

ECOOP 2002 Banquet Speech

13 June in Malaga

*By Kristen Nygaard,
The Norwegian Computing Center, the Simula Research
Laboratory, and the Department of Informatics,
University of Oslo
(kristen@simula.no; [http:// www.simula.no/~kristen/](http://www.simula.no/~kristen/))*

Dear ECOOP dignitaries of all ranks, from Antonio Vallecillo and Boris Magnusson on and sideways to the far borders. (In ECOOP we are democratic and arrange ranks sideways and not in ascending and descending order.) Dear conference delegates, dear spouses of all denominations.

At this early stage of the speech I will, unconventionally, ask you to join me in a toast to you, the audience. Why? At an informatics conference we should pay due attention to the user interfaces, also of speeches. A common nuisance associated with banquet speeches is that the well-behaved, waiting politely for the final toast, after a while become very thirsty, whereas the rest of you sneakily drink, and are instead left with empty glasses when the important final toast is proposed.

For this reason I asked the headwaiter to arrange for the filling up of your glasses to their brims just before the speech, as you may have noticed. I also will, by a series of toasts, accommodate the needs of the thirsty ones as the speech proceeds. (Toast)

You are going to listen to a speech that by tradition is longer than other ritual dinner speeches, but not longer than a lecture. The programme committee wants it to contain deep truths about object-orientation, and that it at the same time is illuminating for spouses, making them patient and understanding for yet another year, because of the overwhelming importance of the work of their spouses, and of this conference.

Let us start with a basic, language analytical approach. What kind of conference is ECOOP? *ECOOP is a language-oriented conference.*

This rich answer creates new questions: Which language do we speak at ECOOP? Not an easy one to answer, as I will show

you. Which languages do we speak about at ECOOP? (That is, which do we speak highly about?) Thanks to inheritance - an object-oriented invention - we may answer that one simply: Object-oriented languages with their subclasses.

At ECOOP, like most other international conferences, the "official language" is described as "English". Many British take this seriously and literally, and try to help the conference organisers by correcting the delegates' pronunciation, grammar and choice of words in a friendly way, like: "I think I understand you. You wanted to tell us that you prefer to fry tomAAatoes in an al-U-MIN-ium pan." Some US citizens may become annoyed, but the Brits self-assuredly state that: "after all, English is the Language of the World."

Microsoft, wanting to grab a profit in all markets, has introduced no less than three "English" languages in their program Microsoft Word. They are: English (UK), English (US), English (Aus), "Aus" meaning "Australian". One is easily aware of the latter as a spoken language, but thanks to Bill Gates it has become also a written one. This only makes the rest of us more confused.

Now the good news. In the Macintosh software world a new category has been introduced: "International English". This term required a definition, and Apple appointed a jury to deal with the question. Because of my long association with the Mac, I was made the chairperson. I was also allowed to announce the result of the jury's proceedings tonight. We all know from Oscar and Eurosong contest nights how this shall be done properly:

"Hello Malaga! Can you hear me? CAN YOU HEAR ME? Here is the vote of the international jury. (Opening of Oscar-like envelope.) The meaning of the term "International English" shall from now on be *"The Language of the World, that is, English as spoken, written and understood by foreigners"*, because of the overwhelming number of people speaking this language, and enriching it every day by adding new terms, interpretations and pronunciations."

