This brief conference report summarizes the first full-scale exposure of a conservative, FORTRAN addicted, programmer-physicist to the all-promising perspectives of object-oriented methods.

The main topics at the conference were:
- O-O programming languages
- O-O software engineering, analysis and design
- O-O software engineering Toolkits
- O-O data bases
- User experiences with any of the above
there were also minor points of interest
- expert systems
- simulation

The first two days were devoted to parallel tutorials of 4 hours each, presented by recognized experts. The last two days were organized in a more traditional way with a few general invited lectures and four parallel sessions, grouped by subject of interest. A commercial exhibition was organized next door to the conference.

The conference was professionally organized (it is quite expensive as well), with TV indication of the talks being presented in each individual room and full respect of the timing, requisite features to be able to zap between parallel sessions. The proceedings of the talks and of the tutorials were available in advance and I have been aware of only one minor change in the schedule.

A slightly disturbing feature, unusual when you are used to scientific conferences, was the intimate mixture of
- academic presentations (pure research or a nice little, partly debugged, tool to give away)
- presentations about internal research results of big firms (products exist but are for internal use only)
- more or less subtle sales talks (in addition to the "Vendor Session" ones), some to sell development tools and some to sell courses about a proprietary method of analysis and design

The general talks and the tutorials (I heard about one exception) were very well presented. The conference talks were quite varied in interest and presentation. The allocation of talks to the various parallel sessions was sometimes misleading and it was hard to make a choice from the title alone without reading the full proceedings.

I will not attempt to summarize the talks I have been attending. The full conference program is appended to this note and the proceedings + some tutorials are available for consultation.

Given my previous almost complete ignorance about O-O programming and Software Engineering I have certainly made many mistakes in my interpretations. Nevertheless I will present in the following a few, hopefully provocative, remarks from a point of view where we would be heavily using those techniques at CERN.
What I believe I have learned (may be not understood)

Programming Languages

Ordered by frequency of quotation at this conference (may not be entirely representative):
- C++, by far the most used for serious stuff
- ADA, claiming very loud to be sort of and O-O language.
- Eiffel, the best one of course, since the conference was organized by its main proponent
- SMALLTALK, new compilers are told to produce "efficient code"
- Objective C
- SIMULA, the elder (1967), still alive and kicking around the polar circle and in ex-socialist countries
  - a few more... there is an Objective FORTRAN for the NeXT, I even heard the name "Objective Basic"

Eiffel is claimed to be well adapted for both the design stage and the programming. It is undergoing a revision to implement concurrency at the language level.

True O-O extensions are being discussed for the next revision of ADA.

Reusability is one of the apparent advantages of O-O programming. It is true that, due to encapsulation, libraries of objects will likely be more reusable than the old "subroutine libraries" that we were used to. However, to a naive spectator, it appears that:
- the principal domains of O-O programming, graphics and user interface, have a much less general scope and are much more hardware or system dependant than computational libraries.
- objects declarations are used at compilation time, so that it is not a priori possible to use an object library made for a language in a program written in another language. Meanwhile, on the contrary, the run-time compatibility between routines written in different "classical" languages is slowly becoming a reality.
- any change in object definition may lead to the recompilation of a lot of code.
- if we pick-up a few objects from many libraries we should be faced with a lot of useless declarations, again with predictable effect on compiling time.
- we will soon be faced with the problems of how to pick-up objects "from the shelf". There are already, public domain or commercial, several libraries of a few hundred objects. I haven't heard of a scheme to classify objects for easier retrieval, but I several times heard the name "Object Librarian" and that was the name of a new profession, not a clever program

Software engineering Methods and Toolkits

O-O programming and O-O analysis and design complementarity must not be taken for granted. It is not necessary to use an O-O language if an O-O method has been used at the design stage and vice-versa. A+D methods are often more related to ADA than to true O-O languages. The "objects" are anyway not at the same level (in large scale A+D an object can be a whole sub-contracted application). Analysis proceeds naturally along a top-down approach, while object technology is rather bottom-up oriented (specially true if one wants to reuse existing objects).

SE has many churches, some claiming to offer the final solution. There are many fashions depending of the country or continent. There is no reason to believe the situation will stabilize before a few years. Heavyweight players are just coming into the arena now. DEC has products in preparation. IBM has just released to developers the first specifications of AD/Cycle, a very ambitious set of methods and products based on a centralized "Repository" which will cover all the life cycle from analysis to maintenance (availability was first foreseen by outside observers for 1995, but they now believe that the first products will appear at the end of 1991).

