Editor's Note: The design of B-side education products often needs to go through many processes, involving all aspects of the education industry. When we design a more complex functional module, how should we reduce the coupling between modules so as to make a low-coupling design? The author has combined his experience to sort out and analyze, and summarized the methodology of how to achieve a simple and elegant design.

As a B-side product manager for an educational institution, you often need to design processes for all aspects of the entire education industry. Of course, the design is usually done according to modules, such as student-related modules, teacher-related modules, exam-related modules, etc.

These modules, functions and processes are either complex or simple, needless to say, when designing complex functional modules, the strong logical connection between modules and the high data dependency is a big headache in the process of process combing and the whole design process.

So how to achieve a simple and elegant design by combing and analyzing at the beginning, such a problem is in front of us.

First, the need to reduce the coupling between modules

In addition to the complexity of the complex functional module itself, the most important thing is that it is built on the basis of the process of other modules, and the output of the module process itself is the premise of the perfection of another module.

Such an intertwined module relationship makes the whole system complex and has a strong degree of coupling, and when doing functional extensions, one hair will move the whole body. Reducing the coupling between modules is a very important thing, and its advantages are as follows.

(a) The lower the coupling between modules, the less they affect each other and the lower the probability of a chain reaction after an exception occurs.

when modifying a module, a low-coupling system can keep the scope of the modification to the minimum possible.

When maintenance is performed on a module, the normal operation of the internal programs of other modules will not be affected to a large extent.

So, how to do to reduce the coupling between system modules?

The author believes that we can start from the following:

Second, the method of reducing coupling

In software engineering, reduce coupling, also known as "decoupling", there is a dependency between modules there is bound to be coupling, theoretically absolute zero coupling is impossible to do, but through certain methods to minimize the coupling B-end system in the design of decoupling methods from the logic, research and development.

As a product manager, the product line development law, management differentiation, system platform support aspects of the logical sorting. Of course, the technical aspects should also be modular microservices, so that the five dimensions of the customer, contract, payment, service, service evaluation of data closed loop.

Data transfer between modules and logical judgment within modules. Developers should try to use data coupling, use less control coupling, limit the use of public coupling, while resolutely avoiding the use of content coupling.

To get back to the point, how can product managers reduce the coupling between application modules at the beginning of the design, which is the focus of this article. It is not easy to achieve optimal decoupling at the beginning, it is a gradual and detailed process from chaos to clarity, I personally think we can start from the following aspects.

1. clarify and understand the requirements, understand the existing system

1) Business flow diagram

B-side is mainly business flow, business flow diagram is the first thing that needs to be clear, and it needs to be general enough and abstract, so that the business process can be divided by different roles, what each role needs to do in each link under what circumstances, what authority, what modules and data are involved, etc., so that the role authority can also be defined.

2) Process state sorting

It is a different business flow, which states, what are the conditions for switching between each state, how many pathways, the need to simulate the data of various situations to go through.

3) System architecture diagram

The overall system architecture is a public ensemble composed of different processes and different states. When designing the system architecture diagram, it is necessary to clarify the front-end business, the underlying system, each system module involved, to clarify the business scope, and if changes involve other systems, it is necessary to sort out and extract the commonality.

Through these steps, the business flow is transformed into an abstract system architecture diagram, which is of course the 1.0 version of the product's own combing, but also requires interviews with all relevant parties affected by the whole system.

2. Interviews with upstream and downstream system related parties, and update the relevant flow chart and architecture chart after comprehensive analysis based on the interview results

Existing system R & D personnel: what the system can achieve what can not be achieved, certainly look at the code will know, combing the relevant logic can be divided into topics and R & D personnel for discussion, must control not to go off topic.

Familiar with the relevant business product or project manager: familiar with the history of lessons learned and organizational process assets can often reduce the risk of subsequent projects, which are designed so, which are legacy issues, the relevant product colleagues will more or less know some.

Operations, end-users and testers: Operations staff and end-users are the most relevant parties using the system, and operations and testers can often make many excellent suggestions for optimization, which is beneficial to product colleagues to sort out.

3. Optimization with marketing and technical indicators, although this is a late thing

Many times the business volume or user volume of the system will reach a certain level of bottlenecks, marketing colleagues in sales will certainly have relevant data indicators, product design, module decoupling can often be considered together with these needs:.

Technical optimization: the degree of tolerance of business delays, the length of the process, the data archiving cycle, etc.

Marketing indicators: lead and business opportunity acquisition cycle, user experience, etc.

III. Specific implementation steps

When it comes to specific operations, we can use the following ideas.

1. Sorting out the atomic services

Combing the modules required by the system that carries the business, then splitting these modules into one function point, merging similar function points into an independent service, providing a separate capability, merging services to ensure that each service can provide services to the public separately and the capabilities provided by each service do not overlap, this is called atomic services.

How well this step is done depends on.

Your ability to understand the business, if it is from 0-1 you can refer to other education industry friendly competitors, if not you can refer to existing systems.

your thinking about the development of the industry in which the business is located, this largely determines the expandability and rationality of your service, this can be combined with the history of industry development and your own knowledge of cognition, more abstract.

Boundaries must be clear, to clarify the boundaries of the functions done and the limitations of the technology in reality, both the sky is the limit, but also to clarify the system boundaries.

2. find out the common items inside these services, make the bottom configuration rules, and then let his services to call

For example: facilities and equipment management and related data collection, often involving multiple hardware, multiple into the language, then each module of the framework language package is a common item, the purpose of doing so is to ensure the consistency of data across dimensions.

3. sometimes a business closed loop actually requires multiple services together to provide services, which requires another layer of service aggregation to ensure the independence of atomic services

In short, we should generally adopt a process of: sorting out independent atomic services (to ensure independence without coupling) - identifying common underlying configuration items (to ensure dimensional data unity) - aggregating services to support the business (to ensure decoupling of atomic services).

IV. Summary

Adequately reduce the coupling of each module is a complex education system must be solved, the decoupling of the system should be structured from both logic, research and development. The development rules of the product line, management differentiation, system platform support aspects of the logical sorting. The technical aspect should be modularized microservices.

This will be of great help to the later expansion of product functions and the evolution of the system.