System Requirements Specification (SyRS)

CMPT475 Spring 2022 – Group 10

Dezheng Zhong & Yuanying Mu & Wenqing Liu & Siwei Wang & Tongfang Zhang

1. Introduction

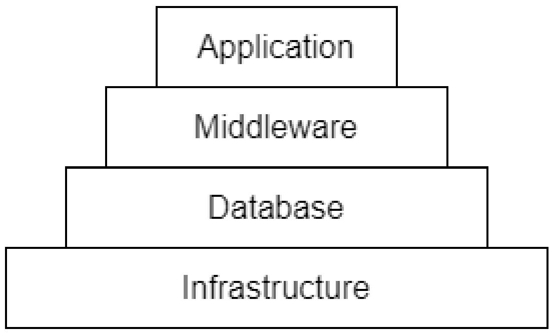
1.1 System Overview

The characteristic of the software is that at noon and evening time, business volume grows significantly. It is because people always take meals during this time and they may prefer to use the software to calculate the calories before the meal. Also, during the peak periods, the limit of queries per second (QPS) of the main service can increase to 10,000, also, the database faces the challenge of growing number of requirements, its QPS can be at least 5,000 for customers to request and the transaction per second (TPS) should be 3,000.

1.2 System Purpose

Because of the pressure of large system traffic, the main work is to increase the software capacity and maintain the stability. We need to build a high concurrency system from the aspects of infrastructure, database, middleware and application.

1.3 System Scope



1.3.1 Infrastructure

Infrastructure is the basic part of our system, it usually includes firewalls, physical servers and so on. We need a comprehensive infrastructure so that the software can finish basic tasks.

1.3.2 Database

For a high concurrency system, the main challenge is that the database cannot hold a large number of concurrencies instantly. To maintain the software's stability, one solution is to use sub databases and sub tables. We need to divide the main database into several databases so that they will resist high concurrency together. In order to improve SQL running efficiency, we use sub tables to make less data in each table.

1.3.3 Middleware

Middleware is the mediator between the front-end and beck-end, it helps us avoid linking to the background resource while building the application. As our database faces high concurrency and needs to be read many times, we choose to use database middleware which can simplify the connection to the database.

1.3.4 Application

To build a high concurrency system, we need to modify the application layer. This part is about the optimization on asynchronization, back-pressure, memory copy and resource consumption.

1.3.5 Cache

One characteristic of our system is "read more and write less". Customers prefer to check their record often but seldom modify it. We must use cache, it can reduce the time of requesting data. We need to make a copy in database and cache, then use much cache to read it. This way can help our database so that it will not be finished because of high traffic.

2. Reference

Spacey, J. 35 Examples of Infrastructure Software. Simplicable. Retrieved March 27, 2018
https://simplicable.com/new/infrastructure-software

Chandrakant,K. Design Principles and Patterns for Highly Concurrent Applications. Baeldung. Retrieved March 31,2022 , from
https://www.baeldung.com/concurrency-principles-patterns

IBM Cloud Education. Middleware.   IBM. Retrieved March 5,2021 from
https://www.ibm.com/cloud/learn/middleware

How to design a high concurrency system?. DevelopPAPER. Retrieved September 13,2020, from https://developpaper.com/how-to-design-a-high-concurrency-system/

3. System Requirement

3.1 Functional requirements

3.1.1 Data coverage and completeness

Food datasets that are used for training should include the most common food categories. Also, the datasets can be scalable once new food images are uploaded. The food density table and calories table should be accurate to make sure the calculation result is reliable and accurate.

3.1.2 Optimization

a) The system should figure out the best computation logic for these gathered data from users.

b) The system should consider integrating factors such as CPU, GPU, memory, storage, and network.

c) The system should use data tracking records, export display, query statistics, modify and update to handle the data and make responses.