General A+D SE methods are geared to very large projects (>50 programmers), typical examples are the US and European Space Shuttle projects. They demand a considerable learning investment (e.g. one week course followed by an extended period of on-site access to a guru). In addition a thorough background knowledge of software engineering concepts (Petri nets, Finite State Machine...) is probably required if one wants to go beyond the religious obedience to the recipes. The benefits may not immediately perceived by the individuals and the method may have to be forced upon them from above.

Small A+D Toolkits, which often go down to the code generation level, are geared at enhancing productivity of individuals or small teams for small projects (e.g. a Macintosh application). They compensate the learning overhead by offering powerful tools of immediate use. I believe that they may represent the least painful way to enter the realm of modern programming.
There is no a priori contradiction between the general methods and light-weight toolkits. There was at least one example demonstrated at the exhibition: STOOD is implementing the HOOD method and goes all the way down to generate ADA code.

Even less ambitious and more accessible modern development environment including Language Sensitive Editors, Source Code Analyzers and symbolic debuggers can already be considered as a step forward. By freeing the programmer of tedious tasks they help him to concentrate more on design. NeXTStep user interface is an example of a productivity enhancing environment.

The best things

A very nice presentation of the NeXTStep user interface (the only talk not in the proceedings!).

The following not only seem to be good products but they also benefited from a good presentation:

- AD method: HOOD, OOSD, Yourdon
- Language: Eiffel
- Toolkit: STOOD (TNI)

Recreation

In addition to buzz words and words which are associated with a meaningful concept, O-O technology and software engineering use a lot of vocabulary. A few examples from the HOOD method:

- ODS "Object Description Skeleton", HCS "HOOD Chapter Skeleton"
- HSER/LSER "Highly/Loosely Synchronous Execution Request", also ASER and TOER

A few nice quotations:

- "Either you believe in it or not" about the HOOD method
- "Badly used the method is an overhead, not a help"
- used as an example of "modern" multi-processor configuration "Let us consider three VAXes."
- "Si le Quoi est un objet, le Comment est une sous-classe"
- "There are a few persons able to handle very large projects, but we have not enough of them, so we must use Software Engineering" by a member of the US Air Force

Disclaimer

This document represents only the views of its author. Other CERN participants at the conference were:

- Doris Burckhart
- Jean-Pol Matheys
- Louis Tremblet

They can be contacted for further information.
Documents brought back

The following documents are available on loan, send mail to JPPVZ@CERNVM or call 5421/5439

Tutorials:
- T1: Introduction to Object-Oriented Technology
- T1A: Case studies in Object-Oriented Analysis
- T1B: New Advances in Object-Oriented Analysis
- TC: Introduction to C++
- TD2: Object-Oriented Database Systems
- TE: Developing Quality Software with Eiffel
- TH: HOOD, Hierarchical O-O Design
- TG: Object-Oriented Structured Design
- TM: Managing Object-Oriented Software Engineering
- TO: Problem Solving in Objective C
- TP: Concurrent Object-Oriented Programming
- TS: SIMULA for the working classes
- TU: Object-Oriented Design for User Interfaces

TOOLS 2, the conference proceedings

A few books recommended by the various speakers

Object-Oriented Software Construction
Bertrand Meyer
Prentice Hall, 1988

Object-Oriented Analysis
Peter Coad and Edward Yourdon
Prentice Hall, 1990

Object-Oriented Systems Analysis, Modeling the World in Data
Sally Shlaer & Stephen J. Mellor
Yourdon Press Computing Series, 1988

Object-Oriented Programming, An evolutionary approach
Brad J Cox,
Addison Wesley, 1986

The C++ Programming Language
Bjarne Stroustrup
Addison Wesley, 1986

C++ Primer
Stanley B Lippman,
Addison-Wesley, 1989

Programming in C++
Dewhurst
xxx

Object-Oriented Programming with SIMULA
Bjorn Kirkerud
Addison-Wesley, 1989

An introduction to SIMULA Programming
R.J. Pooley
Blackwell Scientific Publications
INVITED LECTURES

- Object-Oriented Programming and the NeXTStep User Interface
  Jean-Marie Hullot, NeXT Inc. (USA) 15
- Sequential and Concurrent Object-Oriented Programming
  Bertrand Meyer, Interactive Software Engineering (USA) 17
- Ada and The Objects
  Ben Brosou, Alsys (USA) 29