We should, however admit that UK English is a wonderful language, and a language that has contributed massively to our Language of the World. Because of UK English I may address all of you in the audience very precisely: Some examples: Will all the *frogs* raise a hand? (*Frog* is a long-established term for a person from France.) Will all the *krauts* do the same? (A *kraut* is of course a *Fritz* or *Jerry*,

that is, a German.) I have noticed that we have *Yanks*, a *Kiwi* (raise your hand, James Noble!), and some *Japs* or *Nips* here. Do you know what a *dago* is? (A person living south of the Channel), An *arctic dago*? (A person living north of the Scottish border, including Scots, Hebrideans, Orkneyans, Faeroese, Icelanders, Inuits and Norwegians.) All UK English words. A US English contribution: *limeys from Limeyland*, meaning the English. Danish contribution: *mountain monkey*, referring to a Norwegian. Norwegian retort: *plain monkey*, ambiguity intended, meaning a Dane. What the Danes and the Norwegians call the Swedes may belong in International English, but I hesitate to mention it in a dinner speech. The Finnish International English term for Swedes may be quoted, however, since no one outside Finland will ever understand it: *Ruotsiperkele satänikainen hautajärvi*.

I propose that "International English" from now on shall be the official ECOOP language, to be used by us frogs, krauts, Japs, dagos and arctic dagos, plain monkeys, and others alike. The limeys shall, however, not be corrected when they speak UK English, in this way honouring the ancestry of the Language of the World.

This language issue settled, let us proceed to the next question.

We must ask: What is a speech at ECOOP? In particular: What is a banquet speech? The simple answer, available for you, the audience, is: What you are listening to, just now. This may be a convenient answer for you, but of no help to me.

Speeches and speakers come in many denominations. First, people in "Birds of a feather"-sessions, panels and behind poster sessions are not really speakers, even if they at some conferences are allotted badges with odd colours. The following formal definition will clear up the matter: "A *speech* is an oral uttering in a conference, referred to by a title in the written program and attributed to a named person, identified as the *speaker*."

Applying George Orwell's famous all-purpose quote from "Animal Farm" (1945): "All animals are equal, but some animals are more equal than others", we get: "All speakers are equal, but some are more equal than others."

The big divide is between invited speakers and those who have to convince their institutions that it is of paramount importance that they deliver their lecture in person, thus

competing for funds that have been irresponsibly shrunk in last year's desperate budget compromises.

If you are invited, you may be a *keynote speaker* or just an *invited speaker*. Both are OK, but the first is better. According to Merriam-Webster a keynote speech is "designed to present the issues of primary interest to an assembly and often to arouse unity and enthusiasm". And everyone will thus understand that the best is to be the "opening keynote speaker".

What about the "banquet speaker"? The banquet speaker is a joker in the pack of speakers. And what is "joker"? The three meanings mostly used are 1) a person given to joking, 2) an insignificant, obnoxious, or incompetent person, 3) a playing card added to a pack as a wild card or as the highest-ranking card. I prefer the "wild card". And what is "a wild card"? It is "one picked to fill a leftover playoff or conference agenda slot after regularly qualifying competitors have all been determined". Not very flattering, but certainly a very apt description of me at this conference. But it is also "an unknown or unpredictable factor". I have settled for that. Then, yesterday, I discovered in the program that I am the banquet speaker AND a keynote speaker, thus obliged to "arouse unity and enthusiasm".

We will not learn more by language analysis alone. Let us consider the pragmatics. Pragmatics is, we find, is: "relating to matters of fact or practical affairs often to the exclusion of intellectual or artistic matters". Just that!

The banquet usually takes place at the evening of the day before the last one of a conference. That is the evening during the conference when the delegates are most exhausted. To counter this, the delegates are as a rule served refreshments when they arrive, and trained conference delegates will usually find ways of getting more than the intended number of glasses.

When guests sit down, properly refreshed, wine bottles are circulated. In a country with such excellent wines as Spain, guests will freely avail themselves of this situation. The conversations around the tables reach summits of wit and profoundness. The wonderful main dish is consumed. The dessert is ready in the kitchen. Time has come for the banquet speech.

At this stage, the main challenge for the speaker is threefold: on one hand to quiet down the liveliest guests, on

another hand to keep the least lively guests awake. And, on the third hand, attention should be given to some sense in what is said to the rest.

