The concept 'session' modelled from below *

Anders Moen
http://www.ii.uib.no/ nwpt00

November 1, 2000

Abstract

Traditionally Pi-calculus and its related families of theories attempt to be descriptive to the domain of discourse, by giving as small a set of primitives as possible to model as general a domain as possible. We take the tools from Milner's last book Milner [1999] and related theories in use to analyze the imprecise concept 'session'. We explain how this will be done by applying a scientific approach that we call 'conceptual archaeology'. The concept 'session' will be used in our future work as a benchmark to decide the fruitfulness of applying Pi-calculus as a means for doing conceptual analysis in a precise manner, rather than the common imprecise way done in Software Engineering.

1 Introduction

Theoretical computer science has traditionally been used to give a scientific, which means mathematical, foundation of new and old programming languages and to study computability in its purity [1]. Following the paradigm

*This paper is the result of a collaboration between Thor Kristoffersen, Wolfgang Leister, Bjarte Østvold and Martin Kirkengen. Thanks also to Kjell Åge Bringsrud for references and explanations of the session layer in the ISO-OSI model (he was a member on the commitee responsible for the session layer at ISO), and to my research director Naci Ökkur for trigging ideas initiating the paper. The paper gives the scientific foundation of the method used (or perhaps more correctly describes our attitude) in the Mowgli project, which is an ongoing internal project at Norwegian Computing Center. The participants are Bjarte Østvold, Arne-Kristian Groven, Håvard Hegna, Shahrzade Mazaher, Jon Haugsand and Abie Habtamu and the author. Special thanks to Jason Bagary, Bjarte Østvold, Xiuhua Zhang and Per Thomas Jahr for comments on an early version of this paper.

[1] Complexity theory, subrecursion theory and recursion theory including all their different formulations.

of Milner[2], we want to exploit the various formulations of the Pi-calculus and similar systems like the spi calculus of Abadi and Gordon [2000] and the theory of ambients of Luca Cardelli and Andrew Gordon [3], in order to do conceptual analysis of presumably well understood concepts in computer science, and especially mobile systems.

But it is not obvious that the new theories of theoretical computer science, will contribute to the evolution of mobile distributed systems, either on the conceptual or the technological frontier. Milner writes that:

> "(...) there are many ways of thinking about interactive systems - implies the need to tie these ways together. If a basic set of ideas such as the π-calculus can supply this integrity then designers will respect it, one may dare to hope, in the way that mechanical or electronical engineers respect the differential calculus, which ties together their ways of thinking."[4]

And he continues to reflect about the role his calculus should play;

> "Beyond this conceptual unification, a good outcome for the π-calculus would be to generate new high-level languages and analytical tools, much in the way that its predecessor CCS and CSP contributed to the design of LOTOS, a language designed to express communications protocols." [5]

As we can see from these paragraphs, Milner is certainly ambivalent to the reception of his calculus, both humble [6] and ambitious[7]. And he has good reason for this ambivalence. Distributed systems, mobile systems exists independent of the π-calculus. Why should anyone in the computing industry sit down and learn a new complicated theory when there is already so much to learn? Technology works so why bother? We shall indicate that the increasing complexity of the applications we deal with would benefit from formal and conceptual basic research.

In addition to Milner's conceptual unification and the generattion of analytic tools for constructing highlevel programming languages, we suggest a third approach. We will use the analytic tools for doing *conceptual analysis*, in our version that we shall call *conceptual archaeology.*

[2] As presented in Milner et al. [1992], Milner [1999], and Parrow [To appear]

[3] As presented in Cardelli and Gordon [2000b], Cardelli and Gordon [2000a] and Cardelli and Gordon [1999].

[4] Milner [1999] p.153.

[5] Milner [1999] p.153.

[6] "(...) one may dare to hope (...)"

[7] "(...) conceptual unification (...)"

2 The origins of conceptual analysis

The origin of conceptual analysis is ancient philosophy. In antiquity there was no division between science, theology and philosophy. This meant that scientific investigations were guarded by philosophical reflections, and to some extent, philosophical reflections were influenced by scientific considerations.

