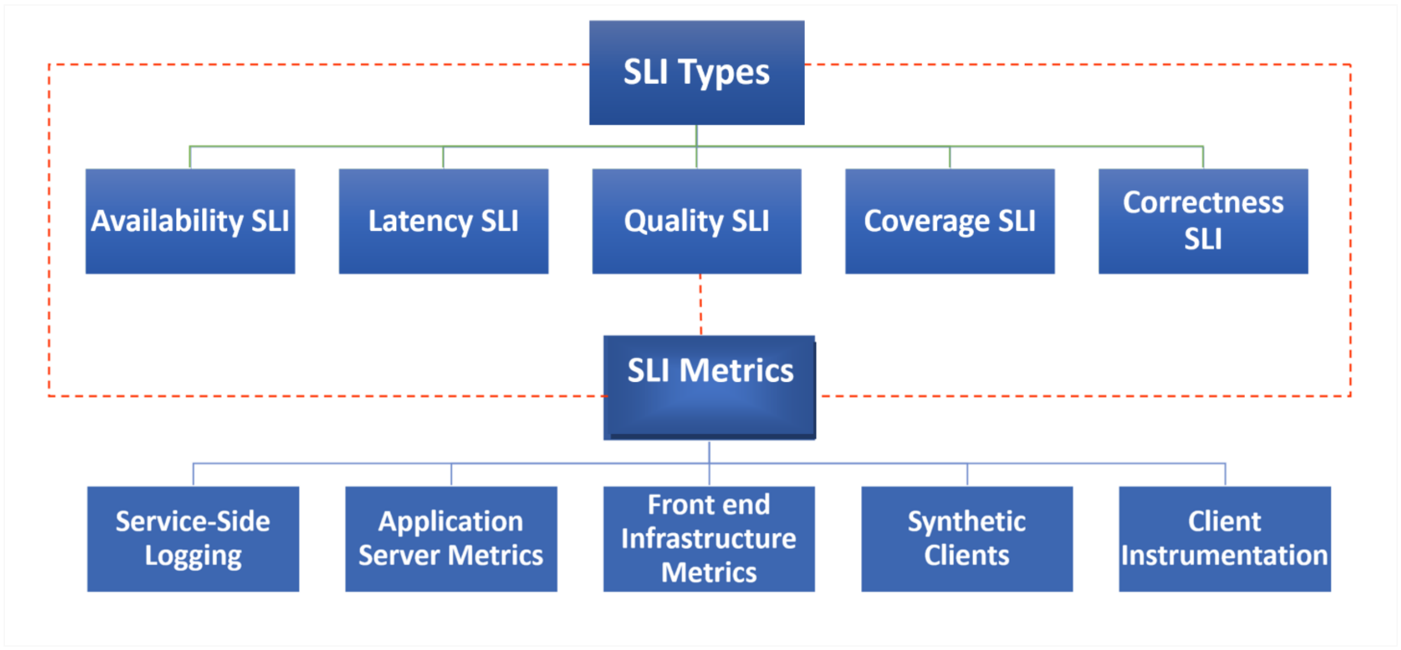


Fig. 2. SLI Metrics and Types Taxonomy that may be considered before building or developing the SLI for user journey

