NASA-CR-192819

Advanced Software Development Workstation

Comparison of two Object-Oriented Development Methodologies

GRANT
IN-61-CR
157387
P.8

Michel Izygon

Barrios Technology, Inc.

December 31, 1992

(NASA-CR-192819) ADVANCED SOFTWARE DEVELOPMENT WORKSTATION. COMPARISON OF TWO OBJECT-ORIENTED DEVELOPMENT METHODOLOGIES (Research Inst. for Computing and Information Systems) 8 p

N93-22042

Unclas

G3/61 0157387

Cooperative Agreement NCC 9-16
Research Activity No. SR.02

NASA Johnson Space Center
Information Systems Directorate
Technology Development Division

ricis

Research Institute for Computing and Information Systems
University of Houston-Clear Lake

INTERIM REPORT

The RICIS Concept

The University of Houston-Clear Lake established the Research Institute for Computing and Information Systems (RICIS) in 1986 to encourage the NASA Johnson Space Center (JSC) and local industry to actively support research in the computing and information sciences. As part of this endeavor, UHCL proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a continuing cooperative agreement with UHCL beginning in May 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The UHCL/RICIS mission is to conduct, coordinate, and disseminate research and professional level education in computing and information systems to serve the needs of the government, industry, community and academia. RICIS combines resources of UHCL and its gateway affiliates to research and develop materials, prototypes and publications on topics of mutual interest to its sponsors and researchers. Within UHCL, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business and Public Administration, Education, Human Sciences and Humanities, and Natural and Applied Sciences. RICIS also collaborates with industry in a companion program. This program is focused on serving the research and advanced development needs of industry.

Moreover, UHCL established relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research. For example, UHCL has entered into a special partnership with Texas A&M University to help oversee RICIS research and education programs, while other research organizations are involved via the "gateway" concept.

A major role of RICIS then is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. RICIS, working jointly with its sponsors, advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research and integrates technical results into the goals of UHCL, NASA/JSC and industry.

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Dr. Michel Izygon of Barrios Technology, Inc. Dr. Rodney L. Bown served as the RICIS research coordinator.

Funding was provided by the Information Systems Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA research coordinator for this activity was Ernest M. Fridge III, Deputy Chief of the Software Technology Branch, Technology Development Division, Information Systems Directorate, NASA/JSC.

Advanced Software Development Workstation

Comparison of two Object-Oriented Development Methodologies
Interim Report

Prepared for
NASA-Johnson Space Center

December 31, 1992

Submitted by
Dr. Michel Izygon
Barrios Technology Inc.
1331 Gemini Av.
Houston, TEXAS 77058

ABSTRACT

This report is an attempt to clarify some of the concerns raised about the OMT method, specifically that OMT is weaker than the Booch method in a few key areas. This interim report specifically addresses the following issues:

- Is OMT Object-Oriented or only data-driven?
- Can OMT be used as a front-end to implementation in C++?
- The inheritance concept in OMT is in contradiction with the "pure and real" inheritance concept found in OO according to B. Meyer)
- Low support for Software Life-Cycle Issues, for project and risk management
- Uselessness of functional modeling for the ROSE project
- Problems with event-driven and simulation systems

The conclusion of this report is that both Booch's method and Rumbaugh's method are good OO methods, each with strengths and weaknesses in different areas of the development process.

Comparison of two Object-Oriented Development Methodologies
Interim Report
RICIS Project

Michel Izygon

This report has been initiated at the request of the ROSE project managers, in order for them to make a decision on the Object-Oriented Method to be used. The ROSE project is a STSOC effort to re-engineer the Flight Analysis and Design System into an OO software. This is an attempt to clarify some of the concerns raised about the OMT method in an internal RSOC note written by Stephen Strom.

First let us summarize the main complaints or concerns rose by Stephen Strom on the OMT method:

- Is OMT Object-Oriented or only data-driven?
- Can OMT be used as a front-end to implementation in C++?
- The inheritance concept in OMT is in contradiction with the "pure and real" inheritance concept (to be found in OO according to B. Meyer)
- Low support for Software Life-Cycle Issues, for project and risk management
- Uselessness of functional modeling for the ROSE project
- Problems with event-driven and simulation systems

In the first part of this report, we will address all these issues by giving our position and by referring to other articles or books. In the second part we will summarize the pros and cons, based on very technical considerations, for using the OMT method as compared to using the Booch method.

1- Is OMT a true Object-Oriented method or a Data-Driven method?