3.1.3 Balance

The contradicted relationship between different components can affect the performance of the whole system. Thus, it is necessary to establish a balance between potential contradicted sides and deliver a satisfactory system.

a) Data processing efficiency and data processing capacity

The system will be fed a large volume of data every day but the server along with the database can only process data up to the limit. The processing efficiency can be reduced if the capacity increases.

b) System construction and construction cost

The system should be constructed with fulfilled business needs and user requirements. On the other hand, the budget limits the system construction to those features that can bring large profits.

c) System maintenance and maintenance cost

The system should be maintained to make sure it can deliver quality service to users 24/7. On the other hand, the labor cost restricts the maintenance frequency.

d) System ease-of-use and customer requirements

The system should still obey the rule of ease-to-use in the design phase while fulfilling customer requirements as much as possible. For those requirements contradict the goal, the system designer should find the balance between these two.

3.2 Usability requirements

3.2.1 Limit flow

a) When the traffic volume reaches the maximum threshold that the server can handle, users can encounter unexpected screen freezes, delays, no responding issues, etc. In this case, the stable throughput capacity of the system is essential for a fluent user experience.

b) The application should utilize a highly concurrent system that adopts a rapid failure mechanism.

c) The system should adopt a traffic limit to guarantee the normal operation of the server. If the current traffic is about to exceed the upper limit, the system should generate an error message, such as “Please try again later” within 5 seconds of the issue.

3.2.2 No status

The system should adopt stateless with no resident memory used to store information between sessions. In this case, it will be easier to deploy the services to any number of servers and no session (state) synchronize logic to handle at the server-side so which effectively reduces the complexity.

3.2.3 Error Tolerance

a) The system should divide each business need into small modules, such as homepage, account, ingredients table, payment, and so on.

b) The front side can be cached to provide more fast and more reliable services.

c) The payment page should adopt remote disaster recovery to detect unexpected emergencies within 1 minute and report to the development team.

3.3 Performance requirements

3.3.1 Response time

a) The system should have a fast response mechanism so the average response time for users’ clicks and scrolls should be less than 1 second.

b) The real-time data processing time to do the query operations should not exceed 5 seconds.

c) The transaction processing time for users should be less than 5 seconds after payment confirmation. Any updated account information should display to users within 2 seconds after closing the transaction.

3.3.2 Maintainability

The system should decompose the whole into small components with standardized interfaces so that the maintenance can be easily implemented in a modular manner.

3.3.3 Scalability

a) The system should adopt a modular design and each distinct feature should be put into a separate part.

b) The system should adopt a highly scalable database so that it can handle millions of food images from users every day.

c)The food datasets for training should be updated every day once it receives unfamiliar images from users.

3.3.4 Reliability

The system should provide comprehensive reliability from the database to the back-end to the front-end with a mean time between failures (MTBF) greater than 10,000. The probability of failure on demand (POFOD) should be less than 0.005.

3.3.5 Availability

The system should provide service 24/7. The server should get requests and return responses to users within 5 seconds. If any unexpected crashes happen, the system should generate an error message within 5 seconds.

3.4 Interface requirements

3.4.1 External interface requirements

a) Provide interfaces that support access to external systems, offering secure and reliable access.

b) Provide an effective system monitoring mechanism to detect errors in time.
c) Provide a complete information security mechanism to achieve comprehensive protection of information.

d) The system should operate normally under a large number of accesses.

e) HTTP should be used as the communication protocol for the application.

3.4.2 Internal interface requirements

a) The control of the interface should be smooth. More specifically, the response time for user to click and scroll should be less than 1 second.

b) Buttons should be evenly distributed in the interface and spaced in an orderly manner to prevent users from accidentally touching them.

c) The design of the interface should be visually pleasing and simple with consistent colors.

3.5 System operations

3.5.1 Human system integration requirements

a) Manpower

The manpower should be enough to design, operate, maintain, train, and support the full capability of the application.

b) Personnel

The personnel need to have the skillset, knowledge base and capabilities required to develop the application. In addition, optimize personnel performance by evaluating workloads and tasks to ensure that each developer is able to complete tasks without becoming saturated with tasks.

c) Training

Training will be required for the maintainers of the system.

d) Occupational Health

The task load and fatigue factors of developers should be considered.