2.1 Gottlob Frege's contribution

A major turning-point in the history of philosophy and logic, is the work the German mathematician and philosopher Gottlob Frege in his writings in the two last decades of the 19'th century. Frege's contribution to science is two-fold. He clears the ground for modern first order logic in Frege [1879], Frege [1884] and second order logic in Frege [1893], as shown in Boolos [2000], by giving a formal system of first and second order logic and by giving birth to semantics. Frege is also the founder of analytic philosophy, by investigating metaphysical problems within the boundaries of language in Frege [1891], Frege [1892a], Frege [1892b] and Frege [1918], in arguing for the distinction between language, meaning and reference [8], pointing out that they should be divided into three distinct conceptual layers. Analytic philosophy is the commitment to do philosophy by scientific methods. This means to argue and formulate philosophical propositions in such a way that the propositions can be falsified. An analytic philosopher formulates her propositions with the commitment of being true or false.

For Frege conceptual analysis and formal logic are closely interrelated. By using a formal language he has a tool against which the concepts in mathematics and science in general can be calibrated. Frege investigated the laws of thought. The laws of thought could only be unravelled by a pure investigation into the logical laws of the assertorical propositions in a formal language [9].

2.2 Conceptual analysis after Frege

After Frege, conceptual analysis in philosophy, has gone in two directions with respect to the role formalization play in the analysis. The the first

[8] Freges levels are denoted Sprache, Sinn, and Bedeutung, which in english philosophical terminology is language, meaning and reference.

[9] Begriffschrift, a language for pure concepts

direction is theories of meaning [10] and reference [11], which we shall not consider in this paper, and the second is applied logic [12]. Frege's work relates more closely to mathematics than ordinary language and its sematics.

Applied logic grows out of Frege's work as the extension of formal methods applied to the domain of common concepts in ordinary language. Applied logic takes the concepts trancending first order logic serious, by introducing new operators capturing concepts like *obligation* and *permission* von Wright [1951], *knowledge* and *beliefs* Hintikka [1962], *time* and *computation* Goldblatt [1992], *typical situation*, *action* Segerberg [1993] and *counterfactual conditionals* Lewis [1973].

First order logic, and even propositional logic has its anomalies as shown in the paradox of the augmentation-principle:

> If the postman comes then you will get your letter. Then by propositional logic we infer: If the postman comes and he burns the letter, then you will get your letter

This anomaly of the materal implication can be solved by introducing a new modal conditional and a new set of axioms where strengthening the antecedent is rejected. But other anomalies may arise from the new system, where implication is given a domain specific meaning.

There are two observation related to this example. First, the example itself serves as an benchmark for testing theories, axiomatizations of non-monotone reasoning that philosophers might come up with. The set of benchmarks forms a finite set. Compared to classical theory of science, the benchmarks play the role of observations sentences, on which the theories (formalism) are calibrated. Second, the method of formalizing, testing, reformalizing, invent new benchmarks and then testing, reformalizing, testing, and so on, can be described as a circle movement. Progress in understanding is achieved by a back and forth movement, where the calculus change, but the set of benchmarks remains a relatively stable finite set.

2.3 Conceptual analysis as part of mature sciences

Conceptual analysis is not foreign to science before Frege. The treatment of infinitesimals in the differential calculus in the 17'th century, the scientific

[10] The theories of what 'meaning' and truth is.

[11] The theories concerned with the question, how can it be that names denote to objects in the world.

[12] Some might claim that this is not the case, that there is no real sharp distinction between the two directions. We find it fruitful to make the distinction because of the difference in perspective, technical skills and methods of the two groups.

dispute preceeding and including Lebnitz and Newton, was a great achievement in mathematics and physics, where the refinement of the mathematical concepts in the differential calculus where guarded by conceptual analysis. By doing this, they prepared the way for modern analytic geometry.

3 Reasons for doing Conceptual Analysis

Concepts are not given to us a priori, as precise and well understood. Concepts are constructed by humans.