First, as cleverly pointed out by Steve Strom, *whether OMT is really "object-oriented" does not necessarily say anything about whether it is useful.* Second and what is more important, this philosophical type of argument about what is pure OO does not seem the best way to achieve clarity and help in the decision making process. So let us try to separate the tares from the wheat in a short and hopefully non-biased way. What is pure OO? Can I define it? No. Can someone in the community define it? Probably not, the field is in too dynamic a stage at this point for anyone to claim to hold the whole truth. Various attempts have been made to identify the properties of the object-oriented paradigm [Blair et al., 1989], [T. Korson and J. McGregor, 1990], and some more are on their way to developing standards, such as the Object Management Group-Object Model Committee. Even though many differences can be found, the common basic principle of all these definitions is:

An Object serves to group operations with the data they will transform.

The OMT method is definitely consistent with this view. Now let us focus for a moment on the history of the Object-Oriented concept. In the beginning there were Object-Oriented Languages. It appeared to take advantage of the power of such languages, it was necessary to develop specific Design Method adapted to the core paradigm. Later on, and relatively lately (1989), Analysis Methods based on the same concept started to appear because of the perceived gap between Structured Analysis and Object-Oriented Design. The definitions of the respective goals of OOA and OOD as given by Grady Booch [Booch 1991], are the following:

> *In OOA we seek to model the world by identifying the classes and objects that form the vocabulary of the problem domain.*
>
> *In OOD we invent the abstractions and mechanisms that provide the behavior that this model requires.*

Clearly, according to Booch, the Design phase is more "Behavior-Driven" than the Analysis phase. It is therefore not surprising to find that OMT is more Data-Driven than Booch's method, as the first is more geared toward Analysis and the second toward Design. The classification of the methods as Behavior-Driven or Data-driven given by Steve maps quite well to the classification of the methods in Analysis and Design (Meyer, Stroustrup, Coplien and Wirfs-Brock being Design-Oriented are also Behavior-Driven, as Coad-Yourdon, Schlaer-Mellor and Rumbaugh being Analysis-Oriented are more Data-Driven. Nothing here is surprising. A point worth mentioning here is the existence of a strongly Behavior-Driven OOA developed by ParcPlace's Adele Goldberg. In an issue of the ACM concerning an article about the Object Behavior Analysis method, K. Rubin and A. Goldberg write:

> *Object-Oriented Analysis endeavors to model a situation in terms of a collection of interacting entities, each of which provides a well-defined set of behaviors and attributes. Most published approaches describe conceptually similar definitions, although they adopt alternate terminology [reference to Rumbaugh]. There is a high degree of agreement on the desired structure of the result; we differ in how to get to the result.*

Obviously the tenants of Behavior-Driven Analysis consider OMT a real Object-Oriented Method.

Let us summarize this discussion:

- There is no complete uniformity on the concept of Object-Orientation. Today no one can claim that one definition is the best and unique one, nor that there is a pure Object-Oriented concept and that all the others are on a wrong path. The diversity of the concept is a sign of a dynamic and maturing field.

- OO Analysis Methods tend to be more Data-Driven, whereas OO Design Methods tend to be more Behavior-Driven. Even so, OMT deals with object behavior through the Dynamic Model that has the goal of modeling the states of an object life-cycle.

- Behavior-Driven Analysis Methods, such as ParcPlace's (Adele Goldberg), differ from OMT in the fact that in order to find the basic Objects of a domain they first look at the behaviors that the system should exhibit. Once the behaviors are identified, they derive the objects by determining who performs them. Their goal is the same: Finding the objects of the problem domain.

2- Can OMT be used as a front-end to C++?

This is a legitimate question, as we agree that OMT is stronger in Analysis than Booch but weaker on Design. Therefore we expect the gap between the OMT Design and implementation to be larger than with Booch. Given this weakness, should we be concerned when implementing in C++? If we look at Chapter 15 of the OMT book, we can see that Rumbaugh explains very clearly how to implement in C++ each of the concepts that Steve refers to: in 15.2.1 how to define a class in C++, in 15.5.1 how to use inheritance in C++, and he goes on to explain how to use the virtual functions, and in 15.6.1 how to implement associations. Would all that be enough? Probably yes for a designer that understands C++, which means that training in C++ must be provided in addition to the training on the method. That is also true for the Booch method and any other OO method. Maybe it is worth mentioning that projects have had success using OMT for Analysis and Design while implementing the code in C++; one of them is described in the OOPSLA 92 Proceedings, p. 359.

3- The inheritance concept in OMT is in contradiction with the "pure and real" inheritance concept.

