5th Workshop for Doctoral Students in Object–Oriented Systems

Held in Conjunction with ECOOP'95 in Aarhus, Denmark

Günter Kniesel[1] Thilo Kielmann[2] Athanasios Demiris[3] Milena Shteto[4]

Abstract. The 5th ECOOP Workshop for Doctoral Students in Object–Oriented Systems provided a forum for lively discussions between PhD students, with topics ranging from the technical work of each attendant to non–technical issues like writing about one's work. This report describes the structure of "ECOOP PhD workshops", summarizes the topics that have been discussed, and reflects the workshop structure, giving some advice for organizing similar events. It is based on the authors' experiences and on feedback from workshop participants.

1 The ECOOP "PhD Workshops"

The series of "PhD workshops" that started at ECOOP'91 is intended to advance the personal and professional development of PhD candidates working in the field of object–oriented systems. To this end the workshops traditionally consist of a technical and a non–technical part, which together provide a unique opportunity to meet peers, discuss one's research, and further develop working skills. The uniqueness of the PhD workshops is due to the combination of the following characteristics:

- Openness – students who have just started their PhD work as well as advanced PhD students are equally welcome; each attendant may rise an issue that is especially important for his or her work and that will be discussed in a technical session.
- "Talkshop model" – in order to provide participants with as much as possible feedback on their ideas, the technical part always emphasizes discussion, rather then conference–like presentations.
- Meeting of peers – in order to encourage the participants, especially novices, to speak out and present new ideas they are still unsure about, and ask potentially "dumb" questions, all participants of the technical discussions are themselves PhD students.
- Non–technical advice – doing a PhD involves much more then just having great technical ideas. Therefore one part of the workshop is always dedicated to questions like how to organize one's work, how to cope with the information (over)flow, or how to communicate ideas and results (i.e. how to write "good" papers, how to give "good" research talks, or simply how to give a simple but comprehensive answer to the question "What's your thesis?"). The non–technical sessions usually consist of longer presentations (20–30 minutes), followed by sufficient time for discussions. It has become a nice tradition that one of the non–technical presentations is made by an invited speaker, usually ECOOP's conference or programme chairperson.

[1]University of Bonn, Computer Science Institute III, Römerstr. 164, D-53117 Bonn, Germany, Email: gk@cs.uni-bonn.de

[2]University of Siegen, Dept. of Electrical Eng. and Computer Science (FB 12), Hölderlinstr. 3, D-57068 Siegen, Germany, Email: kielmann@informatik.uni-siegen.de

[3]German Cancer Research Center, Dept. 0805 Medical and Biological Informatics, Im Neuenheimer Feld 280, D-69120 Heidelberg, Germany, Email: A.M.Demiris@DKFZ-Heidelberg.de

[4]CNET ISSY, PAA/ATR – piece 030B, 38–40 rue du General Leclerc, 92131 Issy–les–Moulineaux, France, Email: Milena.Shteto@Issy.cnet.fr

- PhDOOS network – continuation of the contacts established at workshops is supported throughout the year by the network of PhD students in object–oriented systems (PhDOOS), which maintains a mailing–list and a WWW server.

During the past years, the high interest and the positive feedback of participants proved the usefulness of this overall structure. However, participants often complained that technical discussions had to be interrupted due to time constraints. Therefore, the 5th workshop was the first one that lasted one and a half days, reserving one full day for the technical part. The topics discussed in the various sessions will be summarized in the next sections.

2 Technical Discussion

To participate in the workshop, attendants had to submit a short abstract of their research work, with an included list of keywords denoting topics to be discussed during the workshop. These were "published" on the WWW server of the PhDOOS network, serving as a first mutual introduction of all participants. On the basis of these abstracts and keywords, the following topics for working groups could be determined in advance:

- OO Basics: Classes, Inheritance, Prototypes, Delegation
- Concurrency: Programming Languages and Models
- Concurrency: Databases, Transactions, Real–Time
- Applications of OO to other domains

The first day, dedicated to the technical discussions, started with a brief welcome meeting and the final assignment of participants to working groups. Thanks to the support of the ECOOP organizing committee, each of the resulting groups of four to seven people had its own meeting room, which provided us with optimal conditions for parallel technical sessions. Each of the groups had almost three hours time for discussion plus 90 minutes time for writing summaries of the discussions. Then we met again in a final, plenary session in which the essence of the summaries was presented to all attendants.

