

# Evaluation of Object-Orientation for Industrial Usage

A. K. Onoma

Hitachi Software Engineering Co., Ltd., Yokohama 231-8475, Japan

Email:ako@aqu.hitachi-sk.co.jp

## Abstract

We have a few number of pilot projects in Object-Oriented Development (OOD) for small scale systems whose sizes are around 40 to 80 thousand lines of code in C++. We believe OOD should contribute in the software productivity improvement. But we found some issues in OOD which should be more improved for industrial environment.

## 1 OOD Pilot Projects

We are trying to introduce OOD for user program development in an industrial environment. Under the following conditions, we had six pilot projects in which we applied OOD and Use Case [3, 2]:

- 1) They were actual contract projects. Requested quality level, specification changes and delivered documents were almost same as the traditional case.
- 2) The development project had been done as the first experienced project, that is they were developed from the first step without any re-use.
- 3) The engineers took a off-the-job-training course and are familiar with OOD in the same level as with waterfall model [6].

Two of them were failed. The first project, we abandoned to apply OOD in object analysis phase and we completed by traditional waterfall model. The third project, we had trouble in project progress control and delayed for the delivery. Other four projects we had small troubles as usual, but we can say they were not so in fail.

Figure 1 shows the manpower distribution for each phase which depends upon the work-log of each engineer for every week. We added a slight theoretical correction on it.

Through these pilot projects, we found OOD should contribute to software productivity improvement, but the amount is not so large, 10% around, without any re-use.

## 2 Issues in OOD

The merit of OOD will vary with skill level of the engineers in the applications domain and it will be more increased with ratio of the re-use. The followings show

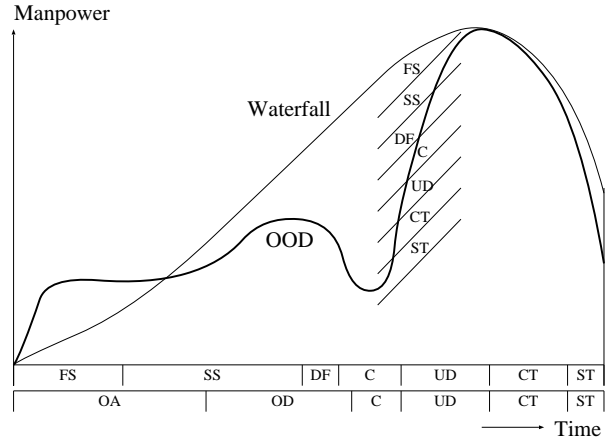


Figure 1: Manpower Distribution for OOD and Waterfall model. Upper and lower x axis is for Waterfall and for OOD respectively. FS: Functional Spec., SS: Software Spec., DF: Detailed Flowchart, C: Coding, UD: Unit debug, CT: Combination test, ST: System test, OA: Object Analysis, OD: Object design.

some issues in OOD which should be solved for industrial environment:

### 2.1 Requirements Specification

In typical projects we receive many modification requests at later phases such as system testing. And this situation remained the same on these pilot projects. In our user program development, we prepare the development contract according to their requirements specification, and we are supposed to deliver a perfect product in as good quality as specified.

However, fixing requirements is very difficult, and most likely, they will submit many modification requests at system testing phase, where the users can actually see the output, which is one of the highest concerns to them. Even if we apply incremental developments, these issues will not be solved.

### 2.2 Design Pattern

We tried to use “design patterns” [1], but we could not enjoy the effectiveness from them so much. They may be effective for re-use and they may be mandatory for *application architecture* of reusable software design in OOD. Design patterns themselves should not be a direct

mandatory tool, but a basic knowledge for the engineers.

We installed and utilized TDC [7] which contains around 2,500 algorithms such as sorting, searching and break point processing for print, including coding fragments. They are effective to increase productivity of program development due to the fact that they will compose of the fundamental *software structure* and so the project manager is released from checking of the program logics in detail.

### 2.3 Documentation

In actual design of these projects, we used "Rose" [5] as documentation tools and rewrote deliverable documents for users from its outcomes such as sequence chart, collaboration diagram, etc. because our users would like to have the traditional documents.

### 2.4 Quality Assurance

Even in OOD, program testing and quality assurance is mandatory, and one of the most important issues. We have applied regression testing for them.

If the total number of PCL for the system is not fixed, using Gompertz curve [4] it is eventually impossible to estimate the quality and to evaluate the test completion date and the delivery date as well. Those estimations were done by the sixth sense or confidence of the project manager in these projects. These managing figures became worse and returned to the ones at the date of non-Gompertz. These mean that quality in the field may be worse than before.

We have gathered and maintained bug statics (bug phenomena and the cause) for preventive quality assurance. For OOD these old database seems to be no longer applicable. This means OOD is perfectly different from the waterfall model.

### 2.5 Development Progress Control

For these projects, we used Guntts chart for the progress management due to the reason why we can not fix the number of design cycle.

We usually used to use PERT diagram. PERT diagram is very cost effective for a large project, but not so good for a small one such as a few thousands lines of code projects. Guntts chart is not so cost effective, from which it is very difficult to find out the critical path.

## 3 Conclusion

We had pilot projects to introduce OOD into an industrial development environment after off-the-job-training for the engineers and get not-failure-result. But we found we need more improved OOD technologies in the following points:

- 1) Software development should not be restricted to be personnel oriented, but it should be done by *engineering oriented*. The amount of the buck log

should not be reduced if OOD is in personnel oriented methodology. We need a better *metrics* from managerial points of view also.

- 2) *CASE tools* should be actualized especially for documentation and testing.
- 3) *Progress management tools* are also necessary. Graphical chart is necessary for progress management. DELTA chart may be one of candidates which should be able to service on the number of design cycles.
- 4) *Design pattern, framework, class library and idioms* is also mandatory. In user program development, these technical terms themselves are different from each others, and they have often same meaning. To use practically and effectively *design patterns*, we might have to use new technology such as machine translation.

They should be released and open to anybody. All the users should pay for these design references. For these we need intelligent rights and easy control system from win-win game points of view.

- 5) Formal specification language should be defined, maybe an "Application Compiler". These outputs may be utilized for quality assurance, especially for testing, program generation and progress management. Formal specification language will contribute very much to increase productivity.

## References

- [1] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA., 1994.
- [2] I. Jacobson, M. Christerson, P. Jonsson, and G. Oevergaard. *Object-Oriented Software Engineering — A Use Case Driven Approach*. Addison-Wesley, Reading, MA., 1992.
- [3] A. K. Onoma, M. Komuro, H. Suganuma, A. Kumeta, and T. Syomura. Management of object oriented development based on ranked use cases. In *Proc. of COMP-SAC97, Washington, DC., August 1997*, pages 246–251, 1997.
- [4] A. K. Onoma and T. Yamaura. Practical steps towards quality development. *IEEE Software*, 12(5):68–79, May 1995.
- [5] Rational Software. *Rational Rose — Release 4.0*. Rational Software, Santa Clara, CA., 1996.
- [6] W. W. Royce. Managing the development of large software systems: concepts and techniques. In *Proc. of WESCON70*, 1970. Also available in *Proc. of ICSE87*, March 30-April 2, pp.328–338, 1987.
- [7] D. Tajima and T. Matsubara. Inside the Japanese software industry. *IEEE Computer*, 17(3):34–43, March 1984.