VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kansliets noteringar Kod	Dnr
2014-20305-115114-19	2014-5478

2014
Project Research Grant

Area of science
Natural and Engineering Sciences
Announced grants
Research grants NT April 9, 2014
Total amount for which applied (kSEK)

2015	2016	2017	2018	2019
1185	1126	1176	1259	1352

APPLICANT

Name(Last name, First name)
Parrow, Joachim

Date of birth
561225-1453

Gender
Male

Email address
joachim@it.uu.se

Academic title
Professor

Position
Professor

Phone
0705733324

Doctoral degree awarded (yyyy-mm-dd)
1986-01-10

WORKING ADDRESS

University/corresponding, Department, Section/Unit, Address, etc.
Uppsala universitet
Institutionen för informationsteknologi
Datalogi
Box 337
75105 Uppsala, Sweden

ADMINISTRATING ORGANISATION

Administrating Organisation
Uppsala universitet

DESCRIPTIVE DATA

Project title, Swedish (max 200 char)
Ett ramverk för parallelprogrameringsspecifikationer

Project title, English (max 200 char)
A Framework for Parallel Programming Specifications

Abstract (max 1500 char)
This project is a direct continuation of the project "A framework for parallel programming models", to accommodate and verify a spectrum of high-level modeling languages. We formally established correctness in a theorem prover, and provided an accompanying prototype analysis tool. As a result we unified many existing high-level modeling formalisms, and corrected errors in several several of them. We have also shown how to construct new such formalisms in a systematic way.

Now we shall extend this framework to make it easier to apply, and include important elements motivated by realistic applications. We shall include specification languages such as modal logics and modal transition systems, and consider the accompanying problems of model checking and refinement checking. We shall address non-functional aspects such as resource usage, time consumption, and probabilistic behavior. We intend to support a user to define a domain specific formalism, by providing off-the shelf components and ways to combine them. We shall continue our pioneering work with the theorem prover and tie it more closely to the prototype analysis tool. Theory and tools will be developed in parallel with significant applications to guide both.

Kod
2014-20305-115114-19

Name of Applicant
Parrow, Joachim

Date of birth
561225-1453

Abstract language
English

Keywords
parallel programming, process algebra, modal logic, theorem prover,

Review panel
NT-2

Project also includes other research area

Classification codes (SCB) in order of priority
10201, 10206, 10205

Aspects

Continuation grant
Application concerns: Continuation grant
Registration Number: 2009-4543

Application is also submitted to

similar to:

identical to:

ANIMAL STUDIES

Animal studies
No animal experiments

OTHER CO-WORKER

Name(Last name, First name)
,

University/corresponding, Department, Section/Unit, Addressetc.

Date of birth

Gender

Academic title

Doctoral degree awarded (yyyy-mm-dd)

Name(Last name, First name)
,

University/corresponding, Department, Section/Unit, Addressetc.

Date of birth

Gender

Academic title

Doctoral degree awarded (yyyy-mm-dd)

Name(Last name, First name)
,

University/corresponding, Department, Section/Unit, Addressetc.

Date of birth

Gender

Academic title

Doctoral degree awarded (yyyy-mm-dd)

Name(Last name, First name)
,

University/corresponding, Department, Section/Unit, Addressetc.

Date of birth

Gender

Academic title

Doctoral degree awarded (yyyy-mm-dd)

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Name of Applicant
Parrow, Joachim

Date of birth
561225-1453

ENCLOSED APPENDICES

A, B, C, D, N, S

APPLIED FUNDING: THIS APPLICATION

Funding period (planned start and end date)
2015-01-01 -- 2019-12-31

Staff/ salaries (kSEK)

Main applicant	**% of full time in the project**	**2015**	**2016**	**2017**	**2018**	**2019**
Joachim Parrow						
Other staff						
PhD Student	80	484	496	534	578	626
PhD student	80	567	496	508	547	592
Total, salaries (kSEK):		1051	992	1042	1125	1218

Other project related costs (kSEK)	**2015**	**2016**	**2017**	**2018**	**2019**
Computer Costs	30	30	30	30	30
Premises	40	40	40	40	40
Travel Costs	64	64	64	64	64
Total, other costs (kSEK):	134	134	134	134	134

Total amount for which applied (kSEK)

2015	2016	2017	2018	2019
1185	1126	1176	1259	1352

ALL FUNDING

Other VR-projects (granted and applied) by the applicant and co-workers, if applic. (kSEK)

Funds received by the applicant from other funding sources, incl ALF-grant (kSEK)

POPULAR SCIENCE DESCRIPTION

Popularscience heading and description (max 4500 char)

De flesta har någon gång råkat ut för datortrassel. Sådant beror nästan alltid på fel i datorns program - fel på den faktiska maskinen är ganska sällsynta. Ett program kan bestå av miljontals rader text som beskriver i detalj vad datorn ska göra, och det är svårt att i förväg övertyga sig om att det kommer att fungera på rätt sätt. En möjlighet att göra detta är att analysera programmet med någon logisk metod. Det finns flera olika sådana som är anpassade för olika typer av program. Men hur ska man veta att metoden i sig är riktig?