As in the workshops before, a variation of the IPA (issue–position–argument) method was used to get the discussion going and to focus it on **one** relevant issue. Prior to the workshop, each participant was requested to prepare an IPA form, containing his name and a provocative question that could be answered by *yes* or *no*. E.g. someone could ask: "Is there a need for multiple inheritance?" Answers to such question are called positions. Arguments support a position and explain why a position is right. To get a discussion started, the form is usually distributed by each attendant to one of his neighbours. After writing down his position and arguments, the neighbour passes it on to the next member of the group, and so on. Finally, the gathered positions and arguments are used as a basis for the discussions.

However, at previous workshops we experienced that the basic version of the IPA method was not enough to guarantee a fruitful discussion. Often participants lacked enough background information to be able to answer the issue. The time intended for discussion was often wasted on clarification of basic notions. Therefore, we decided to complement the IPA method by a short presentation. Each participant had up to 10 minutes to introduce the ideas behind his issue. Then his IPA forms were distributed to all attendants, who simultaneously made their notes, before starting the discussion. At the end, the forms were returned to the one who raised the issue. This method worked well in three of four groups. Only one group did not use it, since the topics of its members were too heterogeneous for good discussions. In the remainder of this section, the topics and their discussions are reported as completely as possible based on the protocols available to the authors.

WG 1: OO Basics

- Trevor Dobbin asked "Is it useful to perform a preceeding business analysis taking a less structured approach in order to define the context for object–oriented analysis?" Most of the attendants answered *yes*, agreing on the need to bridge the gap between clients and software developers. The idea of having two views of the problem, one from a business perspective and one from a software engineering perspective, was found very useful, especially if it were possible to switch between the two views back and forth. This would give common ground for discusion between the two disciplines. It was further pointed out that the mapping between the two views has two be independent from a particular methodology for object–oriented analysis.

- Peter A. Newson asked "Do we need more dynamic classes?" There was overall consensus on the answer: *yes*. However, it remained open from which direction the problem should be solved: by constraining a dynamic language or by making a static language more flexible. From the point of view of users of static languages, an efficient solution should involve no overhead if the additional features are not used. It was also suggested to make a clearer distinction between notions like *object migration*, *class migration*, and *dynamic inheritance*, which seemed to be mixed up in the presentation. Finally there was a bunch of references to related work in the areas of reflection, subject–oriented programming, and especially object–oriented databases, where related problems are discussed under the keywords *views*, *roles*, *schema evolution*, and *class or object migration*.

- Wolfgang De Meuter presented the essence of his joint paper with Patrick Steyaert, which is contained in the conference proceedings. His topic was the improvement of object–oriented languages by adding controlled modification mechanisms under the form of "*encapsulated* object–based inheritance organized by module mechanisms". He pointed out that unconstrained object–based inheritance inherently breakes encapsulation, since any object, O_1, not satisfied with the client interface of another object, O_2, may become an inheritor of O_2, thus gaining access to its "protected" parts. In order to preserve strong encapsulation, he proposed that the creation of inheritors be under the control of the inherited object. Every object should specify in its client interface a finite number of *mixin methods* that create new inheriting objects with a statically specified structure.

  Whereas Wolfgang's critique of object–based inheritance was acknowledged, some of the participants felt that the proposed cure was too radical, since it required *editing* existing objects in order to add new kinds of inheritors. If editing were not possible, e.g. because the objects are part of a precompiled library, no extensions would be possible at all.

- Referring to Wolfgang's proposal, Günter Kniesel asked "Should extensibility be sacrificed for encapsulation?" He proposed an alternative model which integrates dynamic delegation into typed, class–based systems. Instance variables declared in a class may be declared as "delegation attributes". In addition to having the properties defined in its class, an instance inherits all the properties of the objects referenced by its delegation attributes. In this model, delegation is constrained by the type system, since every object can only delegate to objects of a statically specified type. Thus, encapsulation is respected, since objects may not gain priviledged access to objects that are not of the specified type. However, extensions can be made by adding new classes with respect to new object types, instead of editing existing ones.

  Most of the participants agreed that extensibility is essential and should not be restricted. Having dynamic delegation was found to significantly extend the expressiveness of a strongly typed language, e.g. by capturing notions like dynamically changing *roles* and different *views* of an object without any special purpose language constructs. It was pointed out that different *application* objects representing different views/roles of one *conceptual* object should in all contexts look like *one* object. It was also discussed whether delegation, although constrained by the type system, can still be "tricky" for the programmer.

