Serverless Cloud Function

Customer Cases

Service Notice

This document provides an overview of the as-is details of Tencent Cloud's products and services in their entirety or part. The descriptions of certain products and services may be subject to adjustments from time to time.

The commercial contract concluded by you and Tencent Cloud will provide the specific types of Tencent Cloud products and services you purchase and the service standards. Unless otherwise agreed upon by both parties, Tencent Cloud does not make any explicit or implied commitments or warranties regarding the content of this document.

Contact Us

We are committed to providing personalized pre-sales consultation and technical after-sale support. Don't hesitate to contact us at 4009100100 or 95716 for any inquiries or concerns.

Contents

Customer Cases
Ecommerce Industry

Last updated: 2023-12-05 17:10:09

Customer Overview

The customer is a leading ecommerce company with the vision to integrate shopping and community to provide more effective suggestions for more consumers for better shopping decision making.

Customer challenges

There are millions of online users who discuss about fashion and shopping topics every day on the customer's platform. These behaviors generate massive amounts of data. After these data sources generate data, a component needs to get the data from the data sources and write it to Kafka. Previously, the R&D team generally processed the data with open-source data storage solutions such as Logstash and Filebeat or developed internal code to implement this logic.
With the continuous growth of the customer's business, the data volume became larger and larger. In order to ensure the availability, reliability, and performance, high numbers of R&D resources were required. Therefore, the customer urgently needed new solutions.

SCF + CKafka Solution

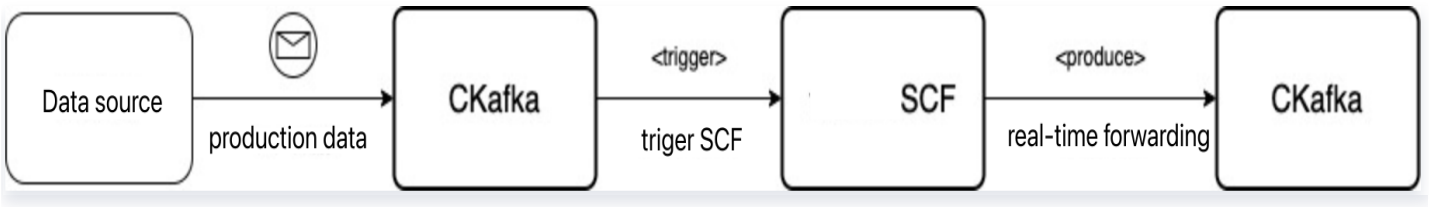
The customer's technical team compared many technical solutions on the market and chose Tencent Cloud SCF for its following natural strengths in terms of learning costs, scalability, manual maintenance costs, and stability:

- Support for multiple languages.
- Low learning costs, with no need to learn open-source solutions and distributed scheduling.
- Unlimited elastic scalability.
- Multiple trigger methods, such as event trigger, timer trigger, and active trigger.
- Few maintenance costs for cluster stability and availability.
- Pay-as-You-Go billing mode accurate down to 1 ms and favorable prices.

> **Description**
> For more information on CKafka, please see Message Queue CKafka.

The solution of SCF + Ckafka provides proprietary console UIs where traffic alarming and scaling can be configured securely and reliably.
The business solution provided by the Tencent Cloud SCF team for the customer is to dump messages of a topic on one instance to the corresponding topic on another instance through an SCF function. Compared with the original Connector solution, each SCF function can be easily managed in the Tencent Cloud console to control the trigger threshold, trigger status, etc. Simply put:

- **Message dump**: is to sync messages in topics to offline clusters.
- **Cluster migration**: plays a dual-write role in the process of cluster migration and merge.



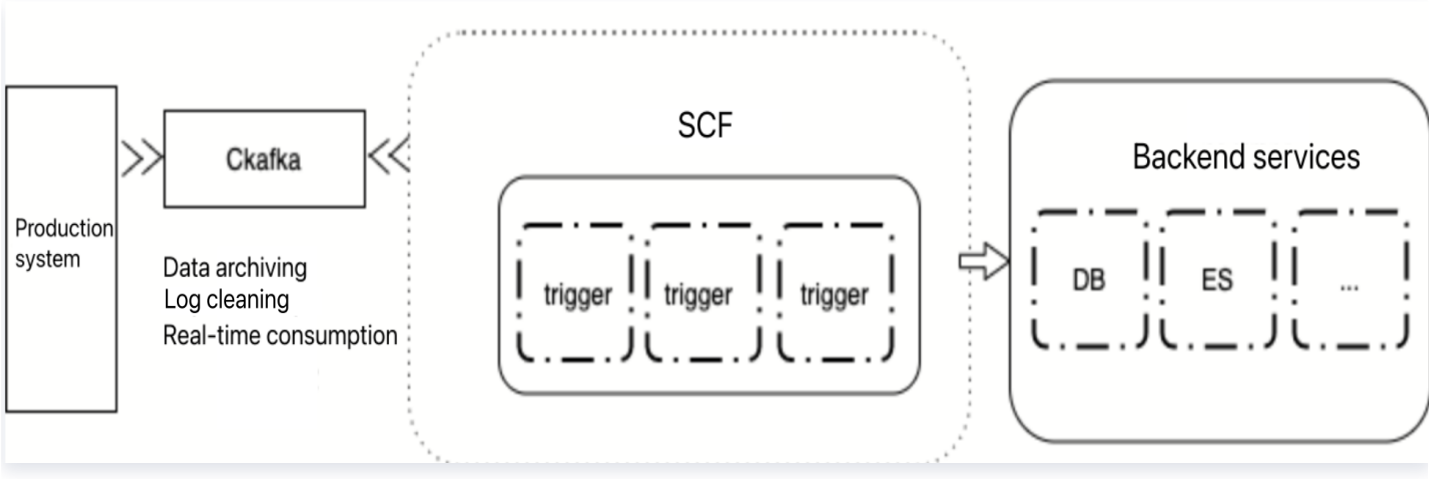
After comparison, the customer determined that the solution of SCF + CKafka was the best choice and chose it for the message sync business.

Solution Advantages

With the prosperity of the Kafka community, more and more ecommerce users have started to use Kafka for activities such as log collection, big data analysis, and streaming data processing. CKafka works together with SCF to offer many practical features with the following optimizations:

- It features distributed architecture, high scalability, and high throughput based on Apache Kafka.
- It is 100% compatible with Apache Kafka APIs (0.9 and 0.10).
- It offers all features of Kafka with no need for deployment.
- It encapsulates all cluster details, eliminating the need for OPS on your side.
- It supports dynamic upgrade and downgrade of instance configuration in a pay-as-you-go manner (under development).
- Its message engine is optimized to deliver a performance 50% higher than that of open-source Kafka.

As shown below, SCF can consume messages in CKafka in real time for tasks such as data dumping, log cleaning, and real-time consumption. Moreover, functions like data dumping are integrated into the CKafka console, allowing users to enable them with a single click, significantly reducing the complexity of use.



Compared to a CVM-based self-created CKafka consumer, SCF helps eliminate a lot of unnecessary overheads:

- The CKafka trigger can be quickly enabled in the SCF console to automatically create a consumer, and SCF will maintain the high availability of components.
- The CKafka trigger itself supports many practical configurations, such as the offset position, 1-10,000 messages to be aggregated, and 1-10,000 retry attempts.
- Business logic developed based on SCF naturally supports auto scaling, eliminating the need to build and maintain server clusters.

Online Video Industry Audio/Video Transcoding Best Practices

Last updated: 2023-12-05 17:10:34

Customer Overview

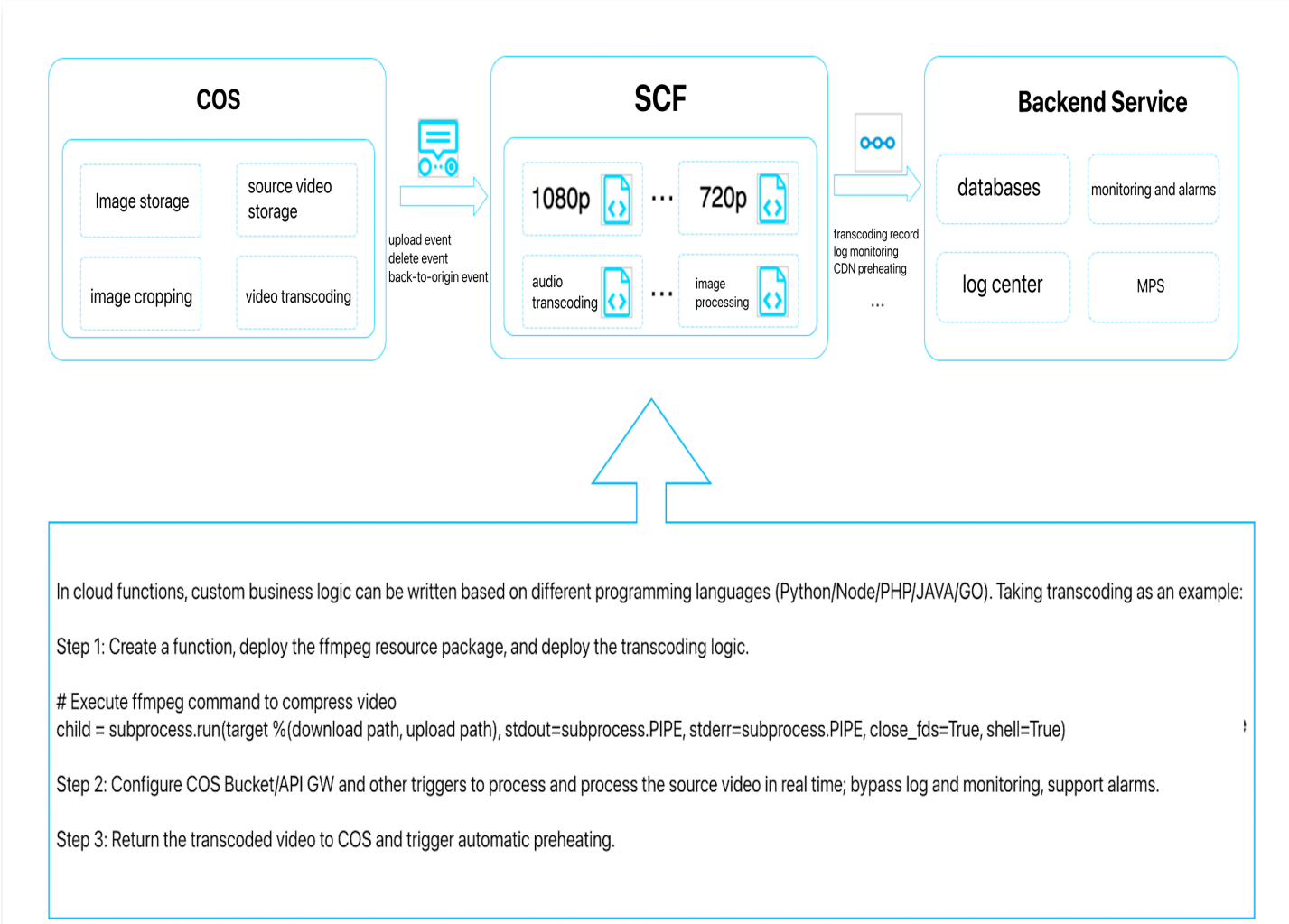
The customer is a comprehensive online education and training group headquartered in Zhongguancun, Beijing, China and listed on the New York Stock Exchange in 2006. Its businesses include foreign language training, K12 education, preschool education, online education, consultancy for studying abroad, and book publishing. It also has several education sub-brands.

Customer challenges

Every summer, a large number of students study on the customer's platforms. Previously, the storage and transcoding logic for audio/video courseware were implemented in a self-built IDC based on servers and NFS. However, due to the high volume of traffic during the summer, the servers in the IDC might not be able to meet the computing requirements, and it would take very long to purchase new hardware devices for self-built services, so the customer wanted to find a flexible method to support rapid business deployment and complete transcoding efficiently.
In use cases such as video and social networking, users upload high numbers of images and audio/video files frequently, which has high requirements for the real-timeness and concurrency capabilities of the processing system. Traditional container services require the customer to maintain the container clusters on their own and have a poor elastic scalability.

SCF Solution

SCF supports custom transcoding functions to help enterprises quickly build customized task processing capabilities. This makes up for the blind pots of current separate cloud services and conveniently migrates the FFmpeg service from physical servers, cloud servers, or containers to SCF.
The operating principle of using SCF + FFmpeg and COS for audio/video transcoding is as follows:



In terms of technical solution, SCF and COS are used in combination in the cloud to achieve elastic scalability and sustain all local traffic switched to the cloud. In the new business process, a task scheduling module is added to automatically or manually forward the received business traffic to the proprietary cloud-based services. In addition, many high-availability technologies are added to the process, including full-linkage tracking by task `TraceID` and automatic local retry upon cloud computing failure. In the new solution, cloud-based services are easier to develop and maintain and more cost-effective. The pay-as-you-go SCF billing mode also reduces resource costs greatly.

Serverless Application Benefits

Strengths of using SCF to implement the audio/video transcoding service:

- SCF provides a standard runtime environment, ensures the high availability and auto scalability of resources, and requires no dedicated maintenance.

- SCF is billed by actual usage, eliminating the waste of resources.
- Development and debugging in SCF are more efficient, and dependencies are decoupled from the business and can be hot updated separately in real time.
- Runtime environments are isolated, so the failure of one single request will not affect the normal execution of other requests.

You need to pay attention to the following to connect your existing business to SCF:

- The introduction of SCF needs to be integrated with the existing CI/CD process, so there may be some changes in the development method.
- You need to make certain modifications on your existing business code to integrate FFmpeg to the function code and deploy it together with the code file to SCF.

Online Efficiency Tool Industry Examples.

Last updated：2023-12-05 17:19:42

Customer Overview

This enterprise offers a multifunctional note-taking application, which is one of the most influential tools in the field of knowledge management. It serves not only as an intelligent assistant for personal information management but also as a corporate tool for enhancing team efficiency. Furthermore, it is a content platform that amasses high-value information and an exceptional producer of intelligent hardware related to knowledge and information. The primary clientele of this enterprise comprises highly educated, high-income knowledge workers, a vast number of professionals, and students, with the number of service users exceeding one hundred million. The product adopts the industry-leading Freemium business model, maintaining a high level of user activity, paid conversion, and renewal rates. Simultaneously, the enterprise actively expands user scenarios, independently developing several intelligent hardware devices to effectively support various input scenarios such as handwriting, scanning, and voice.

Customer challenges

Since the customer completed the capital restructuring in 2018, the product features have been frequently iterated, and the business has grown rapidly. The technical team often needs to address some features or projects that need to be launched quickly. The traditional service or microservice development delivery model can no longer meet the project schedule requirements. Therefore, they began to seek solutions.
The enterprise analyzed its own business characteristics and found that the new features it launched were relatively independent, with clear business logic and low coupling with other modules. At the same time, the concurrent processing volume is related to actual user behavior, and it is impossible to accurately predict resource usage in the early stages.