3.5.2 Maintainability requirements

a) Time

Reaction time should be less than 1 second.
Downtime should be less than 5 seconds.
Turnaround time should be less than 5 seconds.
Meantime between maintenance actions: 460 hours
Meantime to repair: 20 minutes
Meantime between failures: 3 hours

b) Rate

Maintenance staff hours per specific maintenance action: 4 hours
Operational ready rate: minimum 90%
Frequency of preventative maintenance: 7 days

c) Maintenance complexity

Number of people: at least one full-time maintenance staff for every 100 units
Preventative maintenance component: user and/or service manual
Repairing component: faults and breakdowns reporting system

d) Maintenance action indices

Maintenance costs per operating hour: $4
Average staff hours per overhaul: 7 hours

e) Accessibility

The software system should be used with the widest range of capabilities to achieve a specified goal in a specified context of use.

3.5.3 Reliability requirements

The system should consistently perform the specified functions without failure. Consider the probable causes of system failures, the preventive measures or processes needed to avoid failures, failure categories, and reliability indicators.

3.5.4 Other quality requirements

a) Integrity

The data maintained by the software system should be accurate, authentic, and without corruption. The need for daily backup of data to prevent the loss, the data recovery process, and the authenticity of the data relative to the original data source should be considered.

b) Portability

The software system should be easily transferred from its current hardware or software environment to another environment. In other words, the application shall be used from two operating systems: iOS and Android.

3.6 System modes and states

3.6.1 Dual-mode state

Since our application supports the iOS platform. Due to the Apple App Store requirements, the application must maintain a dual-mode state indicating whether it is in AppConnect mode. The possible states are:

a) Undecided

The application has initialized for the first time and has not yet decided whether to run in AppConnect mode or Non-AppConnect mode.

b) AppConnect mode

The application is running as an AppConnect app for enterprise users. It supports the AppConnect features, such as authorization, data loss prevention, and secure file I/O.

c) Non-AppConnect mode

The application is running as a regular application for general customers. It supports none of the AppConnect features.

d) Pending AppConnect mode

If the users explicitly request a change to AppConnect mode, the application will change to this state.

3.7 Physical Characteristic

3.7.1 Adaptability requirements

In order to ensure that the server can run stably during peak periods, so that users will not wait for the server to respond for too long, the server configuration should meet the following standards:

Processor:

1.4 GHz 64-bit processor
Compatible with x64 instruction set
Supports NX and DEP
Supports CMPXCHG16b, LAHF/SAHF, and PrefetchW
Supports Second Level Address Translation (EPT or NPT)

RAM and external storage:

Spare memory capacity should be > 20%
Spare external storage should be >70%

Internet:

An ethernet adapter capable of at least 1 gigabit per second throughput
Compliant with the PCI Express architecture specification.
A network adapter that supports network debugging (KDNet) is useful, but not a minimum requirement.
A network adapter that supports the Pre-boot Execution Environment (PXE) is useful, but not a minimum requirement.

3.7.2 Physical requirements

To protect the steel plate from corrosion, rust or wear, the cabinet will have the surface properly treated. In addition to conventional coatings, there are usually special measures, such as galvanizing and phosphating, to isolate the steel sheet from the air.

3.8 Environmental conditions

The engineering design of the building, structure, heating and ventilation, power supply, lighting, fire protection and other projects of the communication room must be strictly based on the environmental design requirements of the communication equipment, and should also comply with the local standards, specifications, and the regulations on the building design in the special process design. and requirements.
When carrying out engineering design, an address that meets the engineering environment design requirements of communication equipment should be selected according to the communication network planning and technical requirements of communication equipment, comprehensively considering factors such as hydrology, geology, earthquake, electricity, transportation, etc.
In order to ensure that the communication equipment is in a good operating environment, the location of the communication equipment room should avoid the environment with high temperature, dust, harmful gas, explosive and unstable voltage, and avoid places with frequent large vibration or strong noise.
For heavy pollution sources such as smelters and coal mines, the distance should be more than 5km.
For chemical, rubber, electroplating and other medium pollution sources, the distance should be more than 3.7km.
For light pollution sources such as food and leather processing plants, the distance should be more than 2km.
If these pollution sources cannot be avoided, the computer room must be selected in the perennial upwind direction of the pollution sources.