- Jose Canos Cerda said that "Formal approaches to object–oriented databases allow the use of a unique language to define the schema, query/update the database, and program DB applications." He argued that the use of metaclasses allows to model explicitly every aspect of the system, e.g. the inheritance mechanism, schema evolution strategies, etc. In his work he has used dynamic logic to give a precise semantics to such a system. The participants did not question the generality of the approach. However, they pointed out that it is an open problem how to obtain an effcient implementation that can be useful in realistic databases with heavy demands.

- Erik Ernst suggested that "Inheritance is compression and constraints". *Compression* was described as the ability of inheritance to factorise common parts of various concepts. *Constraints* were used to express typing. In Erik's opinion, type constraints are strongly intervowen with the compression aspect. This view gave rise to a very lively discussion, in which many attendants argued that type constraints are largely orthogonal to the compression aspect of inheritance.

- P. Sergio Almeida talked about "Balloon Types" a type–system/language mechanism to control encapsulation and sharing of state. The addressed problem is the fact that the state of an object implicitly depends on the state of all objects directly or transitively referenced by its instance variables. By aliasing of references, external objects may share and manipulate part of the implicit state of some object. That causes interference which may be unexpected or difficult to analyse.

  The basic idea of balloon types is to make the ability of sharing state part of a type, by way of a (binary) classification of any type in either a *balloon* or a *non–balloon* type. Non–balloon types offer full freedom of sharing, while balloon types provide the invariant that all reachable state (directly or transitively) is not shared by any external object.

  This idea was found interesting by the participants. They suggested to investigate further implications/benefits of knowing that an object has a balloon type and to evaluate "real" programs in order to quantify the benefit of having balloon types.

WG 2: Concurrency: Programming Languages and Models

This group unfortunately consisted of people with very heterogeneous interests and viewpoints, which was found out after mutual introductions into each one's work. So, instead of the usual IPA–guided discussions, the group members decided to split into two subgroups in order to provide usefulness of discussions.

The first subgroup, formed by G. Fouquier and T. Kielmann, discussed opposing views of concurrent, object–oriented programming. First, one tried to clarify the notion of "active object". Today, there are (at least) three schools defining active objects:

(a) in the Actor approach, active objects are seen as *data* + *methods* + *concurrency control* + *mailbox*,

(b) in POOL–like languages, active objects are *data* + *methods* + *body* + *concurrency control* + *mailbox*, and

(c) with Objective Linda (Thilo's thesis work), active objects are simply seen as *data* + *body*.

In the first two approaches, objects are servers, in the last one, objects are peers. But there are still more flavours in models known from the literature.

Second, there was a controverse discussion between integration vs. separation of computation and communication which describes the relations between a programming language and a coordination model. With the integrated approach, concurrency control is integrated into OO. There is OO computation and only one sort of objects; they compute and are synchronized. The separation approach, on the other hand, emphasizes the orthogonality between computation and coordination which results in two sorts of objects. An improved variant of the separation approach is based on a dual view on objects where the first

view deals with OO computation and the second view models the (preferably OO) synchronization and coordination of active objects.

Finally, it could be found out that both opposing views are due to the different worlds in which they originated. The integration approach focusses on possible implementations where target applications are "soft real–time applications" (like multimedia etc.) On the other hand, the separation approach has its origin in interoperatability of heterogeneous worlds where clean system modelling is preferred over simple and/or efficient implementations.

WG 3: Concurrency: Databases, Transactions, Real–Time

One subgroup of WG3 dealt essentially with mobile computing and distributed object–oriented databases.

- Babak Esfandiari presented some aspects of network management standards (GDMO and CMIS), which deal with similar problems and adopted an object–oriented model. He suggested that these standards can be a good source of inspiration for anybody who is interested in distributed data management. Then he asked: "Are non–basic speach acts (such as *Offering*, *Promising*, ...) helpful in decreasing the sum of the exchanged messages in distributed problems?" This question seemed to be a little bit out of context. However, one participant agreed that using high–level communication languages such as KQML, which offers a set of "performatives" (sort of speach acts), can be useful in mobile computing and distributed OO–databases.

- Carlos Baquero asked "Do we need more than one or two transaction commit levels in order to cope with mobile objects and disconnected operation?" The discussion focussed on how transaction models could be adapted to support disconnected operations, in particular in a system that allows communication between mobile hosts. Some of the issues were:

  - How to capture the semantics of object operations?
  - The need to bootstrap recursion on the communication medium.
  - Interest on using other communication models besides message passing, like broadcasting information and asking for cooperation. The language KQML was suggested as a reference to this communication model.
  - How would multiple transaction levels be related to versions?