SCF Solution

After adopting Tencent Cloud's Serverless technology, there was a significant improvement in the development experience:

- The development speed has significantly accelerated. Both Serverless and SCF provide a wealth of preset engineering templates, and are closely integrated with other Tencent Cloud services such as CMQ and COS.
- Deployment is convenient, eliminating the need for complex task orchestration and

resource configuration steps in the early stages.

- Once the business is launched, it is easy to maintain, and operations personnel no longer need to consider pressure and scaling issues.
- With comprehensive logging and statistical features, you can instantly and conveniently grasp the operational status of the function service.

Currently, enterprises are gradually trying to use frameworks like Tencent Cloud's Serverless for rapid development and delivery in some internal business data processing and user asynchronous notification functions.
In the future, enterprises will further expand their attempts to use Tencent Cloud's Serverless technology, mainly from the following directions:

- Mini Program server-side functionality.
- SEO/SSR Related.
- User asynchronous interactions, such as regular reminders, account status-related notifications, etc.

Finally, some advice for teams still considering the use of Serverless technology:
If the product team is considering iterations for the product or developing new features, they might consider using Tencent Cloud's Serverless technology. Serverless has a low degree of coupling with other modules and there's no need to worry about user usage. Serverless has the ability to scale elastically without limit, with virtually no maintenance cost for cluster stability and availability.
Additionally, if the team needs to validate some new projects initially, requiring data extraction, data analysis, they might consider using Serverless ETL for data extraction (Extract), data transformation (Transform), and data loading (Load). The advantage of Serverless in this regard is its flexibility. It does not affect the data processing flow of existing projects, can run independently to meet data validation requirements, and has the advantage of lower learning and cost expenses.

Online Video Industry

Last updated: 2023-12-05 17:20:33

Customer Overview

A certain domestic online video media platform, with audio-visual interaction at its core, integrates the characteristics of the internet and television to achieve a "multi-screen integration". It is a new media audio-visual comprehensive communication service platform that offers exclusive broadcasts, cross-screen, and self-produced content. It is also one of the earliest state-owned controlling video platforms in the domestic A-share market.

Business Requirements

Requirement 1. Audio/Video encoding/decoding business

The audio/video team is responsible mainly for the following businesses:

- Audio/Video output of self-made content:
  - Daily production of high volumes of source videos
  - Transcoding and quick adaption and release of videos with more than 50 codecs and different definitions from 240p to 4K
- Transcoding efficiency/algorithm exploration:
  - Compression algorithm for videos with ultra-high bitrate
  - Digital remastering, video super resolution (VSR), and frame interpolation

For playback of self-made audio/video content, high numbers of videos are generated every day, which consume a huge quantity of computing resources. Videos with different definitions from 240p to 4K and more than 50 codecs such as AVC, HEVC, and MPEG are processed daily.
However, in terms of UGC, creator content needs to be delivered to users quickly to make profits for creators.

Requirement 2. Research on region of interest (ROI) video encoding

Based on the visual redundancy principle and in-depth optimization of the encoder engine, this technology can reduce the bitrate by over 40% compared with the native x264 and x265 codecs while maintaining the same subjective image quality.

- **Content-aware adaptive encoding**
  By conducting preliminary preprocessing and analyzing the video's dynamic complexity, scenes, and shots, the system can adaptively match the best cost-effective bitrate on the RDO encoding curve. This allows for a reduction of over 30% in bitrate without compromising the subjective quality of the viewing experience.

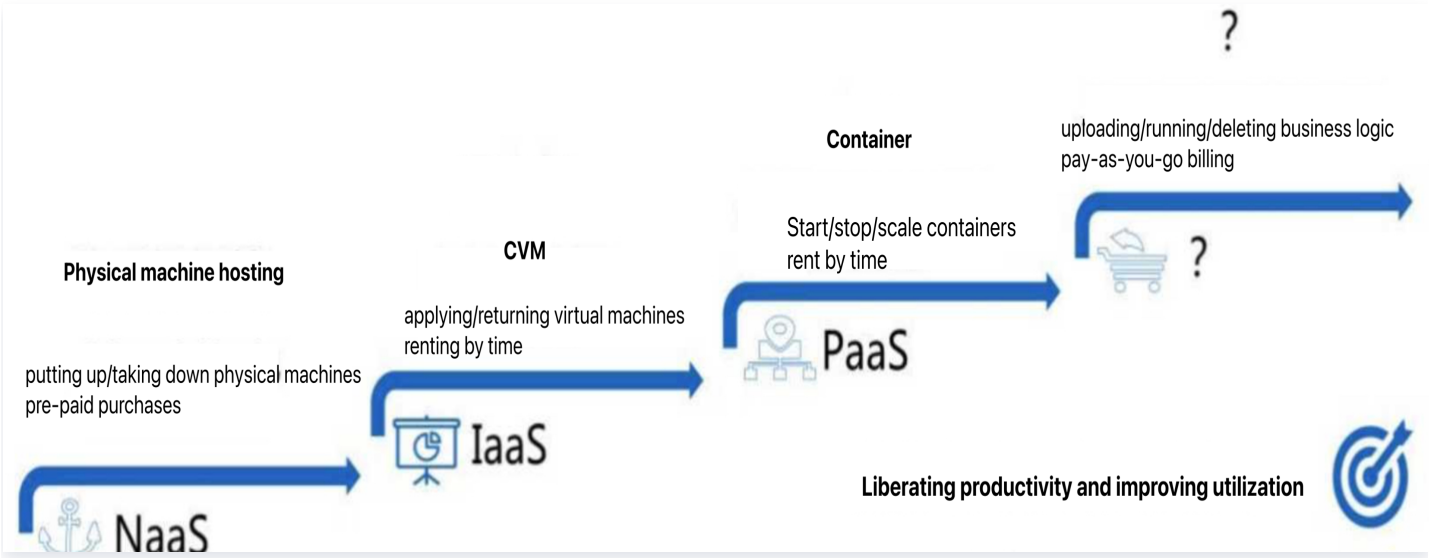
- **Region of Interest Encoding**
  As shown in the image below, the human eye is focused on the car, where one can see a person driving. By leveraging the characteristics of the human visual system's region of interest, an AI-based model can predict the region of interest and guide the encoder to allocate encoding quality weights in different areas. This allows for a reduction of over 15% in bitrate while maintaining the same subjective experience.
- **Video Encoding Image Enhancement Technology**
  Our proprietary series of encoding image optimization technologies can enhance the viewing experience beyond the original video quality without increasing the bitrate.

Requirement 3. Audio/Video transcoding platform product iteration

- First generation: it was released in 2015 and implemented based on Hadoop MapReduce. The size of processed data was relatively small. As the business grew rapidly, subsequent extension of the architecture would be difficult.
- Second generation: the Mesos-based distributed resource management framework was used as the business scale and daily size of processed data increased. Particularly, after the exclusive program strategy was proposed in 2016, audio/video content needed to be quickly produced and released. To meet such needs, segmented video transcoding was implemented.
- Third generation: in 2019, the AI and Tencent Cloud image optimization technologies were introduced, and Kubernetes was used to implement resource scheduling, proprietary scheduling, and workflow orchestration. During the COVID-19 pandemic, the number of videos surged, especially UGC videos, which were 10 times more than before. After SCF was leveraged, cluster scaling could be implemented quickly to provide the required computing resources.

Necessity of SCF

As can be seen from the following figure, the development of cloud computing from left to right shows that cloud computing technology has been continuously improving. From the early physical machine hosting to the emergence of cloud hosts and containers, and now to the advent of Serverless, the development has been very rapid.



The characteristics of early physical machines, cloud servers, and containers determined that the service had low load for 30% of their operation time, and early video transcoding was generally performed in a local IDC containing over 100 servers, which were seriously insufficient in the daytime but had low load at night. Therefore, the server resources could not be used reasonably. In contrast, SCF can ensure reasonable resource usage.

Benefits of SCF

- SCF supports fast deployment and flexible on-demand usage based on auto scaling, reducing the bottlenecks of use.
- SCF implements multi-region task scheduling according to the specific use cases. It schedules resources in each region based on user source to ensure reasonable resource usage.
- SCF ensures uninterrupted business operations through a cloud-based backup and disaster recovery mechanism.
- SCF reduces the resource and labor costs while maintaining a high performance.

SCF Implementation

Audio/Video transcoding

SCF can be used to transcode audio/video in the cloud in three simply steps:

1. Create a function and deploy the proprietary encoder resource package and transcoding logic.
2. Configure a COS bucket trigger to process the source video in real time and generate logs, monitoring data, and alarms in a relayed manner.

3. Return the transcoded video to COS and distribute it to the self-built CDN or Tencent Cloud CDN nodes.

The core strength of SCF is that it can organically integrate video upload , processing, extraction, and storage based on its powerful linkage capabilities.

Flexible processing

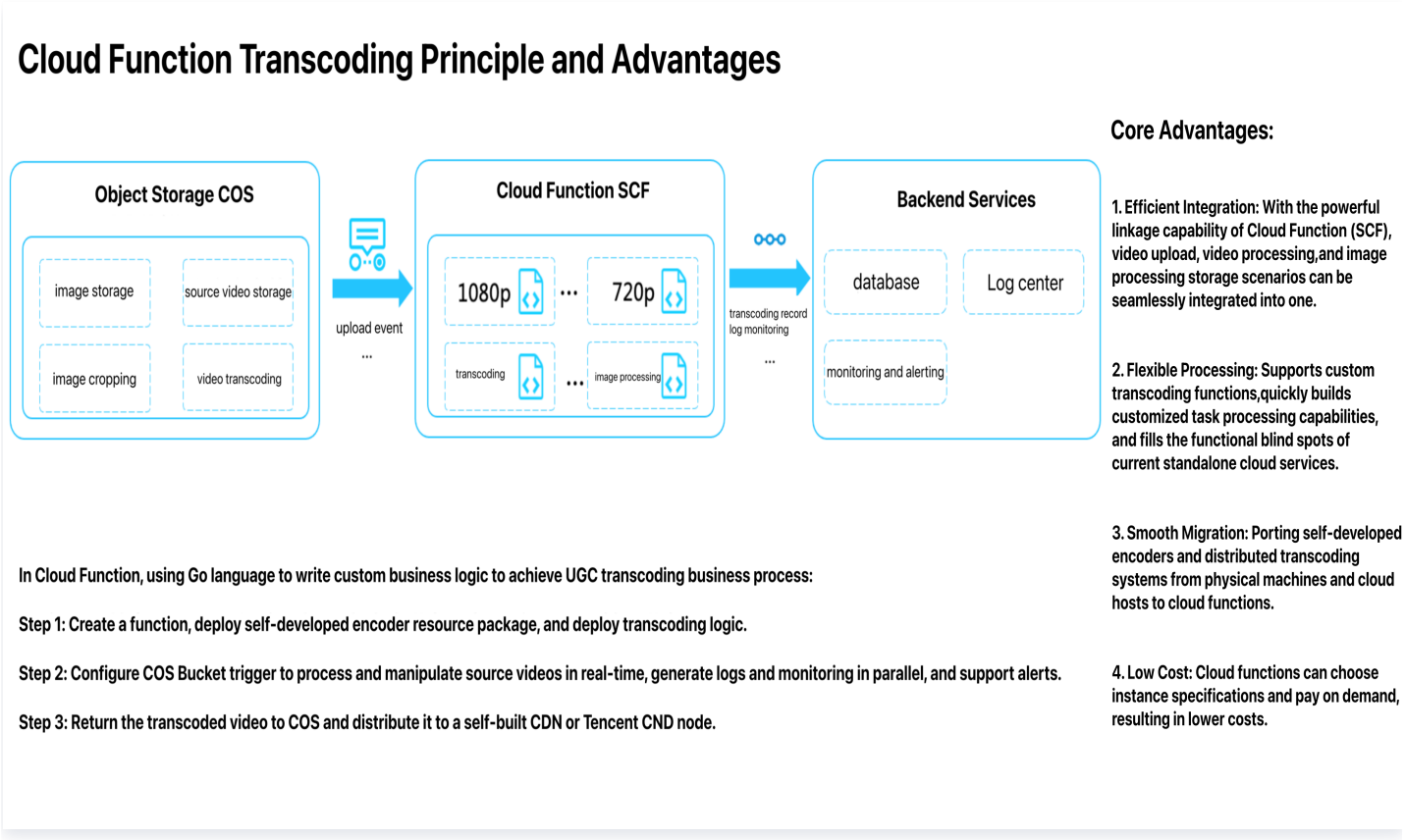
SCF supports custom transcoding functions, and its proprietary encoder can be deployed quickly, eliminating the blind spots of separate services and cloud services.

Smooth migration of transcoding system

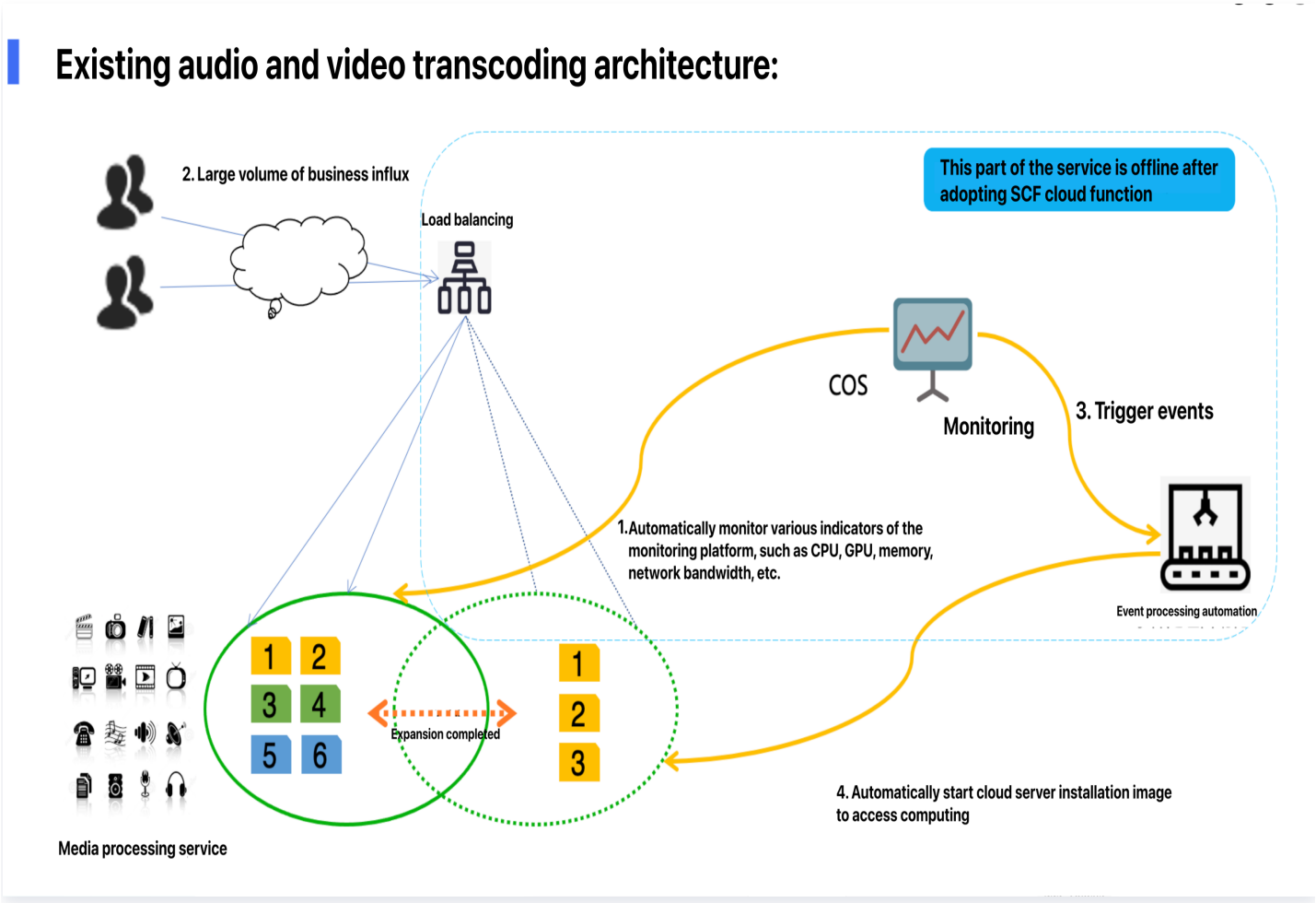
With the VOD service of the cloud service provider of the customer used for the online UPGC content, the encoder was uncontrollable. In contrast, after SCF is used, the transcoding system can be smoothly migrated to SCF, where the required parameters can be adjusted as needed to optimize the video quality.

