The Moose Project

Kors Bos<K.Bos@nikhef.nl>

NIKHEF, Amsterdam, The Netherlands

The Moose or RD41 project aimed to study the viability of the Object Oriented software approach for reconstruction code for the LHC experiments. Methods, languages, tools, persistence and project management were the areas to be studied. The project has come to its end and many of the ideas have been accepted within the ATLAS and CMS experiment and can be found in the Computing Technical Proposals which were submitted in December last year. This paper summarises our experiences and more detailed information will be given in other papers presented at this conference on the results from various parts of the project (see references in this paper).

Key words: methods; tools; languages; process; ATLAS; CMS; OO

1 Introduction

With the long time scale of the LHC experiments it seemed appropriate to study new ways of software development. At the time this project was initiated by some people within the ATLAS collaboration object orientation was becoming increasingly popular among professional software developers. The most attractive promise to us was the possibility for a high level of encapsulation which would make independent software development by dispersed groups of people easier. The maintainability of the software should also be improved by encapsulation which is important for a project with a lifetime longer than the time that most of the developers will be attached to the experiment.

Some initial studies were carried out within ATLAS but when it became an official CERN Research and Development project we were joined by people from CMS. In the first one to two years some common prototype development was done. After that different choices were made within the ATLAS and CMS subgroups and the interests of the two subgroups parted. The Computing Technical Proposals for the experiments effectively brought the R&D project

Preprint submitted to Elsevier Preprint *24 April 1997*

to an end. Most of the experiences acquired and many of the ideas developed in RD41 have been incorporated in the two Computing Technical Proposals [1][2] which can be seen as a success of the project.

2 Areas of Research

2.1 Methods

Several design methods are available and mostly differ in the emphasis which is put on different aspects of the analysis and the design process. Some of them were studied but we mostly used ideas from the "Object Modelling Technique", (OMT) by James Rumbaugh and the Booch method, named after the author of the "Object Oriented Analysis and Design" book. Some other methods like Shlaer-Mellor method were looked at and the Class-Relation model was used for some time when we used a CASE tool which was based on this method. We particularly like various aspects of the what were called second generation methods like Syntropy and Fusion which propose a more rigorous approach than the sometimes loosely defined features in the older methods. Our approach for methods was much like many projects do, we picked what we liked from the various methods. The CMS prototype tracker reconstruction code however followed very closely the Booch method and the design and the code were developed using the tool from Rational Rose [9] which fully supports it.

Recently the authors of the leading methods Booch, Rumbaugh and Jacobson have been working together to define the Unified Modelling Language, (UML) which can be used by all design methods. It is very likely that the UML will become the standard notation. The design tools which are in use now by ATLAS and CMS (see below) will both support UML. It must be noted that the UML primarily defines the notation and not a method and one will probably choose a subset corresponding best to one's development process.

2.2 Languages

Initially, Eiffel was chosen for implementation of the ATLAS reconstruction prototype. It became soon clear however that Eiffel did not get the tools support needed for projects of the size of ATLAS and CMS. There is no realistic alternative at present to C++, in spite of its well known deficiencies. Much of the world-wide development in OO currently is based on C++ and tools of all kinds are mostly only available for that language.

We have looked at some other OO languages but all face the same problem as Eiffel of lacking sufficient tool support, the only exception being Java. We have started looking at Java and were pleased to see that it is a much cleaner language than C++ because it was designed as an OO language from the start which C++ wasn't. In the present design for the ATLAS reconstruction software it is considered that the control part will be implemented in Java. This is much on our current line of thinking of applying the ideas of components in our software.

2.3 Tools

Computer Aided Software Engineering (CASE) tools can help in the design process. They provide the possibility to produce uniform designs for all parts of the software and to document the designs and to make them more understandable. One can generate code (class headers) and documentation from them or reverse engineer existing code into these tools thereby providing a permanent link between the design documents and the corresponding code.

Several CASE tools have been used during the lifetime of the project and one such tool was developed in-house specifically for use with Eiffel. Serious prototyping (more than a hundred classes) has been done with Objecteering, Rational Rose [9] and Software through Pictures, (StP)[8]. The last two have now been selected by CMS and ATLAS respectively.

Rather than listing all the good and bad things about CASE tools we have used our experience to formulate a list of requirements for such tools. These and more requirements for a whole suite of tools needed for software development have been produced within ATLAS [3] [4]. Among the most important requirements for CASE tools is that it supports the method of our choice and that it gives the possibility to be tailored to our needs.

CASE tools alone are not sufficient to support a software development project. In the requirements document mentioned above all tools have been listed that we could think of at that time and in the ATLAS CTP some initial choices have been described for some of these tools.

2.4 Management

One of the important lessons we learned after building one or two prototypes for reconstruction code is that tools alone are not sufficient for proper software development. If the problem domain is not properly decomposed it is difficult to keep clean interfaces between the domains, or categories as they are called by Booch. Even if the code is developed and maintained within the CASE tool this doesn't guarantee a better understanding of the model by other people. Moreover, the CASE tool itself is not the proper way to present the design and code to non-experts. A well defined software process is needed to which the tools are the instrumentation.

During the last year of the project a well specified software process was defined [5] [6] within the ATLAS group. Within CMS not only the Booch notation was followed but also the method to the point where it also prescribes a software development process. The way the Booch's ideas will be used for the CMS software process are presented in [7]. The software process as defined within the Moose project has been adopted by the ATLAS collaboration and is included in the ATLAS CTP[1].