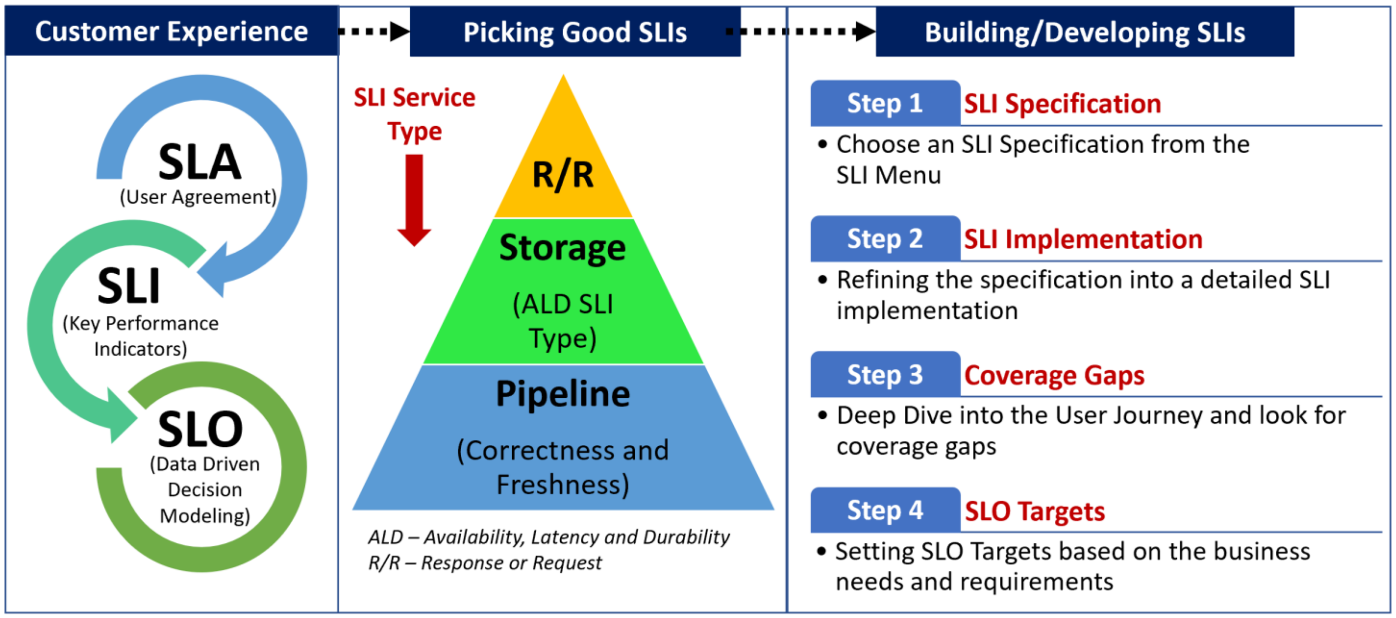
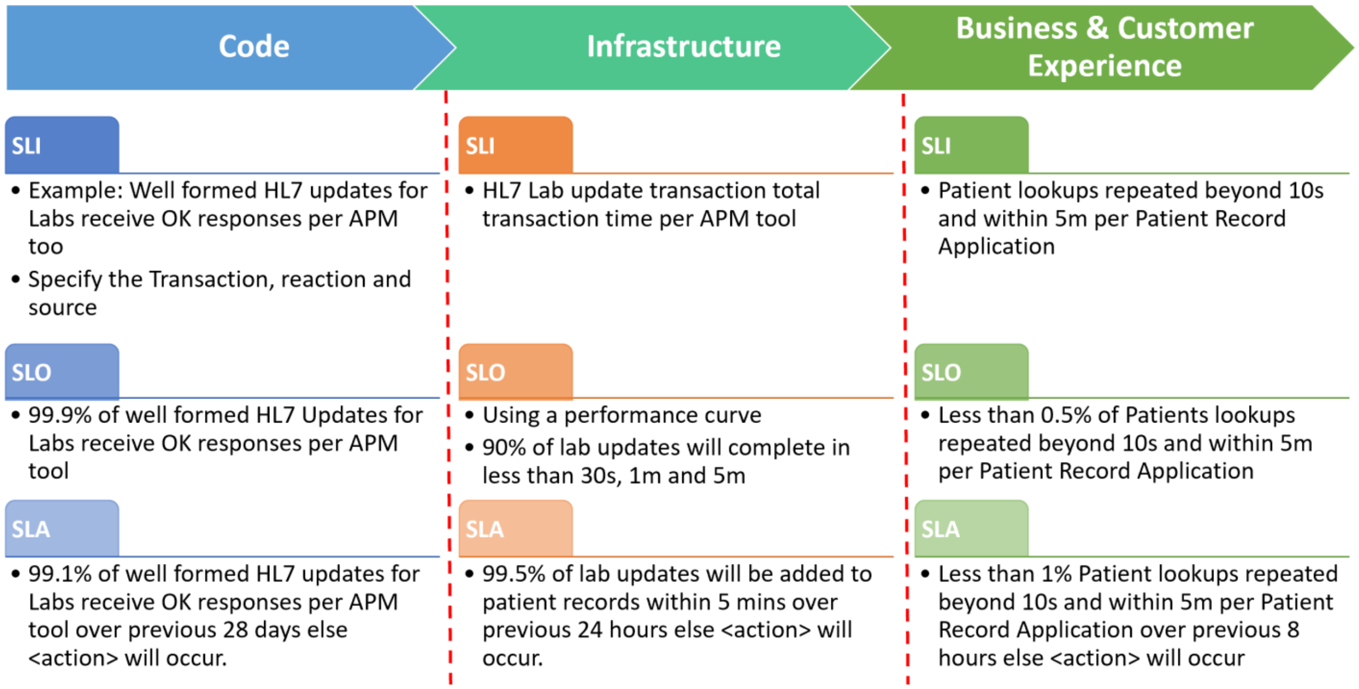