At the first OOPSLA in Portland, Oregon in 1986, I was the opening keynote speaker and could pick all the good points. Alan Kay was the banquet speaker to a somewhat exhausted audience. He decided that a bold move was in place, and started by pronouncing: "We have all convened here to participate in an event dedicated to the most advanced and promising approach to computing - object-oriented programming. We are the pioneers! (Loud cheers followed.) But I have to ask myself: Is this certain? May be object-oriented programming is just a passing fad!" People were shocked, and Alan kept his audience to the end.

I was slightly irritated, though. Alan used the gimmick successfully once more at another conference. As an environmentalist he is known for being in favour of recirculation. The next time we met, I was again the keynote speaker and Alan the banquet speaker. I now really wanted to get rid of the "passing fad" gimmick.

I got hold of Larry Tesler, then at Xerox PARC and Alan's second-in-command, later father of the Lisa computer and Chief Scientist at Apple, now Vice President at Amazon.com. He was going to meet Alan before the banquet. I instructed him to tell Alan the following verbatim: "Kristen asked me to tell you that he in his keynote is going to tell this verbatim to the audience: "Alan may in his banquet speech tonight try to repeat his success with suggesting that " object-oriented programming may be just a passing fad". In order to vaccinate you, I will tell you a story about the ancient country of Arithmania.

In Arithmania the most revered god was Addo, the god of addition, and the mightiest of all his prophets was Subtracto, who in the midst of the dark ages brought light back to humanity through his divinely inspired invention of subtraction. In all sacred places you would find altars to Addo and Subtracto. Every university had separate faculties for addition and subtraction, exams and doctor's degrees were bestowed on the worthy ones.

Then two scientists from a remote province invented multiplication. What a sensation! Extraordinary sessions of learned societies were called, new conferences dedicated to multiplication multiplied (yes, multiplied, now they had a word for it). New chairs were created, in Formal

Multiplication Methods, Multiplication in Business Systems, The Ethics of Multiplication. Was there a need for separate Faculties of Multiplication? I now propose a toast to Multiplications, (Toast.)

Then one evening at a multiplication conference, at a banquet, a banquet speaker said: "We have all convened here to participate in an event dedicated to the most advanced aspect of arithmetic - multiplication. But I have to ask myself: Is this certain? I think multiplication is just a passing fad! I have news for you. The future lies in division!"

Alan got the message. Cunning as he is, he invented some other gimmicks, and delivered another excellent banquet speech. For Alan this was just a missed joke, but other people have seriously voiced similar opinions:

"We are all now witnessing the fall of the Object era" wrote Al Davis, former editor-in-chief of IEEE Software, July/August 1998, and in the Nov/Dec issue he wrote: "I think 'object' has now gone the way of 'structured' This was just as stupid as saying that multiplication is but a passing fad. Why?"

First of all, look around you! In this audience - none of you seem spectral to me. (I assume that you know that "spectral" is a nice UK English substitute term for ghostly.) In ACMs recent press release on the Turing Prize, it was stated:"object-oriented programming is the most widely used programming model today." It has" led to a fundamental change in how software systems are designed and programmed, resulting in reusable, reliable, scalable applications that have streamlined the process of writing software code and facilitated software programming."

Beautiful! Obviously we must now drink a toast to ACM! (Toast.)

And there is more to come: "It has been instrumental in developing a remarkably responsive programming model that is both flexible and agile when it is applied to complex software design and implementation," said John R. White, executive director and CEO of ACM. "It is the dominant style for implementing programs with large numbers of interacting components."

This is of course not news to you - people have known this for many years. But is good that someone outside our community says it. We need these objective, impartial quotes.

More important is the basic scientific stupidity of this passing fad notion. Listen:

Informatics (that is the correct term in *The Language of the World*) deals with phenomena: with computing processes, information processes materialising and developing inside computing equipment, in the brains of people, involving paper, CD-ROMs, telephone lines, and so on. Such a process has three main aspects:

First: Its substance, its material existence.