  In the end, there was no clear answer to the concrete initial question as the discussion drifted into other appealing subjects.

- Giovanna Guerrini asked "Is there a need for views in object–oriented databases?"

  The general answer was *yes*, mostly for analogy with relational databases. Indeed, it seems that the main motivations for introducing views in the relational context (schema reorganization and simplification, content–based authorization, ...) hold as well for object–oriented databases. The applications of views for experimenting schema evolution was also pointed out. Moreover, it emerged from the discussion that no other object–oriented schema concept (e.g. methods for computing derived information, proxies) can fully realize views. Among the main issues of introducing a view mechanism in object–oriented databases, topics of discussion were the generation of new object identifiers for virtual objects and how to keep track of the relationships between virtual objects and the base ones (to propagate changes).

WG 4: Applications of OO to other domains

- Milena Shteto asked "Is a high–level OO language well–adapted to describe algorithms?"

  The general answer was *yes*, because the main point is the configuration of algorithms and this can be definitely modelled in an OO language by a taxonomy of classes.

- Cristian Ionitoiu presented "A class hierarchy for a general CASE tool for distributed system design"

  The intention was to encapsulate the interfacing part and the distribution aspects, such that one can go directly from the GUI to the generated code, e.g. a process knows how to present itself on the screen and how to deal algorithmically with parallelism. The CASE tool uses the process representation and the components library to partially generate the code. One specific feature of his approach was the use of actually two hierarchies: a data hierarchy and an action hierarchy. This was acknowledged by the other participants to be a good modelling approach, as long as the "action hierarchy" is suffciently rich.

- Maurice Amsellem asked "Should the debugging tools in an OO programming environment be unified into a "debugging workspace" that combines all the debugging tasks?"

  - It is a problem to decide between what to hide and what to show.
  - One can think of an "animation context" as a dual pointer to the "debugging context" and introduce some automatic links between the two contexts.
  - In a visual environment like Smalltalk, it is necessary to be able to visualize the dynamic part.
  - Since the visualization of the objects on the user interface takes a lot of time, the visualization of the program should be as short and as easy as possible.
  - Will it be sufficient to have all the needed information to achieve debugging?

- Athanasios M. Demiris asked "Is it good to make patterns fine grained or as general as possible?"

  The question refers to domain specific patterns. When trying to model the knowledge of a certain domain one can extract some patterns out of the way domain experts deal with specific problems. An example coming from the domain of the medical image analysis would be the recognition of a tumour in the liver in a CT scan of the abdomen. Capturing this knowledge, partly in form of a pattern, leads to fine granularity and domain–specific patterns (that do not resemble the patterns found in other more software–engineering oriented pattern collections). The problem of granularity and domain specificity with patterns is similar to the one with classes but since patterns aim solely at reuse it seems to be more important.

3 Non–technical Part

The morning of the second day was devoted to non–technical issues: the network of doctoral students in object–oriented systems (PhDOOS), advices on how to write a PhD thesis and on how to do research in general (invited talk). A retrospective on the workshop concluded the second day.

Athanasios Demiris: "The PhDOOS network and how it could be improved" or: "Words of History, bits of advice"

After the traditional historical notes concerning the evolution of the PhDOOS network, the digital communication medium of the PhDOOS, a presentation of matters of interest took place. Before that, the server's migration to the German Cancer Research Center in Heidelberg was mentioned which could be accomplished thanks to numerous suggestions and the support by Franz Hauck and Thilo Kielmann. The new addresses are:

ftp archive:	`mbi.dkfz-heidelberg.de:/pub/PhD-Network`
mailing list:	`phdoos-request@dkfz-heidelberg.de`
WWW-URL:	`http://MBI.DKFZ-Heidelberg.de/PhDOOS`

The major problem with the PhDOOS Network is the low traffic that characterizes the mailing list. Some possible reasons for this were presented and discussed:

- The lack of time that characterizes the life of almost every PhD candidate. The activities of PhD candidates cover a very broad spectrum.
- The lack of confidence when posting a mail that will be read by about 100 people all over the world. Individual language problems (eloquence in English) and insecurity when asking a thing that might be already known make this problem even worse.
- The lack of meetings. It is often easier to share problems with a person one has talked to and the chance to meet other PhD candidates is limited since ECOOP takes place only once a year.
- The well–known information overflow caused by numerous specialized newsgroups, WWW–pages etc. which sometimes seem more adequate for posting questions. Especially this point seems to be very crucial.
- Underestimating the importance of a PhD–Network? This point summarizes the first and the last reason in a more "aggressive" way.