The air outlet for air exchange in the computer room must be kept away from the air outlet of the urban sewage pipe, large septic tanks and sewage treatment tanks, and the computer room should be kept in a positive pressure state to prevent corrosive gases from entering the computer room and corroding components and circuit boards. The machine room should avoid industrial boilers and heating boilers.
The computer room should not be located on a dusty roadside or a gravel field. If it is unavoidable, the doors and windows must be away from the pollution source.

3.9 System security requirements

3.9.1 Only administrators can log in to the management background through the web page, and ordinary users have no chance to tamper with background data.

3.9.2 DDoS native protection adopts passive cleaning as the main method and active suppression as the auxiliary method. For DDoS attacks, on the basis of standard technologies such as reverse detection, black and white list, and packet compliance, it is ensured that protected users can still be protected when the attack continues. External business services can be provided.

3.9.3 Vulnerability detection and security risk assessment: Identify the system resource of the inspected object, analyze the possible index of this resource being attacked, understand the vulnerability of the support system itself, and evaluate all existing security risks.

3.9.4 Invulnerability: device backup mechanism, fault tolerance mechanism, to prevent the system's backup mechanism to ensure the normal operation of the system when a single point of failure occurs in the system.

3.10 Information management requirements

3.10.1 Establish procedures to protect system data to view a log of each addition or deletion of data. Collection of inspection and reporting of information data to prevent illegal access and malicious tampering.

3.10.2 Use products from reliable firewall companies to prevent user data from being leaked.

3.10.3 Establish effective backup measures to prevent data loss due to hardware or software failure. Regularly check the validity of backups. Periodically review and document the process and results of computer processing of data.

3.11 Policy and regulation requirements

3.11.1 Ensure that all information security employees and contractor personnel are assigned a level of positional sensitivity commensurate with the responsibilities and risks associated with the position.

3.11.2 Enforce discipline against employees or contractors who violate relevant policies and rules.

3.11.3 Establish system rules for system and network behavior. All system users are required to read, acknowledge, and comply with their roles and responsibilities and the system's security rules.

3.11.4 Ensure that responsibility for security impacts is shared among multiple employees by implementing the concept of segregation of duties, which requires that individuals have no control over the entire critical program.

3.11.5 Require appropriate background checks to be completed and favorably adjudicated for personnel assigned to these positions.

3.12 System life cycle sustainment requirements

3.12.1 System Planning Phase
It addresses the "what is" question, what the system means, what are the advantages of the systems used by everyone in the market, and how the new system should compare to the use of these systems.

3.12.2 System Analysis Phase
The main problem is to solve the problem of "what to do" in the system. The most important aspect of requirements analysis is to translate the actual problems into computer problems and then solve them by computer. Models are used to solve problems at this point.

3.12.3 System Design Phase
System design is divided into overall design and detailed design phases.
The overall design is the design of the system architecture, the high-level structure, i.e., the system architecture is the operating model of the program, the hierarchy, the call relationships, the planning of specific implementation technology types, etc. The high-level structure refers to the subsystem division of labor, the interface design. In other words, MVC, Spring and other common architectures.
Detailed design includes many parts, code design, output design, input design, human-computer dialogue design, module detail design, database design, network design.

3.12.4 System implementation phase
It consists of two parts: coding and testing. Coding is the process of program design and implementation. Good coding standards are followed and the program structure is designed.
Testing is a series of sub-processes, unit testing->integration testing-> acceptance testing- system testing. 80% of the problems usually occur in 20% of the modules.
Pay attention to the designed test cases (with sufficient coverage, pay attention to

test boundary values), use white box testing, black box testing to complete the whole testing process.

3.12.5 System maintenance phase
Maintain system errors, maintain new features, maintain adaptation to new operating environments, and maintain to prevent future problems.

4. Verification

4.1 Functions verification

4.1.1 Data coverage and completeness

Verify that existing and publicly available food datasets contain the most common food. Verify that data in the food density table and calories table is matched with the data issued by the certified organization.