Second: The states of the attributes of the substance: The address of an invoice, the speed of an aircraft being followed on radar before landing at Malaga Airport, the price of lobster at a restaurant, the number of calls before yours in the telephone queue of a travel agency.

Third: The transitions bringing the attributes from one state to another: The updating of the position of the aircraft, the adjustment of lobster price to what was paid at the fish market, the counting down of number of calls before you when another customer has completed reserving his ticket or has become desperate by waiting and is leaving his position number 55 in the queue in disgust.

There are three major approaches to how you may organise a description of such information processes, each corresponding to one of these three major aspects.

If you decide to focus on substance, you are concerned with components and their interrelation, you are in object-oriented programming or, e.g., data base languages. *Substance-oriented programming.*

If you decide to focus on a single chain of transitions, you are in the old, well-known procedural programming. Or, if it is mandatory that all transitions are proven to be correct, as in autopilots for aircraft, or complex hardware switches, then you are in functional programming. *Transition-oriented programming.*

If you have a vast database containing knowledge about, e.g., petroleum geology, and have brought up a sample from drilling in the rock 3000 meter below the bottom of the North Sea, and now want to match the measured states of the attributes of that test against the knowledge in the database, in order to

know if you have struck oil - then you are in *state-oriented programming*, That is where logic programming belongs.

To state that object-oriented programming is a passing fad thus amounts to stating that one of the three basic approaches to the organisation of the computing process is a passing fad!

The demise of object-oriented programming was heralded, however, much earlier

The citation for our Turing Award is: "For ideas fundamental to the emergence of object oriented programming, through their design of the programming languages Simula I and Simula 67."

Reading the Award Committee's citation, some may get the impression that we had a brief spell of inspiration in the 1960s, and then were applauded ever after. This is very far from the truth. When Ole-Johan and I embarked upon the first Simula venture at the Norwegian Computing Center, it was said that: 1) this had been done before, 2) that it had not been done before, because it was of no interest, 3) Ole-Johan and I were not bright enough to carry out such a task, and 4) a project like this should not be carried out in a small and unimportant country like Norway.

When we worked on the next Simula language, Simula 67, introducing, e.g., inheritance, we were told that it was a futile effort because the giant IBM was launching its new general purpose programming language called PL/1.

Later, in the 1970s funding agencies around the world turned down language research applications because "everyone knew" that the Ada language effort, backed by the clout of the US Department of Defence, would certainly kill all further programming language research and development.

Finally, in the 1980s, language researchers were told that the Japanese much heralded and heavily industrial establishment- and government-supported "Fifth Generation Computer" project would imply that the Prolog language "would take over", and other languages would fade into obscurity.

What happened? How many new programmers use, or even know about PL/1 and Ada? The Fifth Generation Project is buried many years ago. Instead, during all these years, the ideas embedded in object-oriented programming gradually gained

foothold, among the users and researchers, in one application area after another. *The grass roots won over the giants.*

We had in 1967 declared that our ambition was to make Simula an "existing language" in the following sense: "Simula should be available on most of the major computer systems, and used over a long period of time. Simula should have a significant impact upon the development of future programming languages."

We succeeded in making Simula and object-orientation "existing" in this sense. It was a long and lasting commitment. It was the achievement of a large and increasing number of crusaders for the object-oriented concepts: The compiler implementation teams, the Association of Simula Users, the theoreticians using the class concept in their discussions of abstract data types. The UNIVAC company and all the friends we got there. The Smalltalk people at Xerox PARC who brought us a giant step forward by marrying object-orientation with a graphic interface, The designer of the C++ language, created to save heroes being entrapped in Unix caves, confronting the Jurassic Park monster called C. The Eiffel designer, introducing new, clever features. The Java team, linking up to Internet, and other language developers.

Our culture is to an exaggerated extent obsessed by winners. You may have winners also in research. In research, however, you do not win "over" others. You win "with" other people".