ANALYSIS & DESIGN

- The Sequence Software Set - Design and Development of a Large C++ Program
  J.R. George, Jet Propulsion Laboratory (USA) 35
- Hierarchy Evolution and the Software Lifecycle
  B. Anderson & S. Oussain, Easys U. (UK) 41
- Language Implementation Feedback on Design
  H. Saleh & P. Gautron, Rank Xerox & U. Paris VI (France) 51
- An Application of OSDL
  J. Norgaard, Nokia Telecommunications (Finland) 65
- Object-Oriented Software in Real-Time Systems
  A.M. Jrad, C.C. Lin & T.M. Rehman, AT&T (USA) 77
- An Object-Oriented CASE Tool for an Industrial Environment
  S.E. Bergner, Objective System SF AB (Sweden) 91
- Multiple and Evolutive Representation in the ROME Language
  B. Carré, L. Dekker & J.M. Gelb, U. Lille (France) 101
- Teleological Maintenance of an Object-Oriented Environment
  V. Karakostas & Z. Palaska, UMIST (UK) 111
- A Method for Object-Oriented Programming: The Class-Relationship Method
  P. Desfroy, Softeam (France) 121
- Essentially Objects
  P.D. Sully, Youkon International (UK) 133
- Mecano: a Method for Object-Oriented Software Construction
  X. Girard, L.G.I. Grenoble (France) 145

CONCURRENCY & DISTRIBUTION

- Decentralized Parsing Method Using Communicating Multiple Concurrent Objects
  C. Numaoa, Kento U. & M. Tokoro, Keio U. and Sony Computer Science Lab. (Japan) 159
- From Rock-Bottom to Mesoscale Actors
  A. Senti, U. Montréal & S. Giroux, Zuniq Technologies (Canada) 167
- Concurrency: An Object-Oriented Approach
  D. Carmel, CRIN-CNRS Nancy (France) 183
- Object-Oriented Real-Time Executive
  L. Lamarche & P. Labiche, Objective Systems Quebec (Canada) 199
- An Extensible Distributed Object Monitoring System, EDOMS
  E.J. Burke, T. Domae & G. Johnson, Northrop Research (USA) 213

CONSTRAINTS

- COOL: An Object System with Constraints
  P. Avanzi, A. Perni & F. Ricci, I. per la Ricerca TN (Italy) 221
- Creating 3D Scenes with Constraints
  G. Fente & B. Penche, ENSM St Etienne, & J. Zoller, GAMSAU, E.A. Marseille (France) 229
- Object-Oriented Design for Modelling Parts, Assemblies and Tolerances
  F. Giacomini & T.C. Chang, Purdue U. (USA) 243
- Constraint Propagation in an O-O Environment: A Mechanical CAD Experience
  K. El Daibhan, U. Compiegne (France) 255

DATABASES & PERSISTENCE

- Persistent Graphical Objects in Procol
  F. van Oosterom & C. Lafla, Leiden U. (The Netherlands) 271
- A Practical Approach to Object-Oriented Software Engineering
  J.G.M. van den Goor, Philips Musical (The Netherlands) 285
- Persistency and Reusability for the Effel Language
  P. Lahire, N. Le Thanh & S. Miranda, U. Nice (France) 299
- G-BASE 4: The New Generation of Object-Oriented Database Management Systems
  R. Roffe, P. Anota & E. Bourgain, Grapheal (France) 315
- Designing an Object-Oriented Database and DBMS for a CAD System
  S. Hope, P. Flemming & G. Wolff, Wales U. (UK) 325
- OODMD: an Object-Oriented Database Design Model
  H. Nguyen & J. Thrasher, Millipore Corp. (USA) 335

EXPERT SYSTEMS

- Knowledge Representation Framework for an Expert System in Computer Network Design
  C. Jouye, ENSM St Etienne (France) 349
- An Object-Oriented Knowledge-Based Fault Monitoring System
  D. Porte, Touche Ross Management Consultants (UK) 357
- Ship-Planner: A Conception Support Tool for the Bay Plan Container Ships
  H. Sarsen, Systema (France) 369

PRACTICAL EXPERIENCES

- Genericty, Inheritance and Relations: A Practical Perspective
  G.C. Murphy, Microlab Pacific Research (Canada) 381
- Object-Oriented Geo-Information Processing in Modules
  S. Alagic & Z. Galic, Sarajevo U. (Yugoslavia) 393
- An Object Approach of ODA Implementation for Technical Documents Production
  J.C. Chabrier & J.-M. Guillemin Labone, U. Bourgogne (France) 407
- Strategic Defense Initiative (SDI) Software Reuse Issues
  J.S. Morrison, US Air Force (USA) 419
- Object-Oriented Technology and Reuse in Telecommunication Applications
  A. Simmonds & W. Ye, Lund U. (Sweden) 413