Reduced costs and improved efficiency

One of the greatest strengths of SCF is significant cost reduction. With SCF, the computing resource **costs are reduced by over 45%.**

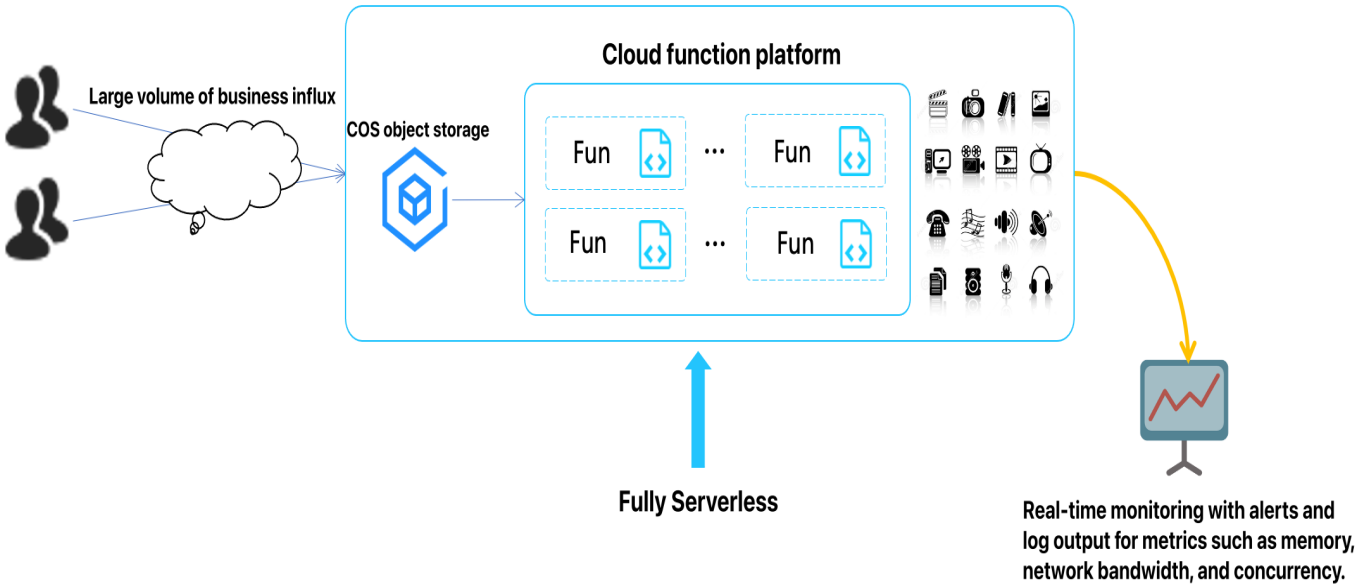


In the original audio and video transcoding architecture, it was necessary to monitor various indicators such as CPU, memory, and network bandwidth. Triggering the installation image of the cloud server to access the computing platform also resulted in a high latency issue. As shown in the figure below:



Previously, resources could not be dynamically reclaimed after startup, leading to a significant waste of computing resources. With SCF, it is possible to handle a large number of user requests through automatic scaling, and with Tencent Cloud's real-time monitoring feature, one can observe the usage of memory, concurrency, and network bandwidth in real time. As shown in the figure below:

Architecture upgrade: Automatic elastic scaling to respond flexibly



After migrating the business to SCF, the customer enjoys the following benefits:

- Ease of use: they only need to implement the business code logic without caring about issues beyond feature development or perform OPS.
- Stability: SCF provides highly stable service in different methods such as cross-region disaster recovery.
- Fast iteration: the versions and API traffic can be assigned as needed to quickly implement grayscale release of solutions.
- Fast start: the service can be started in 1 second and completed in 20 seconds, meeting the business requirements.

Best Practice Of Tencent IEG Going Global

Last updated：2023-12-05 17:24:19

Customer Overview

Tencent Interactive Entertainment Group (IEG), a subsidiary of Tencent Games, primarily focuses on game development, operations, and online gaming communities. In the journey of game cloudification, Tencent Interactive Entertainment has been continuously exploring and breaking through. In March 2021, Tencent Interactive Entertainment launched the online game development platform PGOS for international business, which provides:

- **Backend service platform:** PGOS is an online game solution designed to facilitate game backend development and maintenance and reduce costs, so that developers can focus on the development of gameplay and core logic.
- **Fully managed service:** its complete backend solution eliminates the challenges in setting up, managing, and running servers at scale. Its automatically scalable dedicated servers provide a low latency and high reliability for real-time gaming.
- **One-Stop control panel:** developers and OPS personnel can retrieve player information, view logs, monitor real-time data, and edit service configurations on the its web portal.
- **Cross-platform SDK:** it provides out-of-the-box SDKs for C++ and UE4, making it easy for developers to use the PGOS service on game clients and internal servers.
- **Flexible matching rules:** it enables players to accurately and quickly match up with other players in real-time battles.
- **Scalability and flexibility:** its entire system uses a microservice architecture rather than integrated hierarchical architecture, allowing developers to add and modify interactions between corresponding services.

SCF Solution

Tencent Cloud Serverless naturally supports the above features provided by PGOS and is used by it to provide underlying computing support, better helping teams migrate to the cloud quickly.

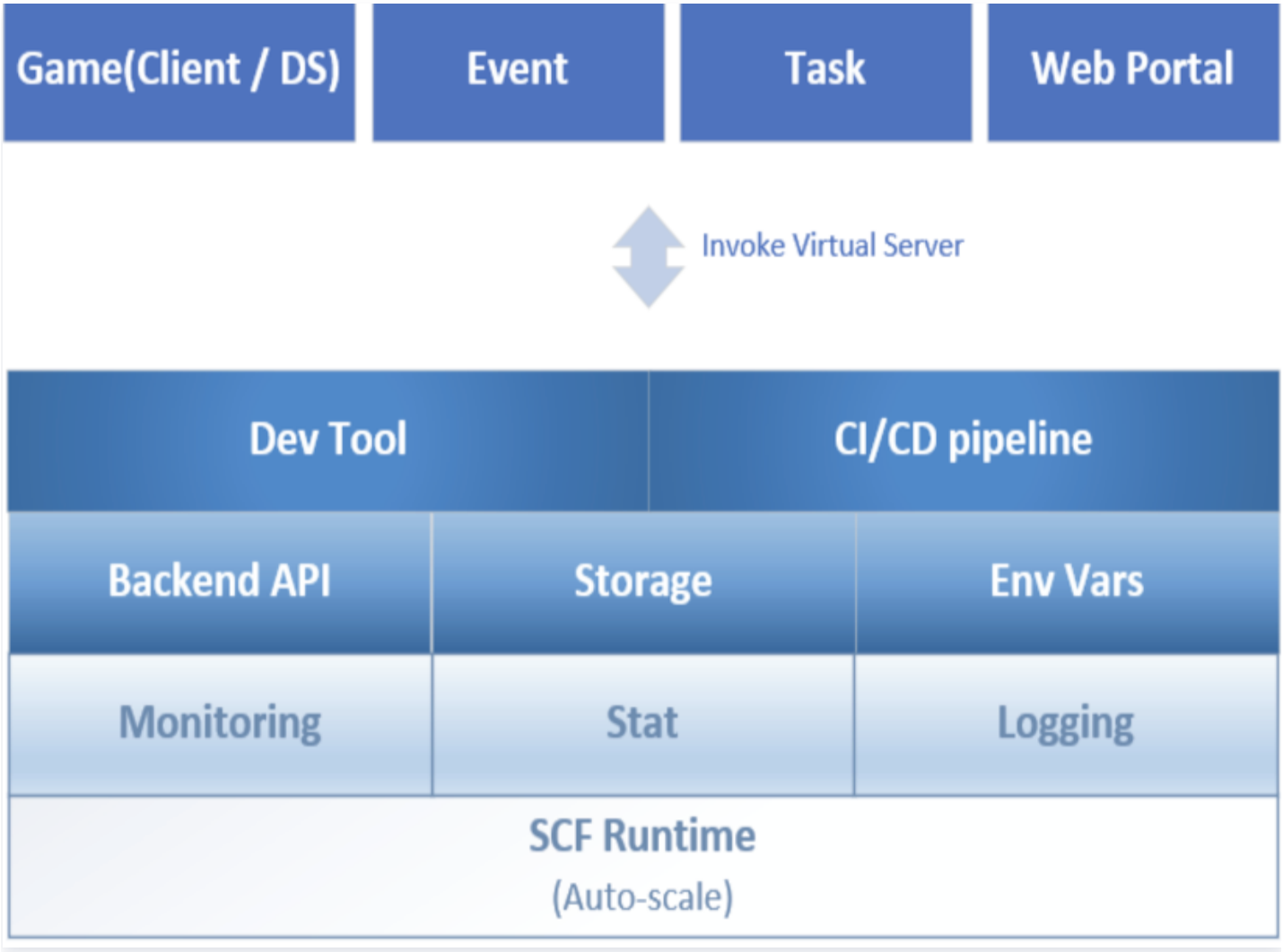
- **Out-of-the-Box serivce:** Tencent Cloud Serverless enables you to focus entirely on business code with no need to purchase, set up, and configure servers. Its architecture not only helps accelerate game publishing and iteration but also significantly reduces the OPS costs. In addition, it guarantees the stability and security of your business and the availability of the resources, eliminating your concerns over underlying resources.
- **Dynamic scaling:** another benefit of serverless is auto scaling, which makes it easier for

your business to withstand traffic surges. Auto scaling guarantees normal business operations when access traffic surges and reduces costs during off-peak hours.

- **Real-time monitoring:** Tencent Cloud Serverless provides real-time logs and monitoring dashboards. Developers and administrators can monitor the operational status of the business in real time. It also integrates with the Tencent Cloud Observability Platform, providing multi-dimensional alarm capabilities for runtime and abnormal status, ensuring that issues can be captured and notified to users in the shortest possible time.
- **Scalability and flexiblity:** the atomic feature of FaaS naturally supports flexible business expansion. Different SCF functions can support independent features, such as mutual function invocation, separate update and deployment, and online function code editing. Tencent Cloud Serverless offers a one-stop solution from business development, deployment, to monitoring.
- **Multiple event triggering methods:** Tencent Cloud Serverless supports around 10 event triggering methods, including timer trigger, API Gateway trigger, and COS trigger, meeting your diverse needs in different scenarios.

How serverless supports cloud gaming with computing power

Tencent Cloud Serverless can provide underlying computing support for the global business of PGOS. You can create one virtual server (corresponding to one or more SCF functions) and write relevant business logic. PGOS relies on it to offer complete monitoring and logging capabilities and connect to backend services. It further encapsulates DevOps tools and furnish fully managed and automatically built and deployed features.



PGOS provides multiple drive methods, and the underlying layer triggers business operations on the virtual server with different function triggers.

- **Timer drive**: a scheduled task can be configured for a game on the web portal to trigger a specific API of the virtual server.
- **Event drive**: the virtual server can listen on specific events and will be automatically triggered when events occurs.
- **Game drive**: The Game Client or DS can actively call the Extension Interface, triggering a specific API of the Virtual Server through the Gateway.
- **Manual drive:** you can manually run/trigger the specific APIs of the virtual server in the web portal.

Full Managed Services

Eliminate the challenges of building, managing and running servers on a large scale with a complete backend solution and DS management service enable server scheduling.

Cross Platform SDK

We have built a C++ SDK and UE4 Plugin that can be used "out of the box" for developers to easily utilize the services in their game clients and dedicated servers.

One-Stop Control Panel

Developers can retrieve player profiles, monitor live data and edit service configurations on PGOS web portal.

Extensions & Flexibility

The entire system is packed with various services, and a microservices architecture pattern is adopted instead of using a monolithic structure with different layers.

WeSure Insurance

Last updated：2023-12-05 17:25:51

This document elaborates on the architectural evolution of WeBao's frontend, as well as the reasons for ultimately choosing Tencent Cloud's Serverless technology to support the frontend architecture.

Customer Overview

WeBao's Million Medical Insurance is an insurance service provided by Tencent's insurance agency platform, WeSure, in collaboration with renowned domestic insurance companies. Unrestricted by the type of illness, it offers reimbursement for treatment costs incurred due to accidents or diseases, with a maximum coverage of up to 6 million.