Customer Experience
Picking Good SLIs
Building/Developing SLIs
SLA
(User Agreement)
SLI
(Key Performance Indicators)
SLO
(Data Driven Decision Modeling)
SLI Service Type
R/R
Storage
(ALD SLI Type)
Pipeline
(Correctness and Freshness)
ALD – Availability, Latency and Durability
R/R – Response or Request
Step 1 SLI Specification
• Choose an SLI Specification from the SLI Menu
Step 2 SLI Implementation
• Refining the specification into a detailed SLI implementation
Step 3 Coverage Gaps
• Deep Dive into the User Journey and look for coverage gaps
Step 4 SLO Targets
• Setting SLO Targets based on the business needs and requirements

Fig. 3. Development plan for Service Level Indicators (SLIs)



Code

SLI
- Example: Well formed HL7 updates for Labs receive OK responses per APM too
- Specify the Transaction, reaction and source

SLO
- 99.9% of well formed HL7 Updates for Labs receive OK responses per APM tool

SLA
- 99.1% of well formed HL7 updates for Labs receive OK responses per APM tool over previous 28 days else <action> will occur.

Infrastructure

SLI
- HL7 Lab update transaction total transaction time per APM tool

SLO
- Using a performance curve
- 90% of lab updates will complete in less than 30s, 1m and 5m

SLA
- 99.5% of lab updates will be added to patient records within 5 mins over previous 24 hours else <action> will occur.

Business & Customer Experience

SLI
- Patient lookups repeated beyond 10s and within 5m per Patient Record Application

SLO
- Less than 0.5% of Patients lookups repeated beyond 10s and within 5m per Patient Record Application

SLA
- Less than 1% Patient lookups repeated beyond 10s and within 5m per Patient Record Application over previous 8 hours else <action> will occur

Fig. 4. Example case study that shows the use of three service performance metrics in three different dimensions i.e. Code, Infrastructure and Business and Customer Experience