After the workshop, a short summary of these points was presented in the framework of the mailing list in order to inform those people who could not attend ECOOP. A reaction to this summary emphasized on the existence of other highly specialized mailing lists or newsgroups and introduced the problem with the impression that a PhD network might not be "competent" enough for dealing with such problems.

In order to avoid these problems and achieve a maximum of efficiency for the network a number of suggestions were made:

- Collect all relevant references of books, articles and Web sites for every keyword in the keyword list, e.g. one per month. This way, people might be more motivated to participate. When talking about a subject one is familiar with, people feel more secure and might post more information than usually. (This might overcome the problem of lacking confidence.)
- Create digests of other relevant groups and post them to the list weekly or every other week. That way, the network would serve among others as a digest of the most–important up–to–date information. The amount of necessary newsgroups and mailing lists that a PhD student needs to read in order to keep in track with the information flow might be radically reduced.
  (This might help with the information overflow and the reduction of necessary time.)
- More publicity in other forums (already suggested in the third workshop). More publicity would definitely enhance the image of the network and perhaps make it seem more attractive to further competent PhD candidates who currently only participate in other forums.
  (This might improve the overall image and the competence level of the network.)

One of the side effects of the above–mentioned approaches is also the chance to read about the problems and techniques in other OO–related domains which enables one to apply some of the proposed solutions to the own field. There could be some minor enhancements in the WWW, too:

- Create a single topic–dedicated page in the WWW maintained by different people every month. This would definitely seem attractive to many people and change the role of some members from passive readers to active writers.
- Dissemination of the survey results as an addition to the protected mail archive. This might become another complementing form of presentation for net surfers.

There were also some suggestions concerning the problem of lacking meetings, e.g. the quest for more workshops. All these suggestions might lead to a better image of the network that might encourage people to concentrate more on technical discussions and answers to problems (and finally solve the problem of low traffic?).

Milena Shteto: "Some rules on writing a PhD thesis"

Milena gave a lot of interesting advice on how to write about one's work and especially how to write one's PhD thesis. Her main messages can be summarized as follows:

- Write permanently! Otherwise you will forget many of the problems that led you to take your specific approach.
- When you write a paper start by collecting everything you have. Write it down.
- Continue by structuring the topics into sections and subsections. Write either just section titles or full sentences describing the essence of the section.
- Write fast, do not care about style. Get it done!
- Use examples. Illustrate your examples and ideas. Use pictures, charts, etc. to help the reader catch the essence of a long or important section in a visual form.
- When you're ready, put it away for several days, then get it again and reread it.
- Separate essential from non–essential parts. Delete the non–essential stuff.
- Give it to colleagues and let them read it.
- At the end, when all restructuring is finished, review your style, grammar, punctuation, etc. Do not bother with this earlier, it's just wasted time.

Milena's presentation was followed by a discussion on whether one should write the thesis in one's native language or in English. The discussion showed that many European universities either require PhD theses always to be submitted in the national language or they admit exceptions only when especially applied for (which can be more or less tedious).

The opinions of the participants on this point were very divergent. Many attendants said that writing permanently in English is a good training, which improves the ability to communicate ideas and results to an international audience. They expressed their discontentment with regulations that disallow writing one's thesis in a foreign language, which they considered an anachronistic hindrance for international cooperation. However, there were also participants who said that they are perfectly happy with such regulations, since they dislike writing in English anyway.

Walter Olthoff: Doing research on object–orientation

The invited talk by Walter Olthoff, the programme chair of ECOOP'95, was the highlight of the workshop's second day. In a relaxed mood he looked back at his own career, mixing in advices and lessons learned. This very personal style made the benefit of his advices stand out clearly and encouraged the audience to ask many questions, such that, after a while, the presentation turned into a lively discussion, which touched a variety of highly interesting topics. Unfortunately, we can only reproduce a few of the given recommendations within this report:

- Select your PhD topic carefully, being aware of what you would like to do when your thesis is finished! Students intending to join industry should have topics with practical relevance to industry people. It should be something "applicable" rather than being "completely theoretical".