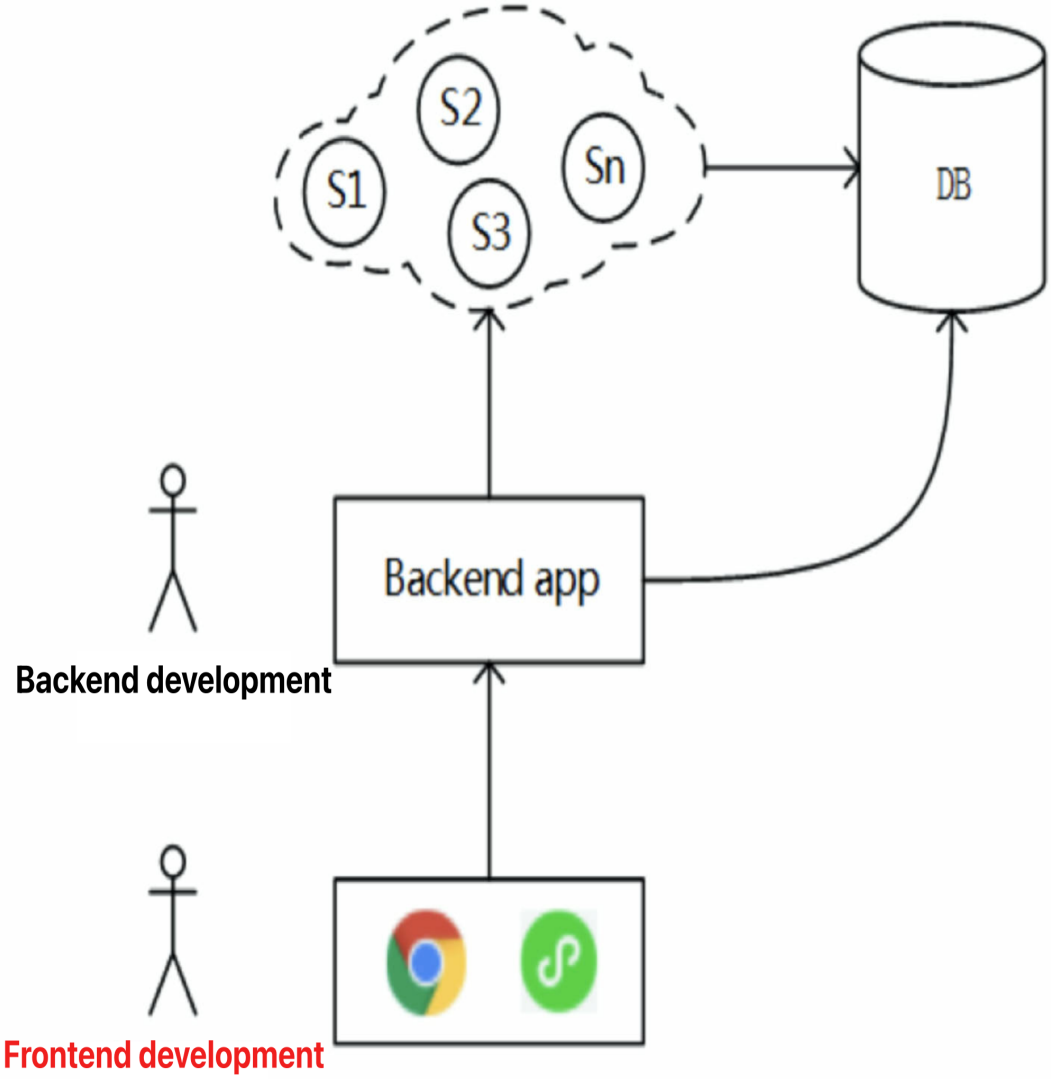
SCF Solution

In the course of business development, WeBao's frontend architecture has continuously evolved and adjusted according to actual business, team, and development conditions. The primary application scenarios for the WeBao team using Serverless technology are as follows:

- Frontend developers apply it at the BFF layer, currently integrating Mini Programs and HTML5 pages.
- Data team members utilize it for risk control and recommendation algorithm applications.

WeBao Architecture v1

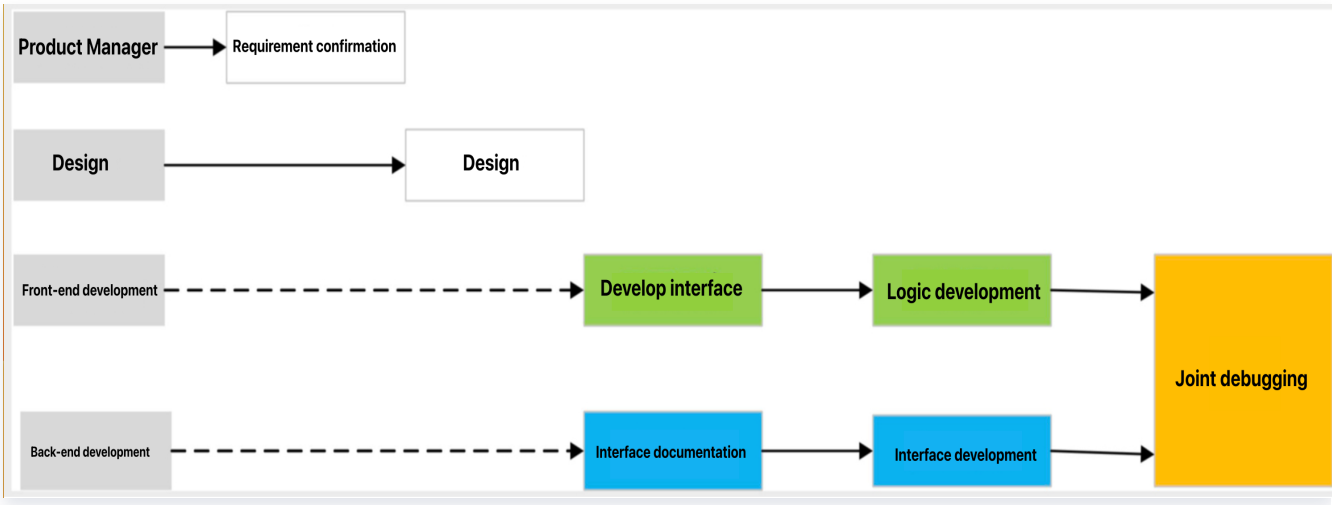
Initially, the WeBao team employed a classic front-end and back-end separation architecture, where front-end and back-end development collaborated through interfaces. The collaboration process is illustrated in the following diagram:



Strengths

The front-end and back-end separation architecture offers the following significant advantages:

- **Decoupling of Front-end and Back-end Development**
  - Front-end and back-end development proceed in parallel, shortening the overall development cycle of requirements



- The roles are clearly defined, making the identification and resolution of online issues more precise. The front-end and back-end independently design and implement their own error monitoring and alert systems. The front-end captures page script errors and reports them to the logging platform. After processing by the logging tool, an alert mechanism is set up to push error information to the relevant developers.
- Facilitates front-end componentization and back-end microservices architecture. After the front-end and back-end are separated, the front-end can use more convenient frameworks and basic UI components based on these frameworks, greatly improving development efficiency. In addition, front-end development will extract business-specific common components based on business characteristics. All componentization precipitations contribute to the enhancement of production efficiency.

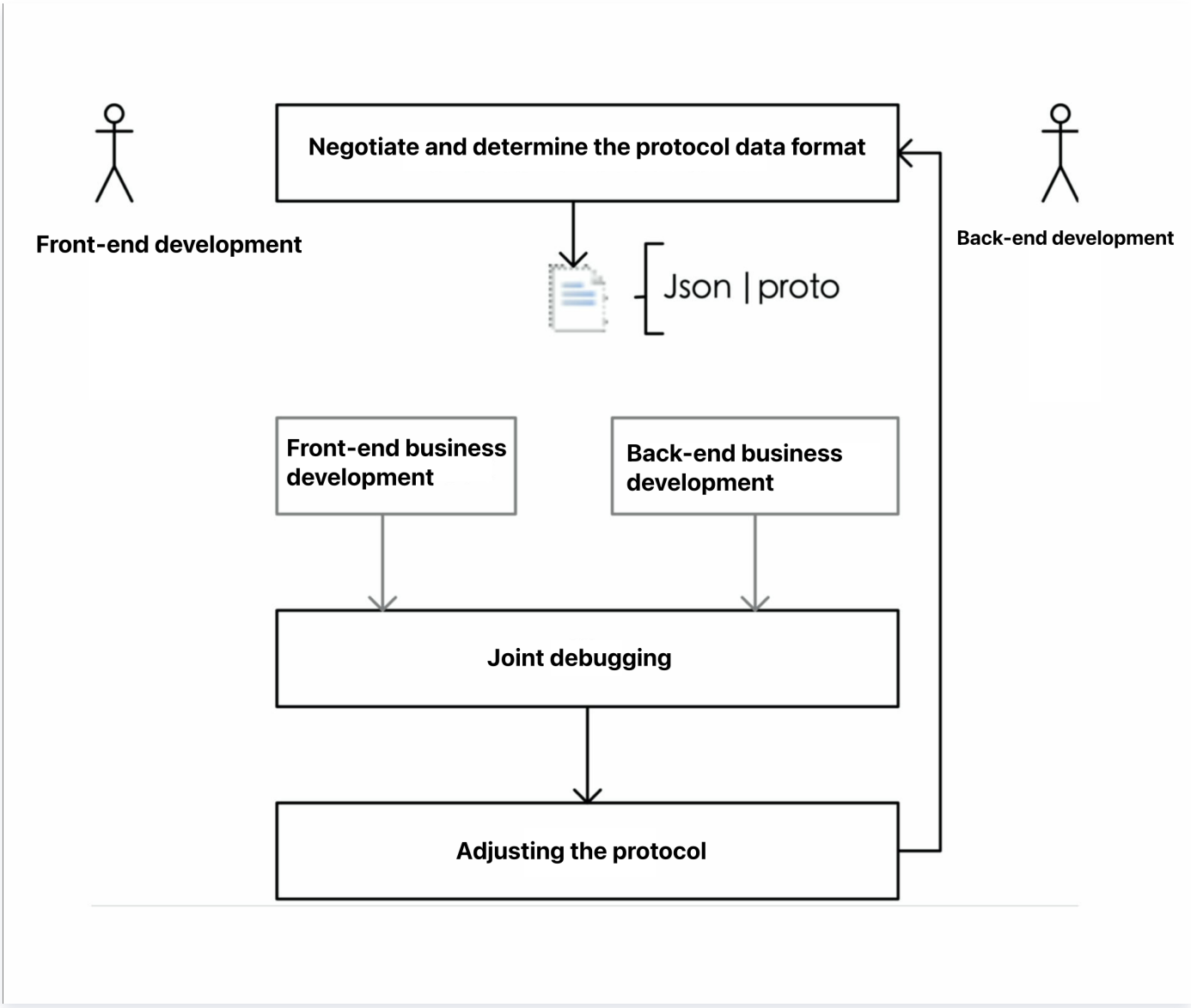
- **Decoupled Deployment**
  - Front-end static files are independently deployed on CDN. The front-end project contains a large number of static files, including HTML, CSS, JS, images, videos, etc. Deploying these files on the CDN, fully utilizing the CDN capabilities of existing cloud services, not only enhances the speed of resource access but also ensures the stability of resource access, especially in high-concurrency scenarios.
  - Faster CI/CD, the front-end compilation process can be easily integrated into CI/CD. The unified deployment of the front-end and back-end is interdependent. After separate deployment, CI/CD can be implemented for front-end projects at the Gitlab repo level.

The front-end and back-end separation architecture is very effective for rapid early-stage business development. Moreover, when the team size is small, front-end and back-end developers work together consistently, and they are familiar with each other's development habits and personalities, so cross-role communication issues are not prominent. However, as

the team and business scale continue to expand, this architectural pattern will gradually bring some side effects to the team. In practice, we encountered several of the following obvious problems:

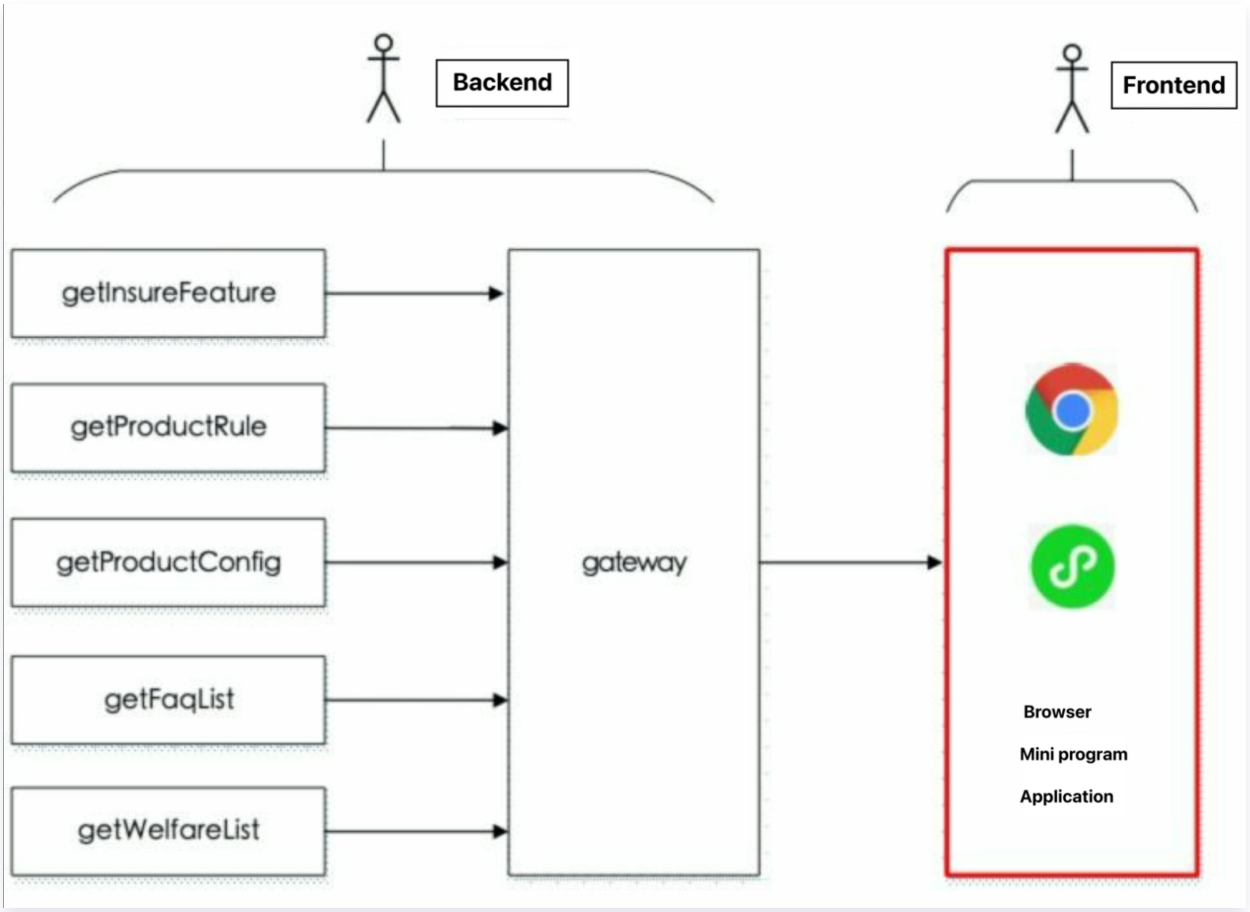
Drawbacks

- **Front-end and back-end collaboration coupling becomes a bottleneck for improving development efficiency**
  The expansion of team size and business scale means more people to collaborate with and an increase in the complexity of interfaces.
  In the early stages, when team members were familiar with each other's development habits and some business aspects, the cost of communication for adjusting business interface parameters was relatively low. However, as the business scale and team members expanded, during various cross-business collaborations, we often encountered colleagues who were not accustomed to reading proto documents, or some complex businesses that required a lot of time to read proto documents, or repeated communication between front-end and back-end on the specific meaning of interface call parameters. As shown in the following figure:



- **Poor page rendering efficiency**
  Taking the product details page as an example, the rendering of the page requires at least five backend interfaces, followed by data assembly and processing. This not only increases the code volume on the mini-program end but also slows down the page rendering speed.
  Even if the front-end page accesses the interface in parallel, the data integration logic will still be very complex. As the main product form of WeChat Insurance, the mini-program has a huge amount of code, almost reaching the code limit set by WeChat. This increase in

code with business growth is also an invisible risk. As shown in the following figure:



WeChat Insurance Architecture v2

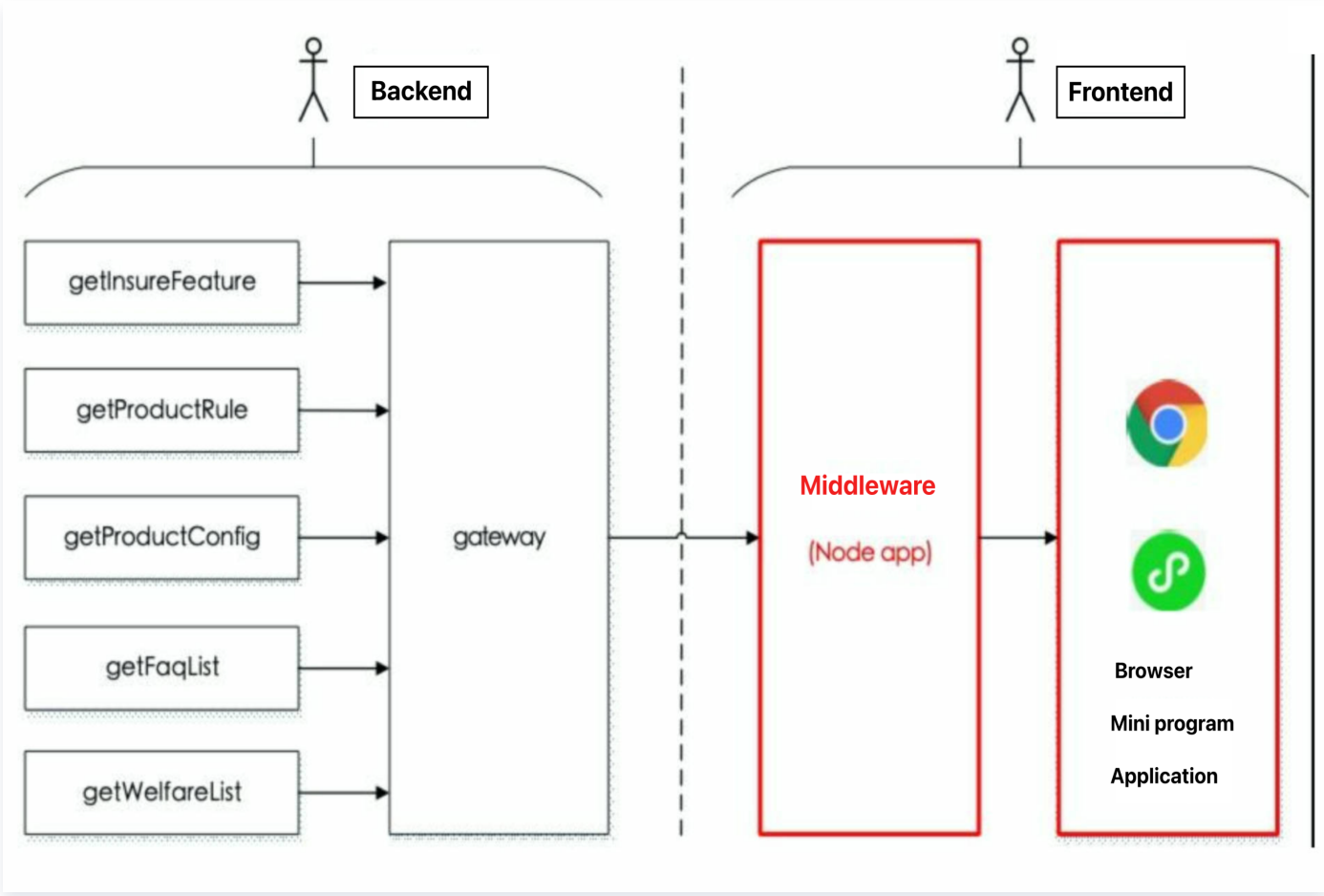
In light of the pain points in the front-end and back-end collaboration model during the WeChat Insurance Architecture v1 phase, the team optimized the architecture again, with the principle of moving the business "forward" and sinking the core. In the early stages of various business supports, the team accumulated some experience in business middle platforms, such as insurance services, renewal services, policy services, etc.
The introduction of the middle layer further enhances the team's development efficiency, and greatly strengthens the front-end's control over the business and the operational space for page performance optimization. Whether it's from the agility of the team or the experience of the application, there are significant improvements, mainly reflected in the following aspects:

Strengths

- **The coupling of front-end and back-end processes is greatly reduced, and the specificity of role responsibilities gradually forms**
  Based on the accumulation of some backend service capabilities, such as policy-related requirements, only front-end developers need to participate in the requirement review and

development process. Front-end developers communicate business logic with business products, query the corresponding service capabilities in the API market or service documents, and complete business development. At the same time, it is very beneficial for the team to gradually carry out business middle-end and front-end componentization, and the entire architecture responds more agilely to diverse business requirements.

- **The rendering layer aggregates backend services, reducing page requests**
  Whether it's an H5 webpage or a mini-program page, they only need to interact with the middle layer for data. Front-end developers can aggregate interfaces according to business demands, and only one request is needed on the client side to start rendering the page. Before the interface aggregation, the display of the product details page required requests to five interfaces, with an average interface request time of about 120ms. After aggregation, the middle layer is used to request five internal network interfaces, avoiding the time consumption of multiple connections between the client and the service.



- **The middle layer processes data, greatly reducing the amount of logic code on the mini-program end**
  Previously, some complex logic in the data integration code on the mini-program end could be handled by the middle layer. The savings of these codes will increasingly show

value as the business continues to grow. Taking the annuity product details page as an example, data aggregation in the middle layer can save about 10KB in volume.

The introduction of the middle layer is a further liberation of productivity, but based on a Node middle layer of a giant App, some problems are also exposed in the later operation and maintenance. The application of the middle layer is deployed on two cloud servers CVM, which has some inherent shortcomings, mainly reflected in the following aspects:

Drawbacks

- **Poor ability to cope with peak traffic shocks**
  Microinsurance often has some operational and delivery requirements, these events can lead to a sudden influx of large traffic, while the expansion speed of the cloud server CVM is relatively lagging.
- **App-level deployment and release**
  As the middle layer continuously accumulates business code, the entire application grows linearly. Each deployment and release is a monolithic application release, resulting in low deployment efficiency and high risk.
- **Increased usage threshold**
  Front-end developers need to check logs and service operations on their machines in the development and testing environment, which raises the threshold for widespread use.

Microinsurance Architecture v3

Based on some limitations of the Microinsurance Architecture v2 phase, Microinsurance began to pay attention to new technologies. The Riselab team of the University of California, Berkeley's computer science laboratory proposed that Serverless is the next wave of cloud computing. Major domestic manufacturers have also begun to layout Serverless. Tencent Cloud Serverless team is one of the earliest teams to deploy in this area domestically, the technology is very mature, and it has been successfully implemented in hundreds of enterprises.

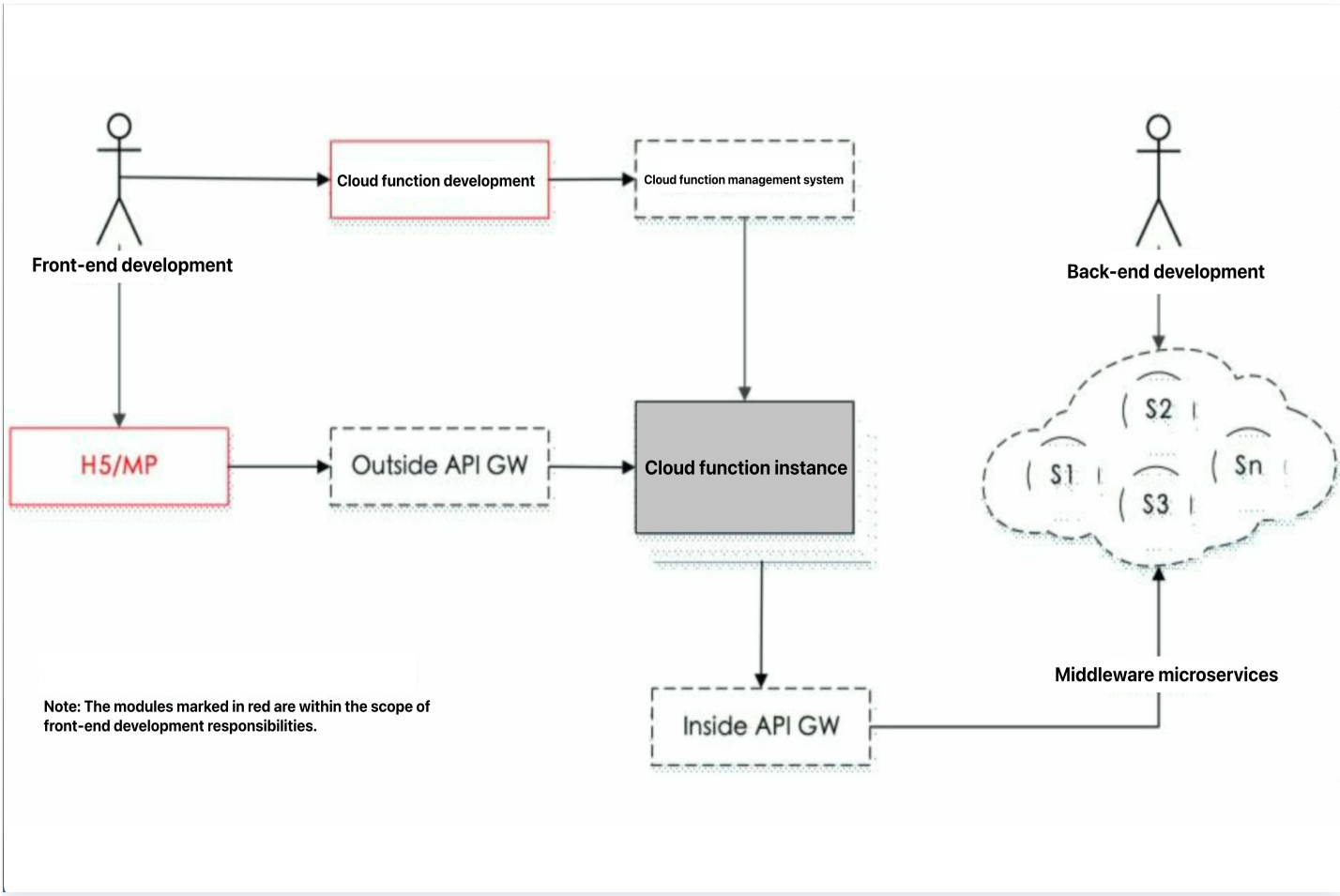
Microinsurance learned about the advantages of Tencent Cloud's Serverless Cloud Functions through research:

- Powerful scaling capabilities, particularly suitable for handling traffic peaks, while maintaining stable performance.
- Each function operates, deploys, and scales independently. Once the user uploads the code, it can be automatically deployed, enhancing the speed of independent development and iteration.
- Cloud Functions provide detailed log records, allowing for convenient viewing of function operation status, and facilitating code debugging, testing, and auditing. It supports the reporting of relevant monitoring metrics, enabling a quick understanding of the overall

operation of the function, and also allows for the customization of Cloud Function monitoring metrics.

- Billing rules are accurate to 1ms, and charges are only applied to functions that are currently running.

In summary, the issues faced in the Microinsurance architecture v2 have been essentially resolved. After a comprehensive team evaluation, Microinsurance decided to use Tencent Cloud's Serverless Cloud Functions for further architectural adjustments. The schematic diagram of the role collaboration process after adjustment is shown in the following figure:



Client-side requests are sent to the Cloud Function API Gateway, which forwards the requests to the corresponding Cloud Function instances. The Cloud Function then requests the service gateway. The most significant change in the entire chain is the replacement of the Node App with Cloud Functions, becoming the technical form of the middle layer.
The decision to replace the Node App with Cloud Functions was based on several considerations, primarily aimed at addressing some of the issues encountered in the practical use of Node App:

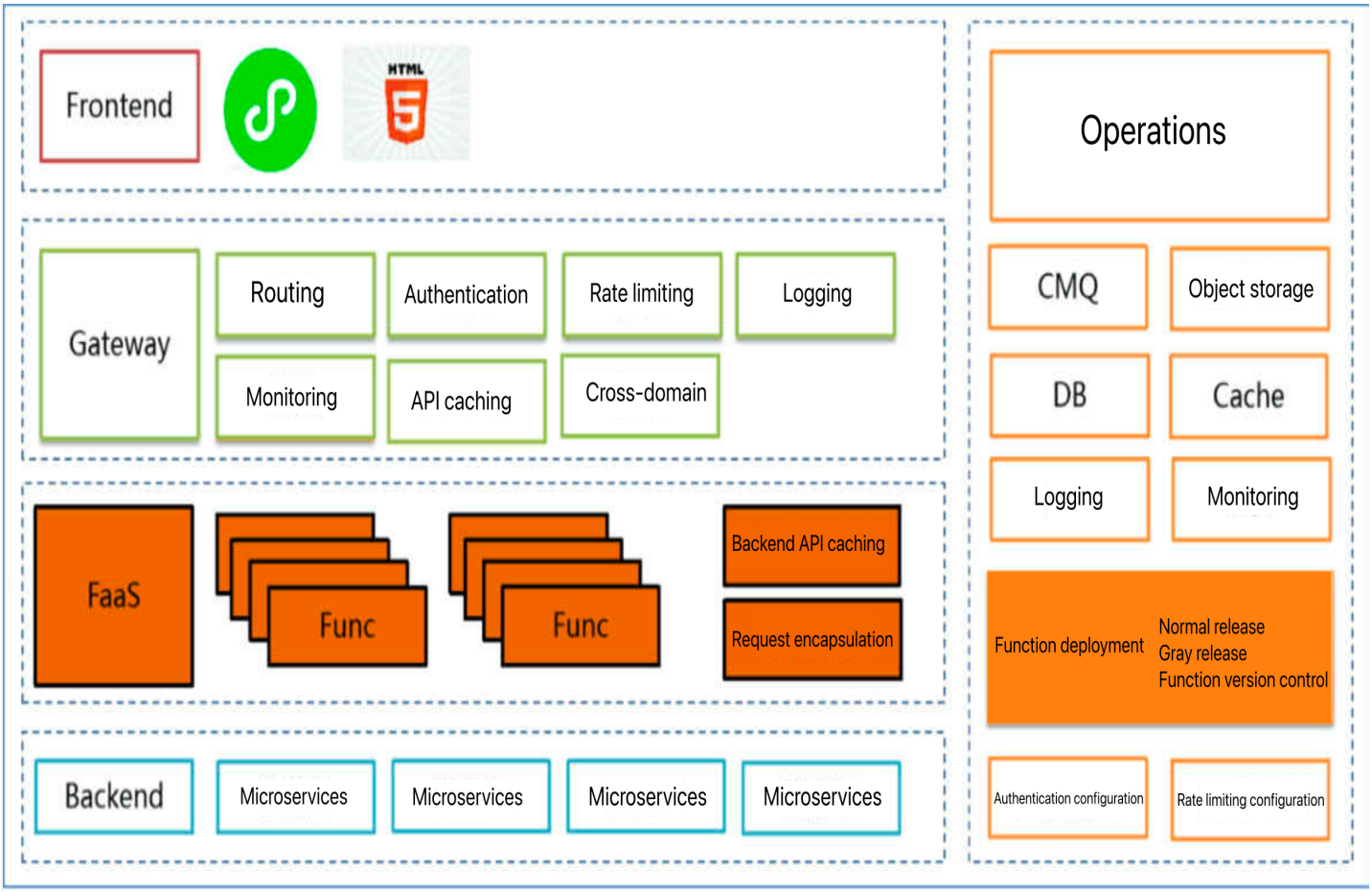
- **Automatic Scaling**
  Developers do not need to specifically configure this, as Cloud Functions can horizontally

scale at the function level based on the volume of requests. Under normal circumstances, an empty Cloud Function (with a runtime of 50ms) with 300 concurrent connections can reach a stress test of 6000+ QPS, which is generally sufficient to handle daily high-concurrency demands.

- **Function-level Development and Deployment**
  Each Cloud Function corresponds to a GitLab project. Function development and deployment are centered around individual projects for efficient and secure CI/CD.
- **Pay-as-you-go**
  Given the traffic characteristics under the financial model, where the volume of requests is relatively low in most cases, the use of Cloud Functions can avoid steady resource investment, significantly reducing costs when idle.
- **Simple Operations Management**
  Developers do not need to maintain services and check logs on the server themselves. With the supporting tools of Cloud Functions, they can easily manage functions, check logs, and set up alarm mechanisms according to their own needs.

Serverless Optimizes Business Architecture

Each adjustment to WeSure's architecture is dedicated to making the responsibilities of various R&D roles more singular and cohesive, and the roles themselves more decoupled. However, such adjustments require corresponding tools, and the trade-offs need to be measured based on short-term costs and long-term benefits. Tencent Cloud's Serverless Cloud Functions have provided excellent support for this architectural readjustment. As shown in the following diagram:



Tencent Online Education

Last updated: 2023-12-05 17:27:39

This article shares a real-life case study of Tencent Online Education's use of cloud functions. The Tencent Online Education Team is:

- The IMWeb team, a subsidiary of Tencent, is one of the leading professional frontend teams in the country.
- Having focused on the frontend field for many years, they have been responsible for billion-level operations such as QQ Profile, QQ Registration, and QQ Groups.
- Currently focused on the field of online education, meticulously crafting three major products: Tencent Classroom, Tencent Penguin Tutoring, and ABCmouse.

An attempt at a technical solution.

The Tencent Online Education team has made numerous technical attempts in the direction of traditional web applications, including traditional offline packages, PWA offline applications, etc., but each technology stack has its advantages and disadvantages. The multi-dimensional comparison of the team's current technical solutions is shown in the following figure:

Solution name	First Screen Time	Development difficulty	Maintenance cost	Selection cost	Network dependency	Client dependency	User experience	SEO friendliness	Logging system
Asynchronous rendering	1.2s ±	Low	Low	Low	Yes	Low	Poor	Fair	Badjs / Sentry
Offline package asynchronous rendering)	900ms ±	Low	Low	Low	No	Yes	Good	Not applicable	Badjs / Sentry
Direct output	700ms ±	Medium-high	Medium-high	Medium-high	Yes	No	Medium	Good	Monitor /Machine log

The comparison reveals that SSR stands out in terms of first-screen rendering and SEO.

Based on these results, the team is more inclined towards SSR technology. When choosing an SSR technology solution, the following two aspects can be considered:

Performance of SSR Applications.

We primarily focus on the following performance optimization issues:

- **Handling a large amount of CPU-intensive computations**: The essence of SSR for applications like React is to call the renderToString method of React on the server side, rendering the React component into an HTML string. For complex SSR applications, this process may involve a large amount of CPU-intensive computations, which is not a domain where Node excels.
- **Enhancing First-Screen Performance**: Given the environmental dependencies of offline packages (dependent on the App), it is worth considering whether a comprehensive solution can be implemented within the traditional Web environment to cache relevant pages, thereby improving the performance of the first screen.

Operational cost of SSR.

Service Availability: Most front-end engineers may not excel in service operation and maintenance, making service availability one of the key considerations during technology selection.

Based on the above two aspects, the considerations can be summarized as the two points in the following diagram:

- How can a method be designed to simultaneously possess the advantages of these three schemes?
- Offline functionality independent of the client environment.
- Short first screen time, superior SEO experience.
- Low maintenance costs, more convenient problem identification.
- Is the solution universal and replicable?

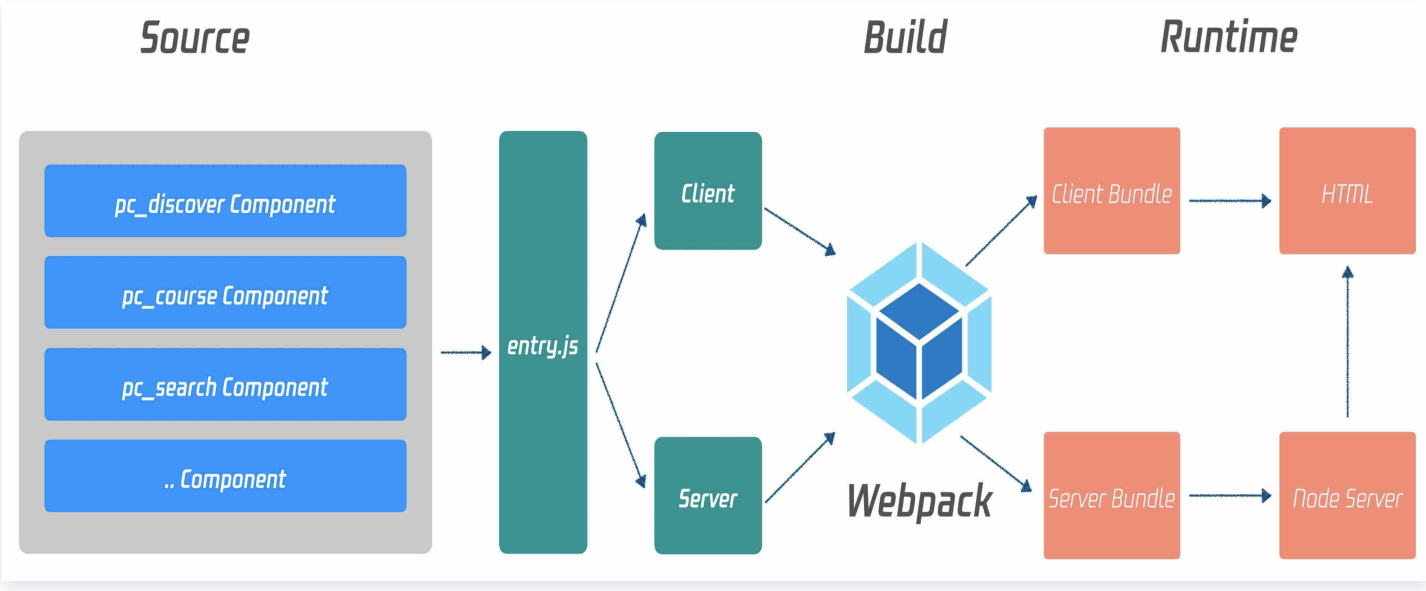
Introduction to Tencent Online Education Team's SSR Architecture Solution

The SSR technology architecture used by the team is as follows:

Next, we will elaborate on the existing solutions of the team from three aspects: code organization, performance optimization, and runtime context.

Code Organization

In both PC and HTML5 projects, an isomorphic pattern is used to build SSR applications. As shown below:



Under the isomorphic pattern, business developers focus more on the functionality of the business itself, rather than overly concerning themselves with runtime issues. However, the following points should also be taken into consideration:

- Traditional usage of constants in browsers, such as `window`, `document`, etc.
- The HTTP data request library must support both the server side and the client side.
- Make judicious use of the lifecycle in React applications.
- Distinguish the current runtime environment by injecting environment variables.

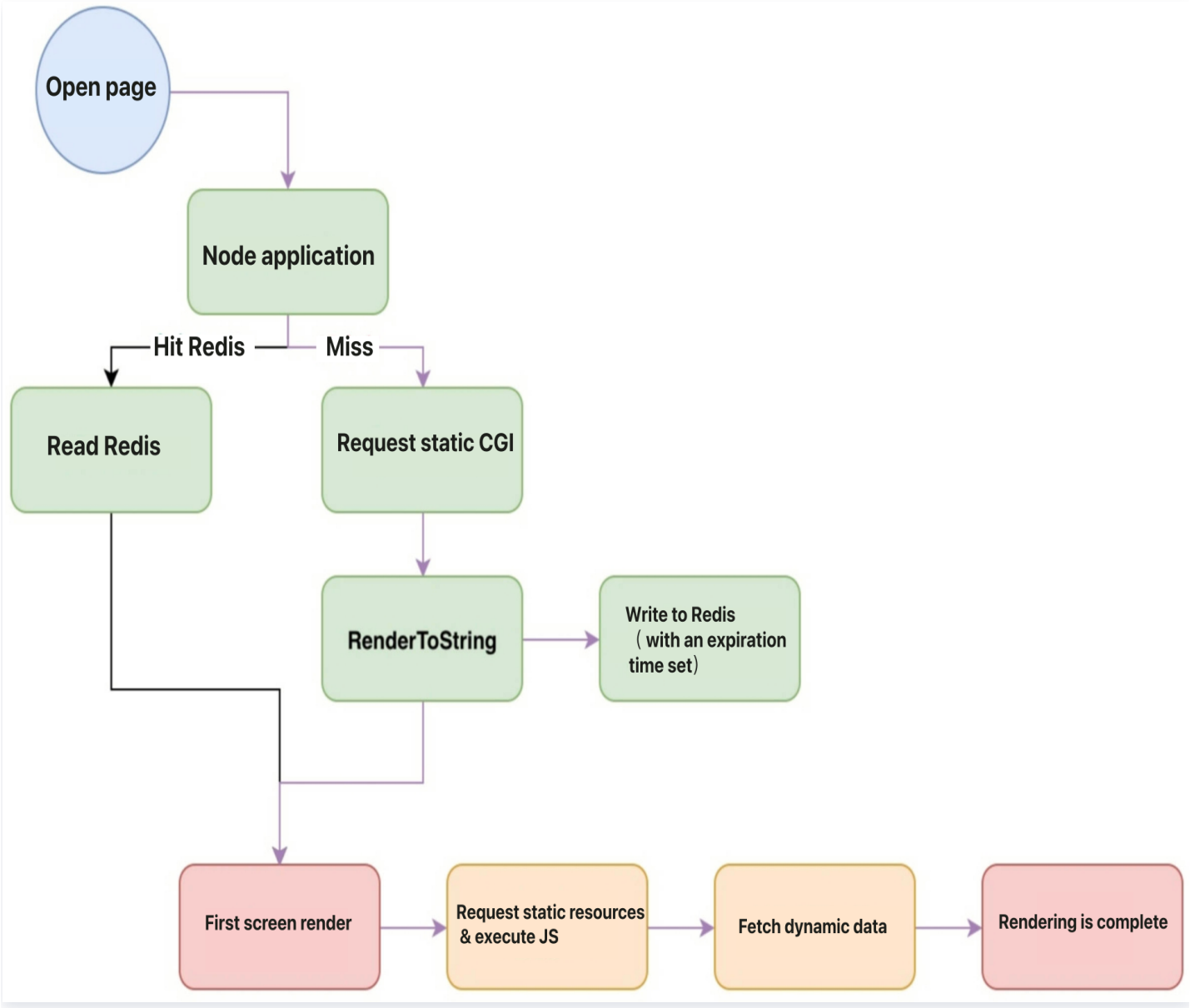
Performance Optimization

Separation of Dynamic and Static APIs

The rendering of a page typically relies on relevant backend data, which can be divided into two parts: dynamic data and static data.

- Static Data: Data within the page that does not frequently change. For instance, the course titles and descriptions in the Penguin Tutoring product.
- Dynamic Data: Data on the page related to the user's login status. For example, whether the course has been purchased, the current course discount, etc.