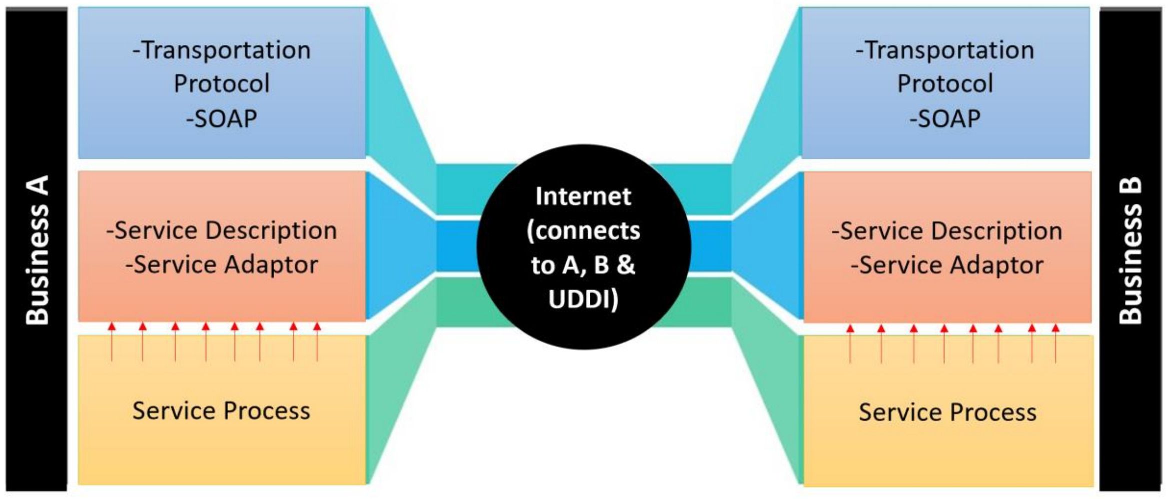

Business A
-Transportation Protocol -SOAP
-Service Description -Service Adaptor
Service Process
Internet (connects to A, B & UDDI)
-Transportation Protocol -SOAP
-Service Description -Service Adaptor
Service Process
Business B

Fig. 5. Model Abstractions of HP Case Study: Web Service and Business Process

Table 1: SLI Metrics (Definition, Advantages and Disadvantages)

Metrics	Definition	Advantages	Disadvantages
Server Side Logging	Processing server-side logs of requests or processed data to generate SLI metrics.	Existing request logs can be processed retroactively to backfill SLI metrics. Complex user journeys can be reconstructed using session identifiers. Complex logic to derive an SLI implementation can be turned into code and exported as two much simpler "good events" and "total events" counters	Application logs will not contain requests that did not reach servers. Processing latency makes logs-based SLIs unsuitable for triggering an operational response. Engineering effort is needed to generate SLIs from logs; session reconstruction can be time-consuming.
Application Server Metrics	Exporting SLI metrics from the code that is serving requests from users or processing their data.	Often fast and cheap (in terms of engineering time) to add new metrics. Complex logic to derive an SLI implementation can be turned into code and exported as two much simpler "good events" and "total events" counters.	Application servers are unable to see requests that do not reach them. Measuring overall performance of multi-request user journeys can be difficult if application servers are stateless.
Front end Infrastructure metrics	Utilizing metrics from load-balancing infrastructure (e.g., GCP's layer 7 load balancer) to measure SLIs.	Metrics and recent historical data most likely already exist, so this option probably requires the least engineering effort to get started. Measures SLIs at the point closest to the user still within serving infrastructure.	Not viable for data processing SLIs, or in fact any SLIs with complex requirements. Can only measure approximate performance of multi-request user journeys.
Synthetic Clients or data	Building a client that sends fabricated requests at regular intervals and validates the responses. For data processing pipelines, creating synthetic known-good input data and validating outputs.	Synthetic clients can measure all steps of a multi-request user journey. Sending requests from outside your infrastructure captures more of the overall request path in the SLI.	Approximates user experience with synthetic requests. Covering all corner cases is hard and can devolve into integration testing. High reliability targets require frequent probing for accurate measurement. Probe traffic can drown out real traffic.
Client Instruments	Adding observability features to the client the user is interacting with and logging events back to your serving infrastructure that track SLIs.	Provides the most accurate measure of user experience. Can quantify reliability of third parties, e.g. CDN or payments providers.	Client logs ingestion and processing latency make these SLIs unsuitable for triggering an operational response. SLI measurements will contain a number of factors outside of direct control. Building instrumentation into the client can involve lots of engineering work.

Table 2: Comparison of Case Study 1 and Case Study II

Parameters	**Case Study – I (Review Type)**	**Case Study-II (Example Type)**
Objective	An exhaustive study on how the basic concepts of security has changed from the past is done. Additionally, several methods for threat protection have been carefully reviewed.	Identify which web service is right for a given customer? Also gauge what modeling service level indicators would be best to generate expectations as to a service's performance level.
Service Performance Metrics	SLI, SLO and SLA at all three dimensions: Code, Infrastructure and Business and Customer Experience Level	Main SLI for this case study is an indicator that can examine the service level of a Purchase Unit (They are measured in time and consider a few other metrics such as duration, quality of service, and if the unit is qualified purchase or not. To measure this SLI, Synthetic clients or data SLI metric is used.
Testing Methods or Models	Testing Methods: Black box and White Box Testing methods (Beyer et al., 2016) Signature based detection methods (Galal et al., 2016) Behavior-based threat countermeasures	Two model abstractions are used – business process and web service are used (Jin et al., 2002)