3 Prototype projects

After some initial, rather trivial projects an inner tracker reconstruction program was written in Eiffel for the ATLAS detector and in C++ for CMS. For the first a the Class Relation method was followed and the Objecteering CASE tool was used which supports this method. For the CMS prototype the Booch method was followed and the design and coding was done in the Rational Rose [9] tool. Both prototypes were made to work and are extensively described in the Moose status report [10].

Some time had to be spend to move from Eiffel to C++ for ATLAS. Although it is often stated that the choice of the language is not the most important one as long as it is an OO language, the change from one language to the other took more time than we had anticipated. It's not only the language itself, but the whole environment, the tools, the code management, the documentation, the training, the culture, which have to be changed. Not only did we drop Eiffel but also the occasion was used to change method and CASE tool. The Class Relation method is not very well known and uses slightly different notations and rules than the more usual methods like OMT and Booch. The Objecteering CASE tool relies on this method and therefore had to be changed for something else as well. Our choice for another method was driven by the wish to get into the mainstream of the developments in the software industry. OMT was chosen with the use of Object Interaction Graphs from Booch. The choice of the StP [8] was driven, first, by the requirement that it had to support OMT and Booch and second that it had to be customisable. The choice seems to have been the right one seen from the fact that it is also used in the ATLAS Trigger/Daq group (see [4]) where it is customised to generate what they need. In the reconstruction group the tool is customised to produce the deliverables as defined by our software process.

In the CMS group there was no change-over in method or tool. During the second half of the Moose project the prototype described in [10] was further developed and made more complete. The results of this work together with the plan how this experience will now be applied for the reconstruction software development in CMS will be presented in [11].

For ATLAS the ambitious choice was made to start designing the whole of the ATLAS reconstruction framework. For this a first global decomposition of the whole problem has been made and some domain teams have been set up. These domain teams now have to work on their specifications and make a design. Some teams are more advanced than others and already have a design and sometimes also code. The plan is to have a complete (but with limited functionality) framework designed and coded by the end of 1997. This is now no longer a Moose but rather an ATLAS project.

Moreover a simple but complete framework has been set up called ARVE (for ATLAS Reconstruction and Visualisation Environment) [12] in which design and implementation ideas can be tested. In ARVE the design and code has been implemented for parts of the ATLAS inner detector reconstruction as well as the muon reconstruction. Moreover, the ideas and the code from the visualisation team have been implemented.

4 Conclusions

- The primary conclusion of the Moose project is that the object oriented approach is well suited for the development of software for the LHC experiments.
- As for the methods, languages and tools it is best to choose for the mainstream. Today this is probably a method which is consistent with the UML, and for the languages is C++ and Java. Mainstream components are generally cheaper, more easily obtainable, better maintained and have better interfaces to other tools.
- Object oriented development with the use of tools alone is not sufficient to arrive at the situation required for the LHC experiments with their long lifetime, fast turnover of collaborators and development teams in far apart places. A well defined software process is needed. And yet, a software process is not enough if it is not followed by the developers. Therefore full loyalty of all participants in the process and the management is essential.
- Building prototypes is essential if one wants to start developing object oriented software. Not only to get experience in design and implementation but also to make one realise what all is needed for an OO software development project. A simple framework is needed at any stage of the project to test design and implementation ideas with real code.
- Software development and more generally computing is an exciting disci-

pline and working in this rapidly moving field gives much pleasure. This makes the field of computing in high energy physics attractive for students who find it easy afterwards to get jobs.

References

[1] *ATLAS Computing Technical Proposal*, CERN/LHCC/96-43 [1]

[2] *CMS Computing Technical Proposal*, CERN/LHCC/96-45

[3] *ATLAS Software Development Environment User Requirements Document*, ATLAS Software Note 34 [2]

[4] see also Ashruf Patel, *Use of a CASE tool for developing ATLAS DAQ software*, Paper presented at this conference.

[5] S. M. Fisher, K. Bos and R. Candlin, *The Atlas Software Process*, ATLAS Software Note 27 [3]

[6] see also Steve Fisher, *The ATLAS Software Process*, Paper presented at this conference.

[7] see also Martii Pimia, *The CMS Computing Model*, Paper presented at this conference.

[8] *Software through Pictures*, Aonix, California, USA [4]

[9] *Rational Rose/C++*, Rational, California, USA [5]

[10] *Status Report of Moose*, CERN/LHCC 95/60 [6]

[11] see also Vincenzo Innocente, *The CMS Reconstruction and Analysis, an Object Oriented Approach*, Paper presented at this conference.

[12] see also T. H. Burnett, *ARVE, an Onbject Oriented Analysis Framework Design for ATLAS*, Paper presented at this conference.

1 http://atlasinfo.cern.ch/Atlas/GROUPS/SOFTWARE/TDR/html/TDR-1.html
2 http://atlasinfo.cern.ch/Atlas/documentation/notes/SOFTWARE/note34/asde.html
3 http://atlasinfo.cern.ch/Atlas/documentation/notes/SOFTWARE/note27/asp.html
4 http://www.aonix.com/Products/StP/StP_OMT.html
5 http://www.rational.com/pst/products/rosecpp.html
6 http://atlasinfo.cern.ch/Atlas/GROUPS/ATLASOO/moose/status_report/report/report.html