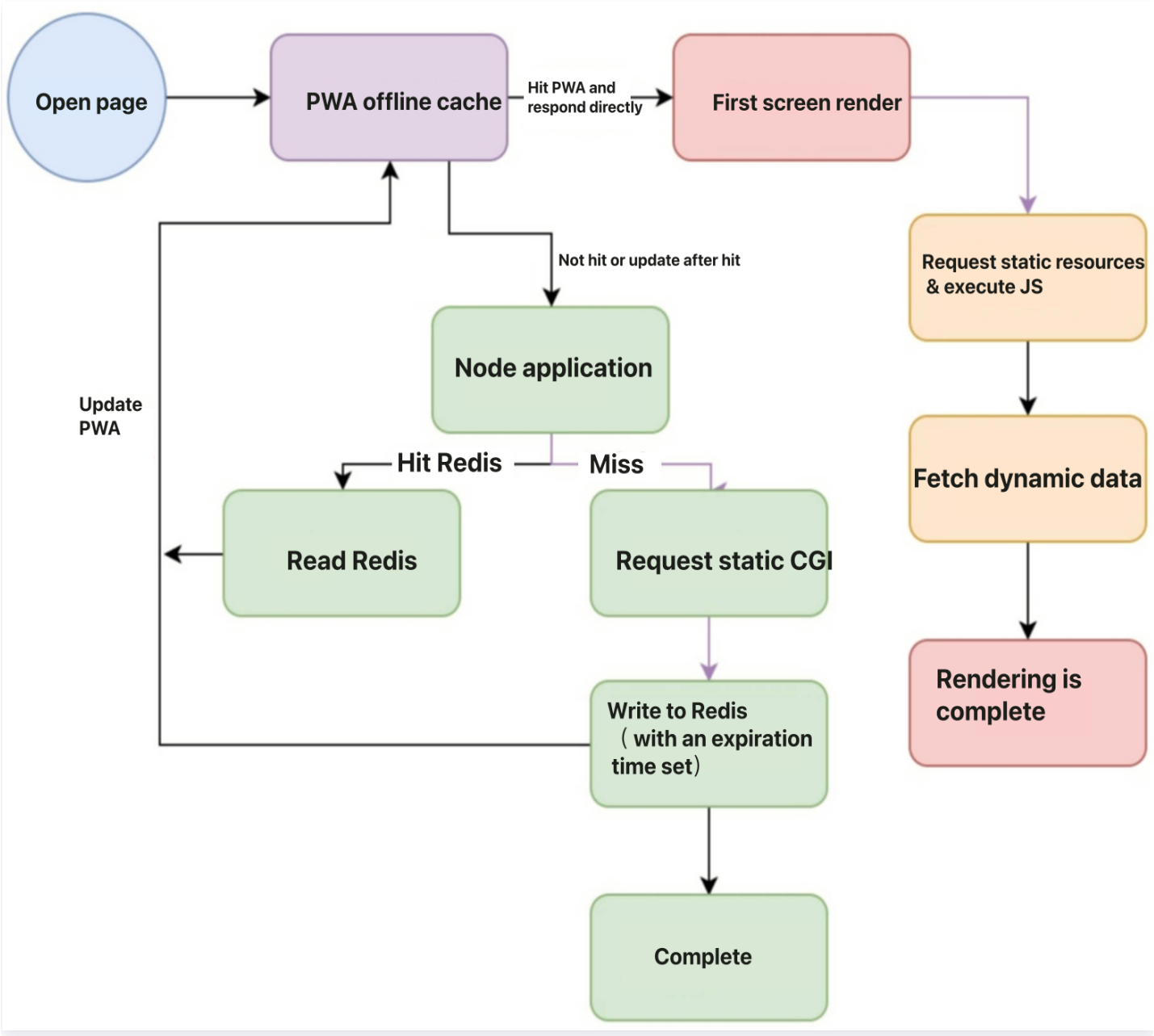
The purpose of separating dynamic and static APIs is to take advantage of the low latency sensitivity of static data for caching. We render the page on the server using static data, and then perform a second rendering on the server using dynamic data. The main logic is as shown in the following diagram:



We use Redis to cache the pages rendered with static data, which not only accelerates the rendering time of SSR, but also improves the QPS of a single machine (`renderToString` is a CPU-intensive operation in a sense).

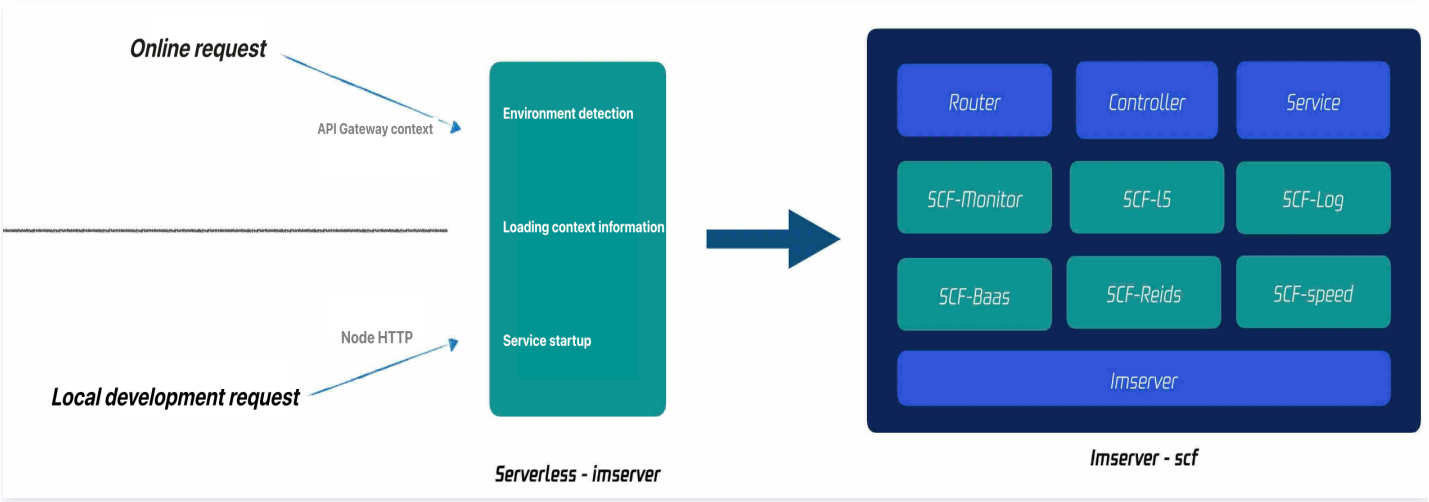
Utilizing PWA for offline caching in the browser

In the client, we can use PWA for offline caching, caching the HTML pages directly output by static data, thereby further improving the first screen performance of the direct output page. The main logic is as shown in the following diagram:

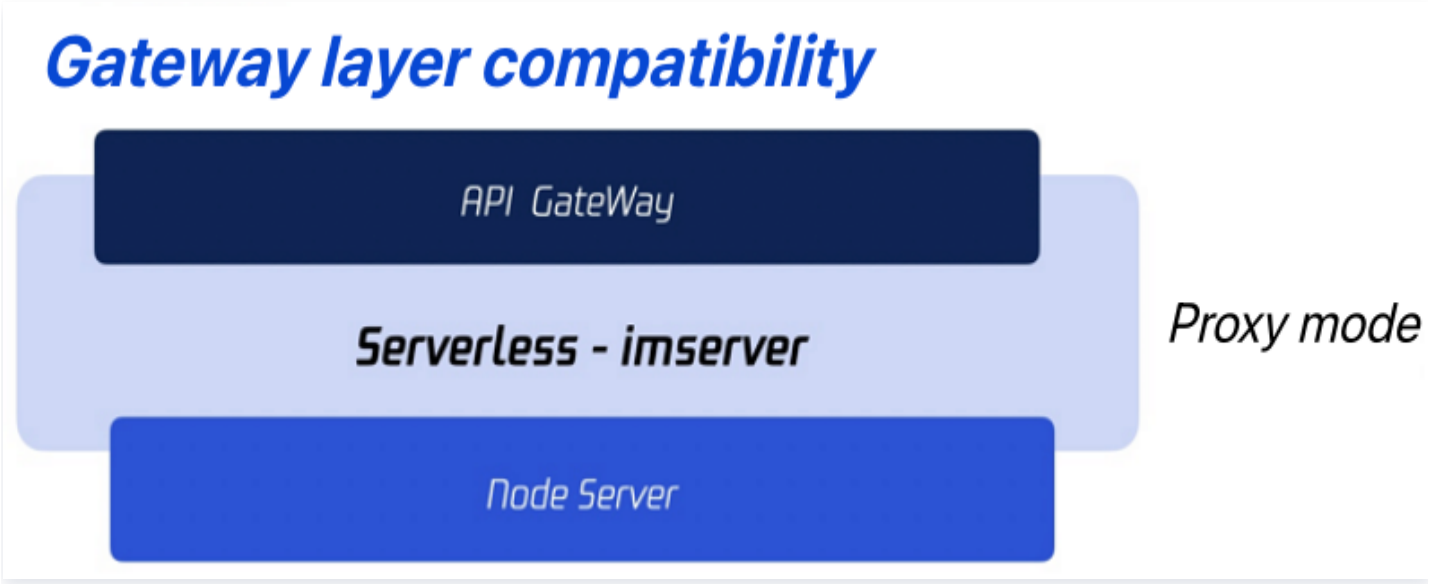


Runtime Context

Due to the complexity of backend application operations and high maintenance costs, we have used Serverless (Tencent Cloud Function SCF) for the deployment of direct output applications. Thanks to the natural advantages of the Serverless architecture pattern, there is no need to worry about service operations and service expansion.



As shown in the figure above, the application of SSR is essentially a Node application, and the call of SCF is essentially an Event event. To address this issue, we have adopted the following methods to accommodate these two modes:
With the help of many standardized interfaces provided by Tencent Cloud Serverless Cloud Framework, we have added a layer of Serverless encapsulation to our self-developed Node framework (imserver). At the same time, compatibility from Event to Koa Request Context was added at the entrance. As shown in the figure below:



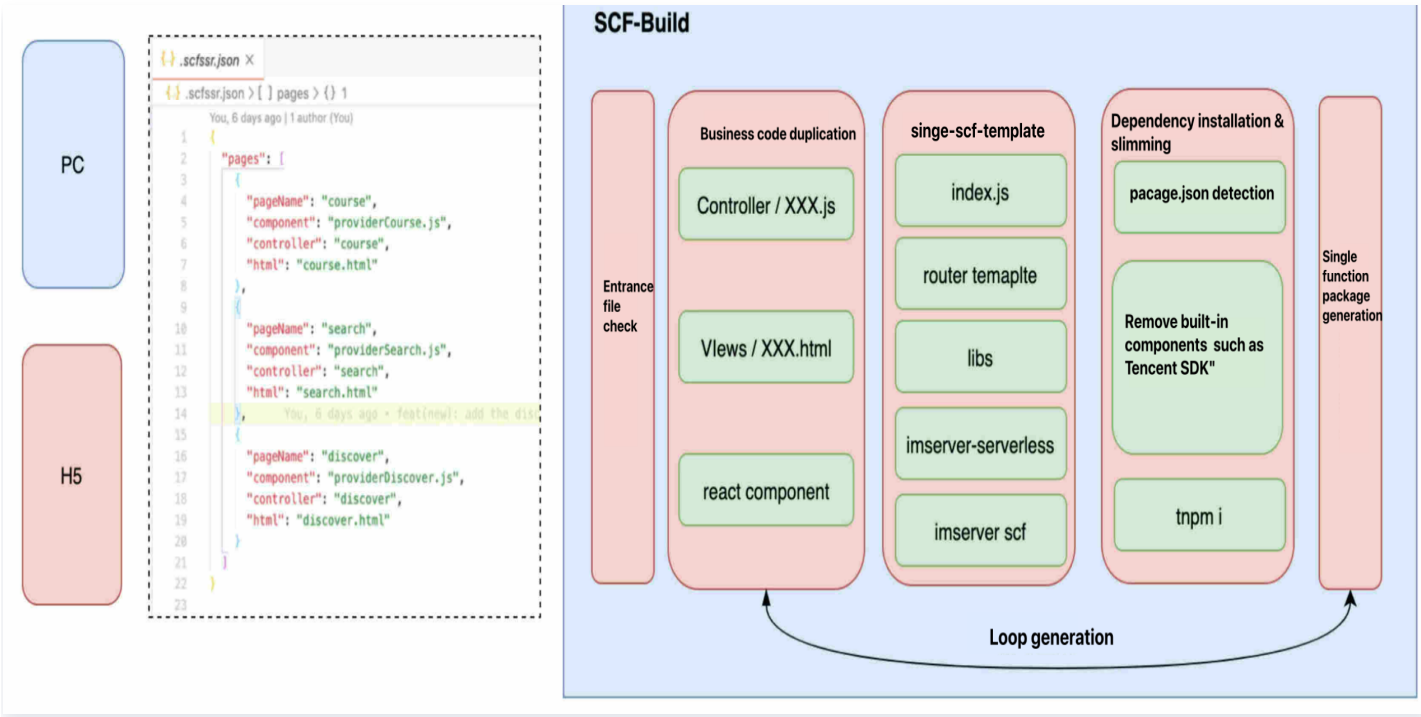
Issues encountered during the implementation of the SSR technical solution

Issue 1: Cloud Function Segmentation

There are multiple pages in the business that are implemented through SSR. After adopting SCF to implement SSR, it is necessary to determine whether to merge into one cloud function (business level) or split into multiple cloud functions (page level). It is recommended to split into multiple cloud functions (page level), with the following advantages:

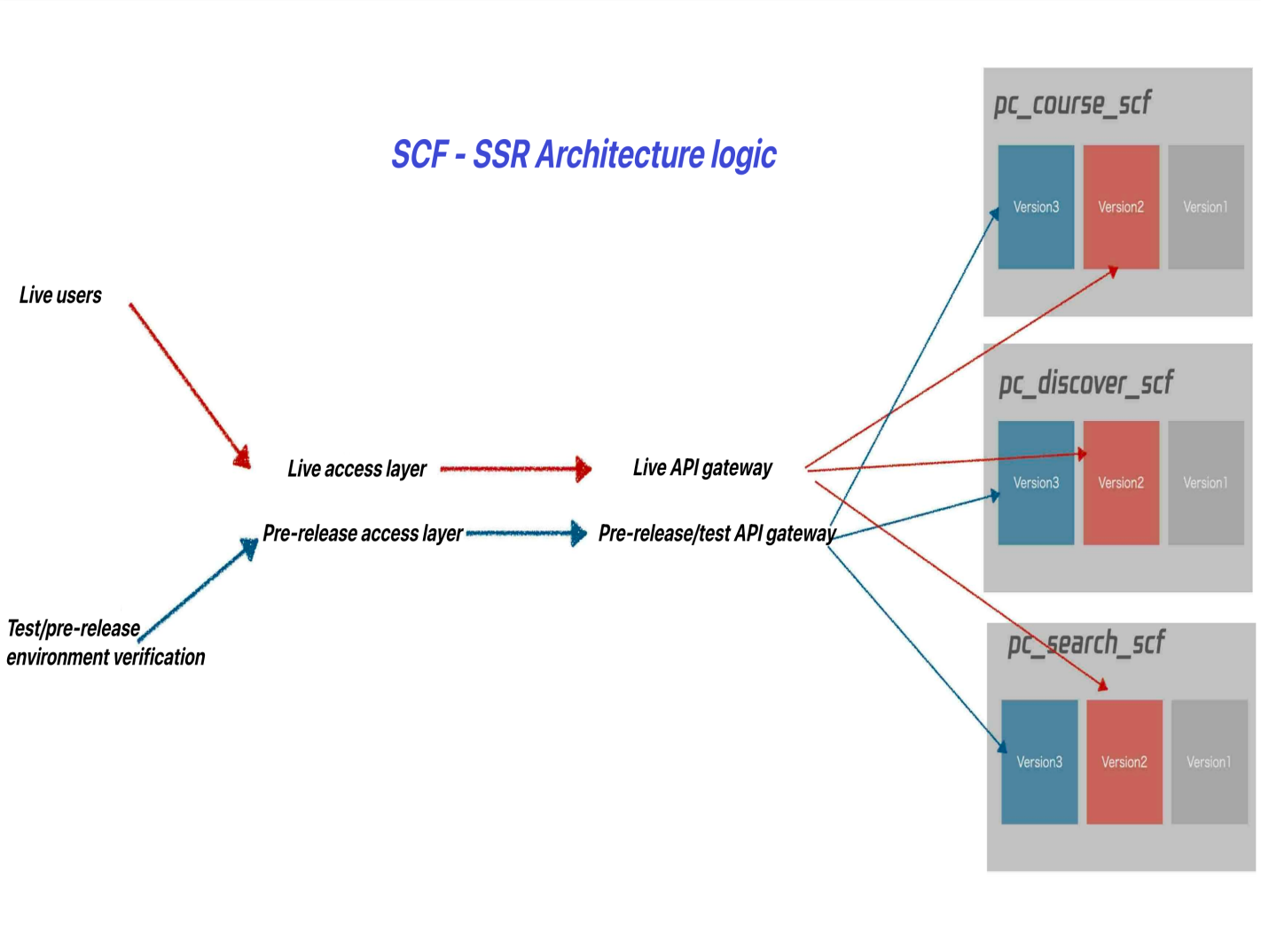
- Cloud functions operate independently. Suppose the cloud function of page A fails, it will not affect the cloud function of business B.
- The size of the cloud function package will decrease. Due to the cold start process of the cloud function, the size of the code package also has a certain impact on the cold start time of the function.