4.1.2 Balance

a) Data processing efficiency and data processing capacity

Verify that the system is able to process all current users' requests within 5 seconds and support uploading 5 million food images every day without encountering any delay or freeze.

b) System construction and construction cost

Verify that the system matches customer requirements and does not exceed promised budgets.

c) System maintenance and maintenance cost

Verify that the system can operate normally and respond to any error code within 1 minute. The maintenance team should fix any issues once the error logs are submitted. Also, make sure the labor and machine cost does not exceed the upper limit

d) System ease-of-use and customer requirements

Verify that the system is easy to use so that new users with an average age of 55 can get calories information from uploading images within 7 minutes. On the other hand, verify that the system matches all requirement descriptions in the Stakeholders Requirements Specification.

4.2 Usability verification

4.2.1 Limit flow

Set the upper limit of query per second (QPS) as 10,000 and verify that all additional queries can generate an error message, such as “Please try again later” within 5 seconds.

4.2.2 No status

Verify that all requests sent to the backend server do not depend upon data from a previous session.

By using the stateless model checking to verify that there is no safety issue and data leaking.

4.2.3 Error tolerance

Maintain the server and shut down unnecessary parts if needed to carry out tests.

Verify that front-end, back-end and database can send requests and receive responses within 10 seconds.

4.3 Performance verification

4.3.1 Response time

Verify that the average response time is less than 1 second when the QPS is 8000. Create 8000 in-app requests to simulate users' behavior and verify that the average response time for each request is less than 5 seconds.

4.3.2 Maintainability

Verify that each component is in accordance with the preliminary design and that each class does its work in separate parts.

For any new classes that have to be added, verify that they are also added to the preliminary design for future references.

4.3.3 Scalability

Verify that the system can keep stable performance when the food datasets are updated without committing any changes to existing code and server.

4.3.4 Reliability

Keep sending requests to the server to simulate a real software environment to verify that the mean time between two consecutive failures is greater than 10,000 and the probability of a failure happening when a service request is made is less than 0.005.

4.3.5 Availability

Keep running the system for a whole day without any manual interactions and collecting failure events and recovery times. Verify that the availability percentage matches the original service level agreement.

4.4 Interface verification

4.4.1 External interface requirements

a) Verify that the database is encrypted. Protect the database behind a firewall and encrypt all data at rest.

b) Verify the efficiency of the software system in terms of processing capacity, throughput, response time, and storage capacity.

c) Protect the database behind a firewall and encrypt all data at rest.

d) Ensure the stability of the system.

e) Verify the communication protocol by ensuring it has no logical errors.

4.4.2 Internal interface requirements

a) Verify the response time of clicking and scrolling is less than 1 second.

b) Verify that the probability of a false touch is small.

c) Verify the interface design is visually pleasing.

4.5 System operations verification

4.5.1 Human system integration requirements

a) Manpower

Verify the existing manpower is sufficient for the development and maintenance of the application.

b) Personnel

Verify that the personnel have the appropriate skills, and the workload is distributed reasonably.

c) Training

Verify the maintainers are well-trained.

d) Occupational Health

Verify the developers are in good physical and mental condition.

4.5.2 Maintainability requirements

a) Time

Verify the reaction time is less than 1 second.
Verify the downtime is less than 5 seconds.
Verify the turnaround time is less than 5 seconds.
Verify the meantime between maintenance actions is about 460 hours.
Verify the meantime to repair is about 20 minutes.
Verify the meantime between failures is approximately 3 hours.

b) Rate

Verify the maintenance staff hours per specific maintenance action is about 4 hours.
Verify the operational ready rate is greater than or equal to 90%.
Verify the frequency of preventative maintenance is about 7 days.

c) Maintenance complexity

Verify there is at least one full-time maintenance staff for every 100 units.
Verify there is a user and/or service manual for preventative maintenance.
Verify there is a faults and breakdowns reporting system for repairs.

d) Maintenance action indices

Verify the maintenance costs per operating hour are approximately $4.
Verify the average staff hours per overhaul is 7 hours.

e) Accessibility

Consider legislation and standards, as well as specific needs such as visual, cognitive and mobility to verify the accessibility of the system.