Computer Science is a young science [13] There are at least four interrelated reasons for 'conceptual confusion' in computer science, which are *the immaturity of the science*, *the layers of abstraction*, *the quantity involved in computing*, *the rapid development of technology.*

First, computer science is a true hybrid of several sciences, physics, discrete mathematics and social sciences. To be more specific *electronics*, in searching for faster, smaller and more efficient hardware, *mathematical logic*, in designing circuits and designing programming languages, *cognitive psychology* and *sociology* to understand human interaction with the computer. The goal of research is to make computers work more efficiently in helping us solving practical problems in our daily life and at work.

Second, the use and design of computers relies on the notion of 'layer' of abstractions. To use in layers of abstractions means to use concrete and abstract concepts. But there seems to be too little awareness of how the creation of concepts take place and how concepts develop over time in computer science, as seen from the community itself[14].

Third, and most importantly, new technology and concepts evolve in a true egalitarian way. Compared to the classical scientific disciplines, computer science is brought forwards by hackers, businessmen and managers and engineers, and to a smaller extent by theoretical computer scientists. Scientific progress in computer science can be described by the processes of both building concepts and technology. Although the scientific communities contributes to foundational research, there is an increasing tendency that

[13] If it can be classified as a science at all. Computer science today is characterized more in its plurality of methods and divergence of perspectives, than a clear understanding of method and a limited domain of discourse.

[14] Theoretical computer science is of course an exeption, but we belong to a minority. Both from the perspective of education and industry theoretical computer scientists are pushed more and out in the dark. Evidence for this can be found throughout Europe, positions in theoretical computer science are withdrawn, and the companies apply for candidates with very technology-specific skills.

the evolution of existing concepts , new concepts and technology (understood in the widest sense) is driven forwards by the mass of people in the computer business outside the classical research institutes and universities.

Fourth, the rapid development of technology itself, relies on the capability of the contributers of the development to conceptualize what they are doing. Understanding means conceptualizing. Conceptualizing means building new concepts.

The consequence of this as seen from the perspective of computing engineering and computer science is a landscape of too many and too unclear concept. It is rather the rule than the exception that a scientific term has more than one meaning, and hence unclear meaning, or that several names have the same meaning in different communities of engineers and computer scientists.

3.1 Being formal about informal matter

To do conceptual analysis in computer science (understood in the widest sense) means to be confronted with actual usage of concepts in both informal discussions and scientific work. More than being a logical, platonical investigation of the internal relationship of the concepts them self, we have to investigate empirically the actual usage of concepts an that the change of the informal semantics of concepts.

The overflow of concepts and the diversity of meanings imposes the need for working in another direction than the usual way, as done in software engineering. It indicates the need for limitation. One such limitation is the decision to stay inside a formal language. A formal language is precise. A sound formal language can serve a tool for calibrating our understanding of a concept. In general concepts are not precise and their usages normally diverge. In the research frontier of applied computer science this is rather the rule than the exeption [15].

3.2 Three perspectives on formalization

The outcome of a process of formalizing a domain can be threefold, *descriptive*, *weakly normative* and *strongly normative.*

Having a *descriptive* perspective means to apply a formalism in order to give a taxonomy of a domain, so that we can reason about the domain and prove facts about it. Two examples fit the descriptive view. Frege's

[15] Mathematics is of course an exception, but not many people have time to listen to the mathematicians in the computing industry.

investigation of the laws of thought is an uncovering of a platonic reality. Verification of programs, to use formal tool in order to discover critical or dangerous consequences of running a program that controls a nuclear power plant [16].

To be *weakly normative*, means to investigate the usage of concepts and indicate, by finding inconsistency and incoherence in meaning, how usage could be changed in order to achieve clarity.

A *strongly normative* perspective means to investigate usage of a set of concepts and then using a formal language to specify a protocol, standard, programming language or a design paradigm, which everybody is supposed to follow. The normativity lies in the commitment for the users or programmers in applying the implemented version of the formal system.