At this point of the speech, I must be allowed to mention an individual: Ole-Johan Dahl. It is impossible for me to describe in words how much he has given to me, and to object-oriented programming, and thus to all of us. Ole-Johan is now very ill. Still, between spells, he is very much himself, and I can convey his wishes for a good conference and a prolific new research year for all of you. Let us all rise and join in a toast for Ole-Johan! (Toast.)

Today a vital object-oriented community carries object-orientation forward. A significant section of that community is now convened here in Malaga. The wide and expanding range of the papers, the depth of their analysis support what I have just said.

Consulting the requirements, I think I now will have to address the "deep truths".

OK. What about the future? I think that the fields now opened up and cultivated should be further cultivated in the future. This being said, and emphasised, and not being a surprise or

a deep truth, I want to make some more subjective, personal remarks.

I think we have to go back to the roots. Object-orientation started with system description in operations research. The task was to devise concepts in terms of which it was possible for people to comprehend, describe and communicate about an extremely wide range of complex systems - and an associated language that could serve *both* as a system description language *and* a programming language, specifying instructions for a process generator (that is, a computer) to generate a phenomenon, a program execution process, that was a *model* of - structurally similar to - the *referent* system being described.

When much research and other work disappears from understanding for most of us, as it penetrates deeper and deeper into the intricacies of hardware and mathematics, it is time to realise that we don't have our conceptual basis in good shape. There is no universally agreed upon platform of concepts, taught to and understood by everyone, that through well-defined further specialisations in many directions may bring us to all the different branches of informatics.

I am convinced that it is possible to find such a platform. The notion of the information process, with its three basic aspects of substance, state and transition, may be defined with both precision and generality, to provide a conceptual layer below object-orientation. These information process concepts will be applicable both in specifying the three major programming styles of today, and in describing and generating the complex, distributed, multilayered, mobile systems now emerging. They should be taught and used in understanding the system development activities, in many of the sciences, in engineering and in administration.

Computing tend to move from one dominant language to another. Fortran, C++, Java are examples. During their dominance, the bandwagon effect is enormous, and system managers, bosses, insecure teachers and researchers cling to the safety offered by doing what everyone else is doing. But history shows us both that better tools emerge and win, and that the essential concepts and insights already gained will be parts of the future.

Even if Java today is a very good adaptation of the object-oriented ideas, it has serious flaws: It reintroduced Simula's and BETA's multithreadedness - hurrah! But it is done in a very clumsy way. For political reasons, the syntax

of C and C++ was adopted, very bad. There are security leaks that perhaps cannot be repaired without basic changes to the language.

How can we contribute in this general state of affairs in our science?

We can take a renewed interest in general system description. And we can make research and development in the teaching of informatics a key field of activity within the object-oriented community. Then we can produce students who do not demand only to become "employable", but instead become much more useful because they swear to Leonardo da Vinci's dictum: "The greatest of all joys is the joy of understanding".

Some researchers rather aggressively want informatics to be a formal science, voicing fundamentalist attitudes about shining purity. They divide workers in informatics into two camps: the Holy Order of The Clean Ones, and then The Others, those who do the dirty work by despicable working habits. I am happy to say that my strong impression is that those self-declared knights either do not know or do not understand object-oriented programming. (Some people know object-oriented programming without understanding.) This state of affairs makes it even more important that the object-oriented community enter the battlefield of education.

All over the world object-oriented languages are being used in introductory courses in informatics. But then the teachers proclaim that it is necessary to start with sufficiently simple examples, and carefully select the subset of old, procedure-oriented constructs for the first couple of months. When the students are thoroughly brainwashed, the teachers dare to tell them about objects and classes.

This must be stopped. Object-orientation is invented to tackle complexity, and teaching should instead start with sufficiently complex examples: Objects, classes, inheritance, virtual quantities and multiple action threads from the beginning. Students should learn like babies have to learn about the world. Do you tell your child to refrain from grabbing the glass of milk till you have made it understand the mechanisms of the eye and how it should cooperate with the muscular system? Start at the decomposition level that we confront in everyday life!