Här kommer så kallade teorembevisare in i bilden. Dessa är en sorts program som specialiserats på att utföra resonemang i matematisk logik. Vi avser tillämpa sådana teorembevisare på logiska analysmetoder, för att bevisa bortom allt tvivel att de är riktiga. Det gör att analysmetoderna

kan tillämpas i trygg förvissning om att de resultat de ger är korrekta.

Ett problem i sammanhanget är att teorembevisare är svåra att använda och kräver mycket arbete för varje tillämpning. Därför kommer vi att söka behandla analysmetoderna på ett generiskt sätt, och hitta delar som är gemensamma för flera metoder. Nya analysmetoder utvecklas ständigt. Därför kommer vi också att bygga upp ett litet bibliotek med vars hjälp en teorembevisare kan behandla nyutvecklade metoder på ett smidigt sätt.

I slutändan leder detta till att fler program kan analyseras och säkras innan de används. Datortrassel kommer nog aldrig helt att försvinna. Men det kommer att minska. Vår forskning är grundforskning och möjliggör bättre vapen mot programbuggarna.

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Name of applicant
Parrow, Joachim

Date of birth
561225-1453

Title of research programme
A Framework for Parallel Programming Specifications

Appendix A

Research programme

Vetenskapsrådet, Box 1035, SE-101 38 Stockholm, tel. +46 (0)8 546 44 000, vetenskapsradet@vr.se

A Framework for Parallel Programming Specifications

Joachim Parrow
Uppsala Universitet
email: joachim.parrow@it.uu.se

April 8, 2014

1 Summary

This project is a direct continuation of the project *A framework for parallel programming models*, to accommodate and verify a spectrum of high-level modeling languages. We formally established correctness in a theorem prover, and provided an accompanying prototype analysis tool. As a result we unified many existing high-level modeling formalisms, and corrected errors in several several of them. We have also shown how to construct new such formalisms within the framework in a systematic way.

Now we shall extend this framework to make it easier to apply, and include important elements motivated by realistic applications. We shall include specification languages such as modal logics and modal transition systems, and consider the accompanying problems of model checking and refinement checking. We shall address non-functional aspects such as resource usage, time consumption, and probabilistic behavior. We intend to support a user to define a domain specific formalism, by providing off-the shelf components and ways to combine them. We shall continue our pioneering work with the theorem prover and tie it more closely to the prototype analysis tool. Theory and tools will be developed in parallel with significant applications to guide both.

2 Motivation and Goals

This project is on basic research in high-level formal specification and modeling languages for computer systems. We use the term *modeling* language for a formalism that describes the structure and behavior of a system, and *specification* language for a formalism to pose requirements on a system. This is not a strict division, there are hybrid variants encompassing both. Our preceding project focussed on modeling formalisms; we here aim to extend it to include specification formalisms and to make it more easily available in practical applications.

Modeling and specification formalisms are needed to give mathematically precise answers to questions such as how programs behave and the effect of combining them, and are important for verification procedures and formal analysis algorithms to operate on a high level of abstraction. An example of a desired proof rule in any such language is substitution: replacing a part with a behaviorally equivalent part will not change the

overall behavior of the whole system. Another is type safety: If a model typechecks, then types will not be invalidated during execution.

Programming and software development is a highly diverse field, and there is no single formalism to take care of all application domains. On the contrary there is now a plethora of languages, and the trend is to stick to those developed in-house. The ongoing proliferation of such "domain specific" languages comes with a great risk: when they are developed hastily and from scratch they may be internally inconsistent. Formal semantics may yield unintended results and formal proof rules may be invalid. We have discovered several errors of this kind, including in the substitution rule mentioned above. The fact that formalisms do not follow their specifications precisely has also been reported by other researchers, e.g. in [38]. If the models used to verify software components are as unreliable as the things they are supposed to verify, the relevance of the whole research area is undermined!

In our preceding project we constructed a general framework for building provably correct high-level modeling formalisms. In this so called psi-framework we have several strong results. It can accommodate a great variety of existing formalisms. The meta-theory of any instance of the framework is verified mechanically in a theorem prover, meaning that the confidence in correctness is extremely high.

The goal of the proposed project is to extend the psi-framework to specification formalisms and make it easily available to significant applications. It should be easy to construct a domain specific formalism, using off-the shelf components and ways to combine them. We will guarantee correctness by construction, in other words, if the formalism is built following our design rules then it automatically obtains a machine verified meta-theory including proof rules such as substitution. The formalism should admit formal specifications of requirements, and give automated support to determine whether they hold for particular models. It should also cover issues related to resource usage and stochastic properties. This requires major new theoretical developments, and implementation efforts in tools and theorem provers.