RM-OPD is an standardization where the objective is "the developments of standards that allows the benefits of distribution of information processing services to be realized in an environment of heterogeneous IT resources and multiple organizational domains" [17]. It not clear whether RM-ODP could be considered to be strongly normative, but the way they formulate their objectives and motivation strongly indicates so. In RM-ODP, the Foundations, Architecture and Architectural semantics are all intended to be normative [18], but normative in what sense? The conceptual framework "is based on precise concepts derived from current distributed processing developments and, as far as possible, on the use of formal description techniques for specification of the architecture" [19]. The rapid development of new technology undermines the work of commitee's like the Open Software Foundation [20] or the Object Management Group [21]. Although the ISO-OSI standard was intended to be strongly normative, nobody follows it, and it is not likely to belive that somebody will in the future. A common situation in computer science is that intended

3.3 Conceptual archaeology

Our scientific method will therefore be to use the elements in the formalism as spades and hoes to dig out the meaning of concepts in computer science.

[16] One could of course claim that the verification showing a dangerous configuration itself is normative to the program itself, by commiting the programmer to go and fix the bug in the program.

[17] ISO [1995a] p. 6

[18] ISO [1995a] p. ii

[19] ISO [1995a] p. ii

[20] OSF for short

[21] OMG for short

Where no coherent meaning can be found we shall give extrapolations and refinements. We shall call our method 'conceptual archaeology'.

- The tools will determine the objects: The difference in expressibility of the systems would give us different shades of the concepts or genuinely different concepts.

  This is similar to the archaealogist using spades to dig in order to uncover an ancient building, but missing an golden ring. If she had used a spoon she would have found the ring but missed the building.

- Start of digging: An investigation of concepts must begin somewhere. The expected layers of diverging meaning covering a concept must be unravelled, but we should be careful not being limited in our investigation by the first community of computer scientists we ask for the meaning of a concept.

  The archaealogist must seek for the most reasonable place and the appropriate tool to start digging. A good archaealogist has an intuition for the landscape, where to start, and when to restart the search in another place in order to find more interesting objects.

- Realism and humbleness: Rather than being strongly normative with respect to the Domain of Discourse, our intention is to clarify and suggest reasonable interpretations that are founded in formal systems of concepts not including the concept for investigation. That is, we pretend only to be weakly normative with respect to the outcome of our analysis.

  The attitude of an archaealogist when finding a historical object is characterized by humbleness and curiosity in the interpretation of the objects for investigation.

- Phases of reflection and action: The work will shift between pure conceptual analysis - reflection and the hard work trying to make the output of the conceptual analysis fit into an adequate formalism.

  An archaealogist is both a scientist, interpreting and reflecting on the objects she finds, and a practical worker not afraid of getting dirty and tired in searching.

4 What 'session' might be

A short journey in the literature and on the web gives no clear understanding of the concept 'session'. It is a concept changing meaning gradually, covering new phenomenas when the time goes on.

In the reference model for Open Distributed Processing ISO [1995b] we do not find 'session'. But 'Liaison' has family resemblance with some of our interpretations of . 'Liaison' is defined by contractual context.

- Contractual context: the knowledge that a particular contract is in place, and thus that a particular behavior of a set of objects is required. An object may be in a number of contractual contexts simultaneously: the behavior is constrained to the intersection of the behaviors prescribed by each contractual context. [22] [23]

- Liaison: The relationship between a set of objects which results from the performance of some established behavior; the state of having a contractual context in common. [24]

But, this does not really help us very much. The problem is that RM-ODP, ideologically, is so closely related to the paradigm of object-orientation, so close that it might be an obstacle rather than an advantage to use the concepts in RM-ODP to explain or define 'session'. "Every ODP system specification is based on the concept of objects" ISO [1995a] p. 11. But it does seem to be the case that 'session' should be understood entirely in the paradigm of object-orientation. It might although be the case that one could *interpret* the concept in the paradigm of object-orientation.