4.5.3 Reliability requirements

Verify the system failure resistance.

4.5.4 Other quality requirements

a) Integrity

Verify the data accuracy and authenticity in the system.

b) Portability

Verify the ease of system transfer, considering aspects related to data, procedures, end-user and developer documentation.

4.6 System modes and states

4.6.1 Dual-mode state

a) Undecided

Verify the application starts in undecided mode.

b) AppConnect mode

Verify the system can connect to the other components and turning off management can go to Non-AppConnect mode.

c) Non-AppConnect mode

Verify the system cannot connect to the other components but turning on management can move into Pending AppConnect mode.

d) Pending AppConnect mode

Verify that from this mode, the system can go to AppConnect mode or Non-AppConnect mode.

4.7 Physical characteristics verification

Verify that the system can be deployed on the server with the configuration specified in 3.7.1. Also verify that the spare capacity of memory and external storage have met the standard of 20% and 70% respectively.

4.8 Environmental conditions verification

a) Verify that the temperature and humidity should be 20-25 degrees and 40-55% respectively.

b) Verify that the power supply consists of one or more uninterruptible power supplies (UPS) and/or diesel generators as backup power sources.

c) Verify that the floor of the computer room is raised by 80 cm relative to the tile floor to provide better even distribution of airflow.

d) Verify that the dry noise in the room when the machine is turned on should be less than 70db(A) when measured at the central console.

e) Verify that the radio interference field strength in the computer room is not more than 120dB in the frequency range of 0.15-1000MHz. The magnetic field interference field strength in the computer room is not more than 800A/m.

4.9 Security verification

4.9.1 Verify that the accounts of managers are only holding by the staff, also verify that the accounts are well encrypted.

4.9.2 Verify that we can quickly switch to a backup website in case of a DDOS attack, also verify that we can defend against attacks below 100Gbps.

4.9.3 Verify that all components have backup mechanisms to fail gracefully by forcefully shutting down key components that could potentially be a single point of failure.

4.10 Information management verification

4.10.1 Verify that queries are logged for a week and a system is in place to analyze the log to be readable for developers.

4.10.2 Verify that the firewall company has long-term service

4.10.3 Verify that monthly snapshots of data are created as back-ups and that back-ups are stored for up to three months.

4.11 Policy and regulation verification

4.11.1

Ensure that all new employees responsible for handling sensitive data undergo a thorough background check, allowing for review of decisions outside of the hiring process.

4.11.2

Verify that employees with managerial authority are divided into different tiers, with the authority increasing as the employee gains higher positions within the company.

4.11.3

Verify that documentation exists that outlines system behavior rules for the system and network, and test all new employees to ensure they understand the rules.

4.11.4

Verify that documentation exists that appropriately penalizes different violations of relevant policies and rules in order to fairly punish violators.

4.11.5

Verify that permissions are limited to the minimum required for each individual role. Also verify that employees cannot gain new permissions unless they cannot work with other employees who have sufficient permissions.

4.12 System lifecycle maintenance verification

4.12.1 System Planning Phase

Verify that a system plan exists based on market research and comparison with existing systems.

4.12.2 System Analysis Phase

Verify that the model used to solve the problem is identified in a single document.

4.12.3 System Design Phase

Verify that the documentation outlining the overall architectural design of the system exists.

4.12.4 System Implementation Phase

Verify that the system has at least 80% test coverage. In addition, verify that all components are white and black box tested.

4.12.5 System Maintenance Phase

Verify that a plan exists to maintain the system after the initial development cycle.

5. Appendix

AI - Artificial intelligence is the ability of a computer or a robot controlled by a computer to do tasks that are usually done by humans

AWS - Amazon Web Services, provides servers, storage, networking, remote computing, email, mobile development, and security.

iOS - iPhone Operating System, is a Unix-derived operating system powering all of Apple's mobile devices.

Inc. – incorporated, a company's business structure is a legal corporation.