- Whether a topic is "worth a PhD thesis" only depends on your advisor and the local referee committee deciding on your thesis.
- Communicate! Get in contact with others (e.g. attend PhD workshops!), discuss your work with peers. Seek people to which you can explain your work – if you succeed, your work is good; if you do not, improve its presentation. It is seldom the fault of your listener if he cannot follow your ideas.
- Do not let yourself be discouraged because you failed to explain your thoughts or the other guy did not like them! Try again! You grow out of each presentation you give, each explanation you try.
- Do not bore people! Learn to summarize your work in a few sentences. For presentations summarize on *one* slide: "Benefits of my work". Both is rehearsing your thesis' defense!
- Use all media you have at your fingertips! Read regularly Usenet News, Mailing Lists, etc. Be visible, get involved in discussions, do not be shy!
- When you publish pick *one* idea into one paper, not the whole thesis!
- Tell it how it is! You do not increase the value of a practical work by wrapping it in a pseudo–formalism. This approach might fire back, as your listeners/readers are generally smart enough to notice that you are trying to fool them.
- Write down names of people working in the same field and send them papers, asking for feedback. Send papers even if they are not yet "perfect". "Drafts" do it – but it must be mature work (except some "holes").
- Send your paper as hardcopy – do not ask the receiver to assemble a Postscript file of several parts just to find out that your Postscript versions are incompatible! Write a *short* acompanying letter, just enough to attract his or her attention. E.g. write why you address him or her and why your work and the receiver's work is related.
- Be open! After several years of doing the same things consider new topics. Sometimes the best way goes via a "deviation".

4 Reflections

The final session was concerned with possible improvements of further workshops and of the PhDOOS network. The following topics have been adressed:

One–and–a–half–days duration

The participants found the one–and–a–half–day duration to be well-suited for the workshop's purpose. The additional collisions with other workshops and tutorials seemed to be outweighted by more time for technical discussions. However, the distribution of available time across the different sessions could be improved. E.g. it was suggested that the time allocated for writing of discussion summaries should be shortened in favour of a mutual introduction of participants in the first plenary session.

Setup of working groups

As it came out during the final session of the technical part, some participants attended working groups that were unrelated to their work. This was partly due to the fact that there simply were no other people with related topics, which is something that can only be improved by having larger numbers of attendants. The other reason for unsatisfactory working group assignments were terminological confusions. Using

keywords as the only criterion for formation of working groups obviously was not enough. This could be ameliorated by earlier formation of groups and in–advance electronic discussions between the suggested members of a group. By taking part in some discussions and listening to others, prospective attendants might find out that a different group might better match their interests.

Combination of presentation with IPA method

Our variation of the IPA method was generally acknowledged. Although some people had difficulties in formulating their issue as a question that can be answered with *yes* or *no*, the IPA method was considered to be a good starting point for discussions and for structured gathering of arguments. The chance to provide necessary background information before starting an issue's discussion, was unanimously appreciated and proved even indispensable for very heterogeneous working groups, like the group on "Application of OO to other domains". However, it was suggested that attendants, especially advanced students, dare more to present something which is still open instead of presenting (almost) finished work.

Social contacts

The opportunity to meet people and establish new contacts was highly appreciated by the participants. For improving this aspect of the workshop, it was suggested to organize an additional informal get–together on the evening of the day before the workshop and to replace the workshop dinner by a reception, so one can walk around and talk to many people.

Sharing information

In the session on the PhDOOS network, it was generally agreed that locations of (electronic) papers and important references should be put on the WWW server. Calls for papers encountered by members should be made available both on WWW and by mail. References of value to the members should be put on the mail together with a short summary (why one thinks it is a good or bad reference). More elaborate summaries can go on the Web.

Session on "writing"

A very interesting suggestion for a non–technical session at the next workshop was to complement Milena's general advices on how to write a PhD thesis by a session giving very specific stylistic advice for writing papers. Every participant should immediately practice the presented "10 simple principles" by applying them to her or his own abstract.

Acknowledgements

In the first place, we have to thank the organizers of the ECOOP'95 conference in Aarhus, especially Jørgen Lindskov Knudsen and Eric Jul, for giving us every kind of support we asked for in organizing this workshop. This includes Erik Ernst, who organized our workshop dinner, and all the anonymous people who worked "in the background", making the whole conference a pleasant experience.

Special thanks are due to Walter Olthoff, for his very informative and stimulating invited talk and for encouraging discussions with the workshop organisers. Last, but not least, we wish to acknowledge the efforts of the workshop participants, who filled the whole event with life. We want to thank all who helped us writing this report by providing summaries of their issues and by commenting on previous versions.