Second, we could look up in a dictionary and find its meaning: In Oxford Advanced Learner's Dictionary we read that 'session' among other things means "single continous period spent in one activity". This captures the one of the common knowledges of how to use the word, and should be taken

[22] ISO [1995b] p. 13

[23] The examples of liaisons which result from different establishing behaviors are
a) a dialogue,
b) a binding,
c) a transaction,
d) an (N)-connection,
e) an association between (N)-entities enabling them to participate in (N) connection-less communication (as in OSI) f) a relationship between files and processes which access the files

[24] ISO [1995b] p. 13

Thirdly, one could look up in a canonical book on Computer networks and find that 'session' is a layer in a model for network-architecture that nobody uses anymore. In Tanenbaum [1996] p. 32-33 and Tanenbaum [1995] page 253-256 the concept session appears respectively as the 5'th layer in the ISO-OSI reference model and in the context of the discussion on the problems with file sharing in a distributed system. The important capabilities of the session layer is threefold, dialogcontrol, synchronization and resynchronization. To have *dialog control* meant that, in the old network architecture where two agents communicating had to share the same channel, there had to be a control mechanism deciding which agent that where supposed to communicate (transmit) at the moment. To *synchronize* meant to set timestamps on the data to be transfered over the channel, in order to be able to *resynchronize*, which meant that the session layer had capabilities of retransmitting lost packages from the sender to receiver, by using the timestamps known by both.

4.1 Examples of sessions

Instead of looking up in a book telling us what the concept really is, we could take the empirical approach and search after actual and potential usages of the concept. Let us describe some contexts of usage that is called sessions, or might be called sessions in the future:

- I have logged in on my computer without any network-connection. [25]
- I am logging in, then running several processes on a server machine from my computer at my office, and log off at the end of the day. [26]
- An ultra thin client, on which I can use a special card keeping my ID to access my session from the ultra thin client next door, and continue to work within the same context of programs running on the server. [27]
- I use special software Y to access a server-machine from a terminal X_1 equipped with an operating system Z_1, get an ID for this session and use it for a while, and then log off the and the next day I connect via a a terminal X_2, running an operating system Z_2, and run the software

[25] Personal session on a personal computer at home.

[26] A session on a remote time sharing system.

[27] Session mobility, realized in network of Sunray terminals connected to a Sunray-server, where a Sunray-card gives you access to your session.

Y on X_2 by the same ID as yesterday and continue to work with the same processes running on the server machine. [28]

- I start writing a letter to a friend on my stationary computer, but have to leave in a hurry, so I continue to write on my portable computer equipped with a antenna keeping the connetion to the server computer. After some minutes somebody must borrow my lap-top and therefore I continue to work on the letter on my little Personal Digital Assistant, still connected to my original server, being able to look what I have written up till now [29]

- I join a MUD-like game, that started at time t_0 on a server machine X, and start playing at time t_1, but get killed at time t_2 which is game over for me. The game itself will go on for some while and end at t_3. My participation in the game is certainly a session. But it is likely to suggest that the game or the administrator relation to the game is a session as well. [30]

- I join an ongoing MUD-game just like above, but while the game proceed, the Game-administrator moves the game-server from its original terminal to a new one.

- I am a participant in a real-time game over internet, where an arbitrary number players interact in real-time where the whole game event appears as a continous event for every participants, in spite of the delay on the network connecting the players. [31]

4.2 The primitive concepts

The examples given above might suggest a preliminary informal definition or rudimentary description of 'session':

> A session is an event limited in time. It is a temporal concept, with a start-point and an endpoint. A session is a relation be-

[28] Sessions are kept alive even though one is logged off. VNC - Virtual Network Computing. An ID-number chosen by you, and the UNIX server machine memorizes your processes and when requested, unfolds the encapsulated processes again. My request the next day can be from a MacIntosh and I only need to download a VNC client for this specific machine to have a graphical userinterface exactly the same as yesterday.

[29] Seamless movement of a session through several devices.

[30] Distributed sessions.

[31] This is a very hard technological challenge. There are no really good solutions at the moment on how to do this.

tween a subject and a possible many objects. A session involves concepts of communication.

To know the meaning of a concept, is to know what falls under the concept and that which does not. It entails the capability of making distinctions. One way to do this, is to find the primitive concepts that can be used to define the concept, but are theoretically simpler. A suggestion might be:

time, place, state, communication, interpretation, identity, process

4.3 Five questions one could ask about the concept 'session'

- What is an example of the simplest session?
- Which other fundamental concepts would session rely on?
- Should it be possible to fork and merge sessions?
- Are there any eguality predicates for sessions?
- Are sessions ordered in hierarchies?