For this point, we are back to a philosophical argument. Should an heir class be allowed to remove an inherited method from its parent class? This subject is being discussed in many different articles by the theoretical community (see OOPS Messenger, Vol. 1. No. 1, Aug. 90, pp. 38,39). In his book *Object-Oriented Reuse, Concurrency, and Distribution* [ACM Press and Addison-Wesley: 1991], Dr. Colin Atkinson writes:

> *Using inheritance, a new class, the heir, may be defined as a specialization of another class, the parent, by inheriting its methods and instance variables and adding to them. The heir class therefore usually conforms to its parent, since it normally inherits all the methods in its interface. This need not be the case, however, since many languages allow heirs to remove inherited methods from their interface and to redefine method parameter types.*

Even if we take the stand that inherited methods should not be discarded in an heir class (which is anyway always done for the **create** method as admitted by B. Meyer), the redefinition (by overriding the implementation) of an inherited method takes care of the problem.

4- Low support for Software Life-Cycle Issues, for project and risk management

These subjects are not specifically addressed by Rumbaugh's book. He takes great pains to stress that the process is highly iterative rather than sequential. However, some of Rumbaugh's articles published in JOOP (see specifically the JOOP issue of May 1992) are tackling the problems of the Life-Cycle development process in more detail. Still, it should be noted that the use of OO methods in large real-world applications is only now being generalized and that many lessons will be learned and made public in the near future. As far as the Booch method is concerned, we do not feel that chapter 7 of Booch's book give sufficient explanation of these subjects either. As stated in the summary of our discussion with Booch, he will publish a full book on the subject within a year. Other sources can be found in different articles reporting the experience gained in Object-Oriented development Projects. (OOPSLA 91, 92). The book "Object Lessons: Lessons Learned in Object-Oriented Development Projects" by Tom Love gives some guidelines on OO project management and addresses the risk management issue. Clearly these lessons will be applicable whatever the method used, as project and risk management are orthogonal to the OO methods.

5- Uselessness of functional modeling for the ROSE project

It is not clear to us that functional modeling will or will not be helpful for the ROSE project; however, whether it is useful or not does not mean that the whole OMT method is useless. We do not agree with Steve's interpretation of Rumbaugh's discussion about dynamic simulation, i.e., that functional modeling is of little use. Rumbaugh implies that simulations cannot be properly modeled using <u>only</u> data flows, as some traditional methodologies would attempt to do, thus emphasizing the need for all three views (i.e., object, event, and functional models) and therefore the superior abilities of the Object-Oriented approach over traditional methodologies.

Some lines later (p. 215) he adds: *Simulators often have a complex functional model as well.* For certain projects clearly less emphasis should be given to the functional model. Rumbaugh's next book will de-emphasize the graphical notation and replace it (when necessary) with a PDL type of process description. In any case, learning the functional modeling part of the OMT takes only a short time, and it can be shortened even more if needed.

6- Problems with event-driven and simulation systems

We feel that the critiques on the event-driven and simulation systems made by Steve are not justified. All that Rumbaugh is doing in the discussion on software control is to explain what the most common types of control mechanisms are. In doing so, he describes the event-driven sequential implementation that is the most widely used today in the UNIX workstation world, i.e., X-Windows. As this has become a de facto standard, we believe that it is a constructive and needed introduction to the concept. In any case, it does not imply that the OMT method could not be used with an Object-Oriented event-driven system. In fact, much of Rumbaugh's description of the concurrent system control mechanism could then be used. Finally, it is worth mentioning that a local IBM team has been working on building a generic Flexible Simulation Environment using OMT and claims to be very successful in applying the method to a problem that is fairly close to the ROSE project [SIMTEC 92]. If needed, more information can be provided on this subject. As for the question regarding whether a simulation is generally driven by a timing loop at a fine time scale or is event-driven itself, we have had experience with both types of simulation systems.

Pros and Cons of using Booch's method or Rumbaugh's method.

Both methods are valuable and will allow the team to do a good job. There is not a bad choice that will drive to failure. Our choice of Rumbaugh is motivated by the fact that the method is stronger in the Analysis phase, and we believe that a good OOA is important for such a project. It is at the analysis level that commonalties are best captured and that Reuse is best achieved. Furthermore, our experience makes us think that it is an easier method to learn for those people who have had no exposure to Object-Oriented concepts.

On the other hand, if Ada is chosen as the implementation language, Booch's method is better adapted. The analysis phase should, in this case, be carefully handled using some heuristics of OMT, OBA, or of the HP-UK's Fusion method (a new method whose goal is to integrate and extend OMT, Booch and CRC methods, and which will be published in January).

In any case, we believe that the tool that is best for the ROSE project is Paradigm Plus, as it already implements both Rumbaugh's and Booch's methods. It has also added a Project management method to OMT. Furthermore, Paradigm Plus will be supporting Fusion very soon, and it is extensible so that it can support any other method in the future. Choosing another tool that implements only one method might prevent us from adapting to a very dynamic methodology situation, and might force us to stay with an outdated method after the Pilot project is completed.