This project has implemented automated construction of cloud functions. Through the configuration file `.scfssr.json`, corresponding cloud functions are automatically generated, which does not affect the existing development work. It only generates multiple cloud functions during construction, reducing the maintenance and development costs of the application.
Simultaneously, based on the construction process of cloud functions, the code of a single cloud function can be made more concise. By analyzing the dependencies in `package.json`, remove the toolkits already built into the cloud function container, and conduct corresponding introduction analysis on the third-party packages that the cloud function depends on, to eliminate redundancy. As shown in the figure below:



Issue 2: Cloud Function Deployment Optimization

The diagram below illustrates our multi-cloud function direct output solution logic based on SCF. It shows that when there is a version update, the release process and steps can be quite complex.



Configuration Process: Initialize Once

Create aliases for release and prohub in the function, which can point to the `$LATEST` version in advance.

- Create Service A in the API Gateway, configure the API Gateway to point to the release alias of Function B, and publish it to the release stage of the API service.
- Modify the API Gateway to point to the prehub alias of Function B, and publish it to the prehub stage of the API Gateway service.
- Modify the API Gateway to point to the default traffic of Function B, and publish it to the dev stage of the API Gateway service.

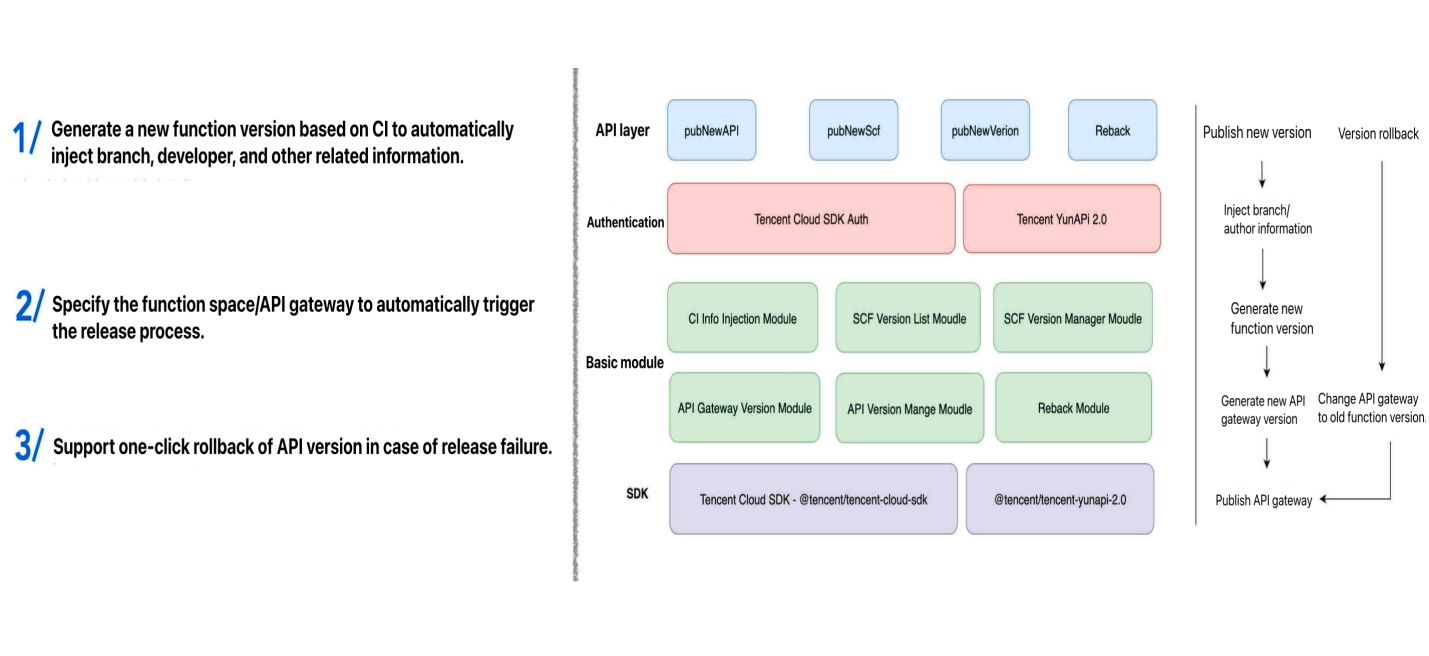
With this, the configuration of the API Gateway is complete, and there is no need for further modifications or publishing configurations on the API Gateway.

Development, Testing, and Release Process: Continuous Development, Testing, Release, and Deployment.

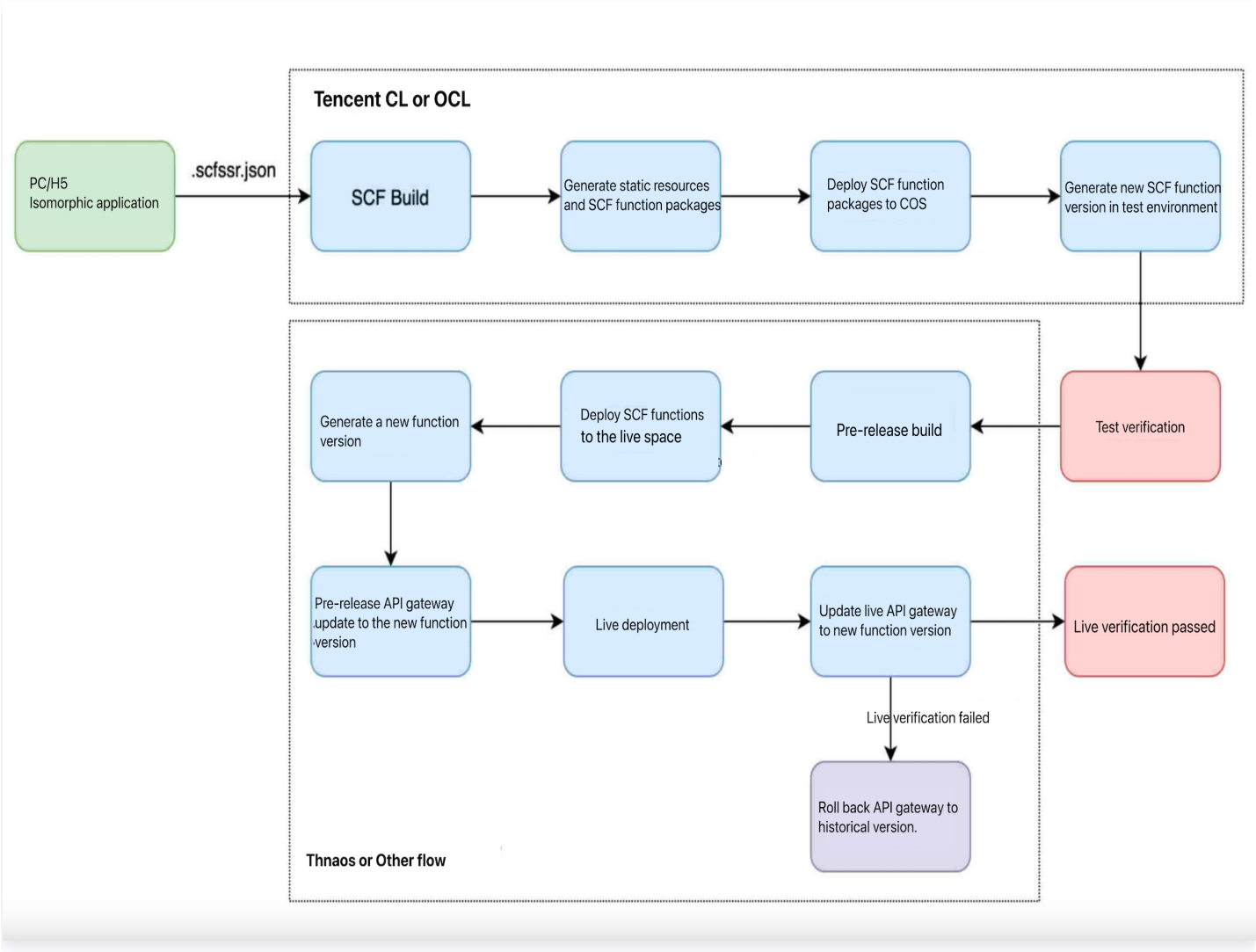
Continuously develop on the function, and sequentially release versions 1, 2, and 3.

- When development and testing of the latest version is required, configure the `$DEFAULT` alias to point to the `$LATEST` version. Continuous development and modifications can be made based on this version, and the version can be published once the modifications are complete.
- When Version 3 is ready for the pre-release environment, configure the prehub alias to point to Version 3. This allows for testing and trial in the pre-release environment.
- Having completed the trial in the pre-release layer, Version 2 is ready for deployment. The release alias can be gradually transitioned from Version 1 to Version 2.
- Monitor and review the logs of the canary release process to determine whether the traffic for Version 2 is increasing as expected, whether the traffic for Version 1 is decreasing as expected, and to identify any errors that occurred during the release process for each version, as well as the overall error situation.

To optimize the release process of cloud functions, we have developed a one-click SCF publishing tool based on the Node SDK provided by Tencent Cloud Serverless. As illustrated below:

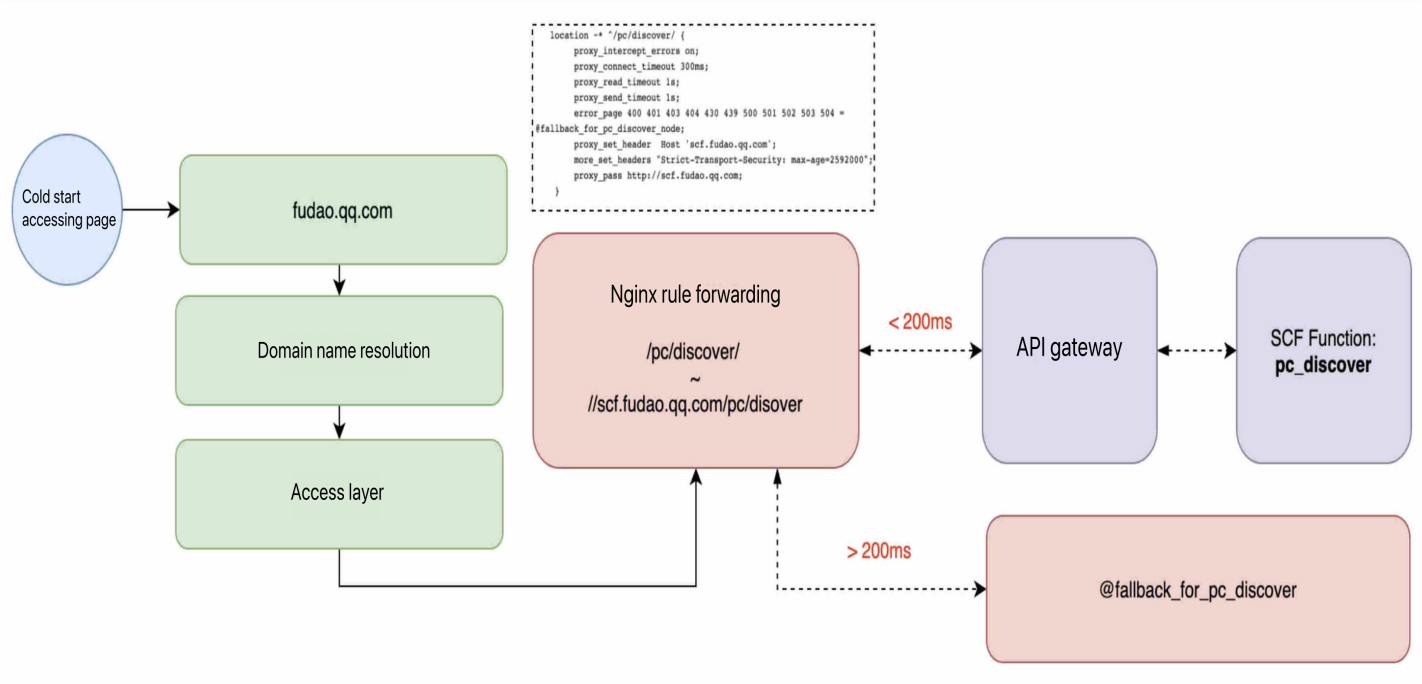


A complete lifecycle of an SCF SSR application is depicted as follows:



Advantages and future plans of Tencent Cloud's Serverless SSR solution

Utilizing the SSR solution based on SCF significantly reduces the cost of service operations and maintenance. Leveraging Tencent Cloud Serverless's logging system, every single SSR application request has a complete link in the log platform, which greatly enhances the speed of problem identification and resolution.
Serverless architecture patterns have the issue of longer cold start times. SCF has technically optimized this issue, such as pre-starting containers. In terms of actual business, we can also attempt to optimize. For instance, service degradation optimization has been done at the access layer. As shown below:



Future optimization plans can be improved from aspects such as canary releases and multi-dimensional degradation.

Recommendations for using SSR technology

- To pursue a superior user experience, it is recommended to optimize SSR for core business operations, coupled with Serverless for service deployment and operations. With the support of Serverless, we no longer need to concern ourselves with machine maintenance and expansion as before, further enhancing team productivity.
- After using SSR, it is advisable to further refine your business's devops process, connecting the entire research and development chain. This allows for efficient progression from development to testing and finally to deployment.
- It is recommended to use the business access layer for service degradation, thereby enhancing the availability of SSR applications.