There are more deep truths, but these are the ones that I will use time on in the years I may have left to live.

I mentioned at the beginning of my speech that the organisers also wanted some illuminating words for the spouses about the true nature of object-oriented programming. OK, dear spouses. I will do my best, and my very best is to treat you as if you were royalty. I am not joking.

Those of you who know my country will no doubt have noticed that in Norway even staunch left-wingers are royalists at heart. How can this happen, when Webster's Thesaurus gives the following synonyms to "Royalist": REACTIONARY, Bourbon, Colonel Blimp, diehard, reactionary, ultraconservative? Let me give a contribution to a small part of the explanation.

Norway never got a feudal system. The farms were too small to support the accretion of excessive wealth, because of the topography, with valleys, mountains, forests and lakes. Also, a proletarian King in the late twelfth century beheaded all local chieftains cherishing such aristocratic aspirations. At the same time he said "No" to the Rule from Rome - the counterpart of latter-day's Treaty of Rome - initiating a still vital Norwegian tradition.

Even if we have no nobility, we have one Order: The Order of Saint Olav. Meticulous readers of my web site (that is, if such exist) may have observed that Ole-Johan and I were appointed Commanders of the Order of Saint Olav by King Harald of Norway in October 2000.

An important part of the ritual is that the Knights and Commanders are given an audience with our popular King at the Royal Castle

Since my last ECOOP banquet speech in Utrecht, I have had, because of that, the experience of explaining object-oriented programming to a King.

During my audience, the King insisted on getting a serious presentation of the main ideas in object-oriented programming, understandable for ordinary people. (His words, not mine.) Of course, I thought, he wants to be certain that the Commander's degree was well considered.

I explained to him (in much more detail than I am giving now) that scientific calculations started with tasks consisting of a very long sequence of operations, each operation usually involving the evaluation of some number given by a *formula*. Often the same kind of evaluation reappeared, as, e.g., the computing of a square root or a trigonometric function, or the arranging of a large set of number in ascending or

descending order. A correct, fast, safe and/or elegant rule for such a partial task was called a good *algorithm*. Thus the "world" of scientific programming was regarded as consisting of the tasks related to the specification of well-organised computing sequences, with formulae translated into subsequences, using well-designed algorithms.

When the first programming languages for scientific calculations appeared, they focused on the translation of formulae into the specification of sequences of computer executable operations upon data stored in the computer. The most formidably *successful* language of this kind was FORTRAN, *meaning formula translation*. The most *influential* language of this kind (in terms of long-range impact on language development) was ALGOL, *meaning algorithmic language*. This style of writing programs is usually called *procedural programming*.

Object-oriented programming is intended to describe and master a very different, and much richer world. Some examples:

- Our everyday tasks of city traffic and of air space control.
- Production by interaction and cooperation of many, often very dissimilar kinds of machinery, coordinated by telephone and data communication.
- Data processing of a varied and changing set of tasks in a complex organisation with many specialised branch offices working in parallel. The offices cooperating by executing partial tasks, often in shifting patterns, exchanging partial results until some final results are produced.

In object-oriented programming the programs are not organised in a single sequence. Instead, an *object* is created for each individual component of the traffic system, the production system, the organisation system: Cars, aircraft, machines, operators, metal parts to be processed, offices, documents - all are represented in the computer as objects.

Each object is described by 1) the set of *data* associated with it: Speeds of aircraft, weight of metal parts, addresses of documents, etc., 2) then the descriptions of *actions* the object is carrying out: traffic lights changing colour in a well-defined sequence, machines working on metal parts selected from a queue, and 3) finally also the *interactions* with other objects: a controller's office inspecting invoices

and sending messages about payment authorising to the appropriate other offices, and so on.