The questions are both ontological and algebraic. They are ontological in the sense that the answers will determine which events fall under the concept and which does not, in other words they will determine what 'session' is. They are algebraic in the sense that confirmations to the questions will unfold algebraic properties of session-expressions as formulated in a formal theory [32] .

5 Concluding remarks

Our group is settled in an applied Research Institute where one of our concerns is inventing new ideas for applications and speculates about future requirements and technologies for mobile systems. There are two reasons why we consider Pi-calculus to be a promising framework to stay inside [33].

First, Pi-calculus is a formal calculus expressed in an abstract language, that gives the opportunity of stating technology independent properties of a

[32] Where sentences in a language of sessions compile down to the language of π-calculus or some similar system.

[33] Nisse Husberg proposed to us that pertri-nets could be a fruitful framework to stay inside, and we shall invesigate this track in the future.

mobile system or an application running on a mobile device. Secondly, technology evolves rapidly and freely with a continuous conflict between the need for standardization and specialization, between compatibility and incompatibility involving competing hardware, software and telecom-companies.

We suspect that formal methods has a role to play in on-the-edge technology that we are exposed to where the needs for tools for thinking is critical. As discussed above, we also expect to find several 'session'- concepts, and hopefully a unifying framework to reason about a subset of these. But time will show whether we succeed or not.

References

M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: the spi calculus. *Computation and Information*, 2000. to appear.

G. Boolos. *Logic, Logic and Logic.* Harvard University Press, 2000.

L. Cardelli and A. D. Gordon. Types for mobile ambients. In *POPL 1999*, 1999.

L. Cardelli and A. D. Gordon. Anytime, anywhere. In *POPL 2000*, 2000a.

L. Cardelli and A. D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177–213, 2000b.

G. Frege. *Begriffschrift: Eine der arithmetischen nachgebildete Formelsprache des reinen Denkens.* Halle, 1879.

G. Frege. *Foundations of Arithmetics.* Oxford University Press, 1884. Translation from Breslau: Wilhelm Koebner.

G. Frege. Function and concept. In *Translations from the Philosophical Writings of Gottlob Frege.* Oxford University Press, 1891.

G. Frege. On concept and object. In *Translations from the Philosophical Writings of Gottlob Frege.* Oxford University Press, 1892a.

G. Frege. On sense and meaning. In *Translations from the Philosophical Writings of Gottlob Frege.* Oxford University Press, 1892b.

G. Frege. *Grundgesetze der Arithmetic.* Georg Oms, 1893.

G. Frege. Thoughts. In *Logical Investigations.* Basil Blackwell, 1918.

R. Goldblatt. *Logics of Time and Computation.* CSLI, 1992.

J. Hintikka. *Knowledge and Beliefs: an introduction to the logic of the two notions.* Ithaca N.Y., Cornell University Press, 1962.

ISO. Open distributed processing - reference model - part 1 overview, 1995a. Obtained from the Open Group.

ISO. Open distributed processing - reference model - part 2 foundations, 1995b. Obtained from the Open Group.

D. K. Lewis. *Counterfactuals.* Harvard University Press, 1973.

R. Milner. *Communicating and mobile systems: The Pi-calculus.* Cambridge University Press, 1999.

R. Milner, J. Parrow, and D. Walker. A calculus of mobil processes part i and ii. *Information and Computation*, 100(1):1–77, 1992.

J. Parrow. An introduction to the π-calculus. In *Handbook of Process Algebra.* ed. Bergstra, Ponse and Smolka, To appear.

K. Segerberg. *Dynamic Logic.* unpublished, 1993.

A. S. Tanenbaum. *Distributed Operating Systems.* Prentice Hall International Editors, 1995.

A. S. Tanenbaum. *Computer Networks.* Prentice Hall, 1996.

G. H. von Wright. Deontic logic. *Mind*, 60:1–15, 1951.