**演讲稿：**

Good evening everyone, now I will report to you on behalf of the 12th group.

Last class we studied Relation-Oriented Architecture. Relation-oriented software architecture is a high level of abstraction of a system, a computable software architecture, which is the theoretical basis for realizing software architecting and re-architecting, especially in the era of the Software Defined and the Internet of Things.

ROA has the following five characteristics:

①A system view from another corner;

②Higher level abstraction;

③Commonalities of the existing three;

④A computable artifact;

⑤Enabling transformation between different architectural styles;

Additionally, Software Re-architecting is a process to reorganize the organizational structure of the software into a more suitable one.

Physical Re-architecting is a process to reorganize the organizational structure of the software into a more suitable one, without changing the functionality and externally visible behaviors of the software.

Logical Re-architecting is a process to reorganize the organizational structure of the software into a more suitable one, by changing neither functionality , externally visible behaviors nor source code.

On the question of Relation-oriented software architecture, we find perspectives from other scientists.

**Software refactoring can often do the following:**

1. Optimize software structure: refactoring emphasizes optimizing software structure and supporting multiple ways of expansion, while the actual implementation only provides the most matching solution for the task, and only triggers the extension implementation when new requirements arrive.

2. Enhance the maintainability of the software: There will be thousands of duplicate codes in the program, and one of the goals of refactoring is to eliminate duplication, strive to make the same code appear only once, and make the code easier to maintain without changing the efficiency of the software.

3. Enhance the comprehensibility of code: Refactor classes with single functions and functions with single responsibilities, reduce local variables, reduce system coupling, and design meaningful naming rules so that different developers' code can be understood consistently.

4. Simplify testing: Refactoring emphasizes clarity of code structure, requires "small steps", and rigorous testing of each step, which will make the problem easier to highlight and the test simpler.

In the era of cloud computing and big data, we are facing the challenges of software systems that are becoming larger and more complex, so that the software maintenance crisis continues to increase. To this end, this report proposes a Relationship-Oriented Architecture (ROA). Traditional architectures such as function-oriented architecture (FOA), object-oriented architecture (OOA), and service-oriented architecture (SOA) are oriented towards software components (or services), while ROA is oriented towards the relationships between these software components (or services). Compared to the traditional architecture described above, ROA provides a higher level of abstraction and makes the software architecture computable objects. This is important for automating the refactoring of large and complex systems and thus effectively improving the maintainability of the system.

**Part 3**

Relationship-oriented Architecture is an indispensable technology in today's Internet of Things era. As a computable software Architecture, Relation-Oriented architecture is an important theoretical basis for realizing software architecture and rearchitecture.We believe that as software design becomes more and more complex and individual projects require more and more cooperation, Relationship-oriented Architecture can simplify code and improve the maintainability of code without affecting the software structure. To sum up, Relationship-oriented Architecture is of great importance and is one of the important technologies to be mastered. And we believe that in the future, there will be more improvements to ROA and more substantial help for software design.

**PPT:**

**What is Relation-oriented software architecture?**

Relation-oriented software architecture is a high level of abstraction of a system, a computable software architecture, which is the theoretical basis for realizing software architecting and re-architecting, especially in the era of the Software Defined and the Internet of Things.

**Properties of ROA**

①A system view from another corner

②Higher level abstraction

③Commonalities of the existing three.

④A computable artifact.

⑤Enabling transformation between different architectural styles

**Software Re-architecting**

*Software Re-architecting is a process to reorganize the organizational structure of the software into a more suitable one.*

Physical Re-architecting is a process to reorganize the organizational structure of the software into a more suitable one, without changing the functionality and externally visible behaviors of the software.

Logical Re-architecting is a process to reorganize the organizational structure of the software into a more suitable one, by changing neither functionality , externally visible behaviors nor source code.

**Software refactoring can often do the following:**

1. Optimize software structure: refactoring emphasizes optimizing software structure and supporting multiple ways of expansion, while the actual implementation only provides the most matching solution for the task, and only triggers the extension implementation when new requirements arrive.

2. Enhance the maintainability of the software: There will be thousands of duplicate codes in the program, and one of the goals of refactoring is to eliminate duplication, strive to make the same code appear only once, and make the code easier to maintain without changing the efficiency of the software.

3. Enhance the comprehensibility of code: Refactor classes with single functions and functions with single responsibilities, reduce local variables, reduce system coupling, and design meaningful naming rules so that different developers' code can be understood consistently.

4. Simplify testing: Refactoring emphasizes clarity of code structure, requires "small steps", and rigorous testing of each step, which will make the problem easier to highlight and the test simpler.

1. ZHOU Longliang,YU Xiang,TANG Xuejun. Information Recording Materials,2020,21(11):214-215.DOI:10.16009/j.cnki.cn13-1295/tq.2020.11.140.

In the era of cloud computing and big data, we are facing the challenges of software systems that are becoming larger and more complex, so that the software maintenance crisis continues to increase. To this end, this report proposes a Relationship-Oriented Architecture (ROA). Traditional architectures such as function-oriented architecture (FOA), object-oriented architecture (OOA), and service-oriented architecture (SOA) are oriented towards software components (or services), while ROA is oriented towards the relationships between these software components (or services). Compared to the traditional architecture described above, ROA provides a higher level of abstraction and makes the software architecture computable objects. This is important for automating the refactoring of large and complex systems and thus effectively improving the maintainability of the system.

1. [Academic reports]Relation-Oriented Architecture – Li Haikuan

**Our view**

Relationship-oriented Architecture is an indispensable technology in today's Internet of Things era. As a computable software Architecture, Relation-Oriented architecture is an important theoretical basis for realizing software architecture and rearchitecture.We believe that as software design becomes more and more complex and individual projects require more and more cooperation, Relationship-oriented Architecture can simplify code and improve the maintainability of code without affecting the software structure. To sum up, Relationship-oriented Architecture is of great importance and is one of the important technologies to be mastered. And we believe that in the future, there will be more improvements to ROA and more substantial help for software design.