Is OO Working As It Promised?

Sahng-Mann Han
Software R&D Laboratory
Hyundai Electronics Co., Ltd.
598-6, Shinsa-Dong, Kangnam-Gu, Seoul, Korea
smhan@hei.co.kr

To discuss whether or not OO (Object-Oriented) paradigm for the software development is working as it promised, we need to review what the OO software development paradigm promised. Users in different communities introduced different advantages of using OO for different applications. Those differences could be reached to the consensus of the following advantages.

The OO concepts are well suited to problems using real world concepts rather than computer concepts. This means that OO can give better understanding of requirements for the target software applications we want to develop. The better understanding of requirements can give rise to the cleaner analysis of problems and the cleaner design to solve those problems. Operator polymorphism as one important feature of OO paradigm provides an easier software maintenance. The caller of an operation of a class need not concern about how many implementations of a given operation exist and need not be modified when a class is added. OO paradigm also provides an environment in which designs and code are reusable on future projects, because its features such as abstraction, encapsulation, and inheritance, can make libraries of reusable components to be built. Reusability can make us able to reduce the code size and the software development time.

Let’s look at what the real problems of using OO development paradigm are in practical sense. People in OO community, especially OO gurus, have shown that OO technology have several benefits to software developments including the benefits mentioned above. However, they didn’t show how difficult a sudden transition from one technique to another was, especially in business organizations.

First of all, we have to change the existing development culture. This should entail commitment of the top level management. Our experience confirms that this is the hardest job, since we have to convince the top level management that OO development paradigm is of great benefit to the time and cost reduction. One way to show the top management that the new development technique is beneficial is to refer them to several sites where the development target and environment are similar to ours and the new technique has been successfully applied. Unfortunately, the number of such successful reference sites are not enough in the current business applications.

Further, we may train software developers, even customers, before we launch a new project and we may assign OO gurus to monitor the full development phases of the project. In the current business environment, it is not common for companies to own OO gurus to train developers and to monitor the development phases. In this case, we need the external consultation and this might entail an increase of the project cost and difficulties of persuading the top level management.

Another problem is that we may concern about the legacy systems. To deal with the preexisting non-OO software systems, we have to reverse-engineer them or have to provide an interface between the legacy systems and our intended OO systems. Currently, not enough solutions are provided.

Finally, one major advantage of OO paradigm, reusability, is a well-known hard job that we should overcome. To get that benefit we should carefully prepare the design and implementation reuse plan. It might increase the software development cost.

We believe that more obvious techniques and methods are needed to make the transition (from old software development paradigms to a new one called OO software development paradigm) easy and feasible. Unless those problems are resolved, it will be very difficult to plant the promised OO development paradigm in our current and future development environments.