In this way a system, including information systems, may be understood, described and represented by a collection of objects in a computer, each object acting and/or is acted upon through the carrying out of the parallel object-related computing sequences.

The King leaned back in his chair, reflecting. Then he said: "Yes, I can see that. That is also how the data processing here at the castle in fact is happening."

Yes! I must admit that I was quite impressed.

I started this speech focussing on languages. I have often made the point that we should encourage reflection on spoken languages a part of building up intuition and comprehension of programming languages. The invention of inheritance is an important example on how spoken language constructs may successfully be carried over to programming languages. I have talked about the new Language of the World that has brought such a fantastic richness in words, interpretations and, most of all, pronunciations into what once was English (UK).

I cannot end this lecture, however, without a serious note of warning. I will tell a true story, an event that I personally witnessed at the very first large conference on object-oriented programming, the 1986 OOPSLA in Portland, Oregon. It shows how our Language of the World is open to distasteful, even shocking corruption in our multicultural global society, filled with new opportunities, but also with dangers.

I know that most object-oriented people are beings of delicate senses and exquisite tastes. Still, in system development they also have had to confront the ugly realities of life. Some of you probably even watch television, or have inadvertently been exposed to rap music texts. This makes me dare to tell the story. I must, however, ask those who are liable to faint when exposed to rough language to leave the audience now. (Pause to allow frail guests to withdraw.)

In Portland the organisers got three times more people than was expected. Consequently, the financial officer was in a happy and generous mood during the first lunch break. I was the opening keynote speaker, and was cordially invited to join the exuberant conference committee for lunch. I had not been promised any honorarium, but now I was told that instead I would be allowed to eat and drink, with my guests, what I

wanted in the hotel, at the expense of the conference. We had an excellent lunch, at the end of which the committee cheerfully left me at the table, thanking me for the wonderful meal I had treated them to.

I now realised that they were serious, and soon became popular with headwaiters and bartenders for my free-spending ways.

The final evening I had collected a farewell party in the restaurant, with two happy waiters satelliting our table, taking orders. Our current ECOOP Program Committee chair, Boris Magnusson, was among the guests and may corroborate what I tell you. Ole Hanseth, a fellow Norwegian, ordered a steak. "How do you want your steak?" asked the waiter, whereupon Ole started laughing, and an explanation was in place.

"A Norwegian couple, friends of mine," he told, "spent a weekend in London. On Saturday evening they dined at a plush restaurant in Soho, being attended by a waiter with a very stiff upper lip. The husband ordered a steak. "How do you want your steak, sir?" asked the waiter. Now I must tell you that blood in Norwegian is "blod", and the husband answered by translating the standard Norwegian expression "Jeg vi ha en blodig biff" into "I want a bloody beef!"

The waiter did not for second lose his composure, and responded immediately: "And would you like some fucking potatoes to go with your steak, sir?"

I must admit that we laughed. Then we became aware of choking sounds from the waiters. They were running for shelter in the kitchen. On their way they were intercepted by the headwaiter, they told him, and as he started laughing uncontrollably, he joined them,

This was bad enough. But bad went to worse. At the end of the meal we were approached by a very polite and correct headwaiter who asked us inordinately respectfully: "How did you gentlemen enjoy your fucking potatoes?"

In the opera, one should end with a *morale*, telling the audience what should be learned from the plot. Let this true story remind you that we have to be very careful with the languages we use and develop, and that mistakes may establish themselves quickly, and have long-lasting effects. In particular, I beg you not to answer too fast and

thoughtlessly next time a waiter asks you: "What kind of potatoes do you want to go with your food, sir?"

This being said, I have gone through the standard table of contents for banquet speeches, with some sprinkles of keynote added. The happy people seem still to be happy, the sleepy ones are not snoring, and I may conclude by proposing a toast to all those who are making this ECOOP a memorable one. (Final toast!)