3 Background

3.1 Theories for concurrency

Models for concurrency are notoriously difficult, not least because the programming languages are complicated and their semantics, in a mathematical sense, largely undefined. There are many models and languages, often with a core of primitive concepts, such as "input", "output", "parallel", "scope" etc. Therefore the problems have been studied in the context of skeletal languages, which are small enough that they can be given a complete and manageable definition, yet containing enough essential constructs from real programming languages to be of interest. A large class of these languages are the so called process calculi [9].

Process calculi began around 1980 with pioneering work by Milner, Hoare, and Bergstra. Ten years later the pi-calculus (Milner, Parrow, Walker) demonstrated convincingly how two kinds of name bindings (scope localization and input parametricity) coexist in a very parsimonious formalism, and this became the starting point for many subsequent developments. Today there are very many descendants of the pi-calculus to

incorporate a variety of aspects, the most well-known being the spi-calculus [5] and the applied pi-calculus [4], both including arbitrary algebraic data types and with strong applications in cryptography. Theory issues center on various kinds of operational semantics, bisimulation equivalences, and type systems. There is no good overview of the field and efforts to systematically compare different strands [35, 23] are moderately successful. Many process calculi have significant tool support and important practical applications, for example μCRL [18].

Specification languages for process algebras include modal logics, and most are based on the so called Hennessy-Milner logic (HML), originally developed in the early 1980's [26, 27]. The idea is to extend ordinary predicate logic with modalities of the kind "can do an action *alpha* and continue to satisfy a formula F". Considerable expressiveness is gained by allowing recursive formulas defined as minimal or maximal fixed points (as first demonstrated by Larsen [30]). HML was extended to the pi-calculus [34] in the early 1990's where the main issue is how the name bindings in the actions interact with the modalities. Logics for more general message passing systems [25] have also been investigated for more applied languages like LOTOS [14] and the applied pi-calculus [36].

3.2 Theorem provers

A so called theorem prover is s computer program that determines if a logical proof is correct. The first was developed by de Bruijn in the late 1960s and there are today a large number of different ones, each with their own strengths and weaknesses. Most theorem provers are interactive with automated tactics, and the user can provide additional proof strategies. Significant advances in applications related to software are summarized in the POPLmark Challenge [7], a set of benchmarks intended both for measuring progress and for stimulating discussion and collaboration in mechanizing the metatheory of programming languages. In the CompCert project [2] a realistic C compiler is being completely verified. These types of tools are now being transferred to commercial products.

In our project we use Isabelle/Nominal [39] for two main reasons. The Nominal datatype package allows an efficient treatment of bound names. This is a major concern in all advanced theories of computation, and of particular importance for theories with scope openings, as in the pi-calculus, where computations can result in a bound name becoming free. The Isar interface greatly improves on proof readability. This is particularly useful in a project like ours where specialists and non-specialists on theorem provers continually interact during theory developments.

3.3 Summary of achieved results

We have developed the framework of psi-calculi, which provides machine-checked proofs that important meta-theoretical properties, such as compositionality of bisimulation, hold in all instances of the framework. The main idea is that a psi-calculus is defined by a number of parameters: what are the possible data structures that can be transmitted, what predicates are defined on them, when do different terms represent the same communication channels etc. Most importantly, the framework contains a means to define logical properties that hold for the data structures, so called

assertions (for example that decrypting with the correct key always yields the encrypted data). Processes can use logical predicates to test properties, and also to assert new logical facts (akin to constraint programming) and scope such information locally through name bindings. Extensions of the framework includes broadcast communication and higher-order communication, and a sorting system to determine things like which data are allowed where. We have demonstrated that a significant amount of existing advanced process algebras can be represented in the framework by instantiating the parameters in the right way. Enclosure D of this applications contains a more detailed account of our results.

The psi-calculi workbench is a generic tool for implementing particular psi-calculi (by defining the appropriate parameters), and for analysing processes in the resulting calculi. It currently provides functionality for bisimulation equivalence checking, simulation refinement, and symbolic simulation (or execution) of processes, and a base library for implementing new psi calculi.

A substantial part of the meta-theory of the psi framework has been formally verified in the interactive theorem prover Isabelle using the Nominal datatype package. The current total size of the proof files is around 300 KLoC, making this formalization effort the largest in the process algebra community and the largest application of the Nominal package. As we have accounted for elsewhere the benefits of formalization have been huge in comparison to the costs. Of all effort spent on the psi framework, around one third of the manpower has been on formalization, and it has provided us with unique abilities to obtain correct and extensible theories.

We have gained substantial experience on practical applications in the area of Wireless networks, analyzing a routing protocol in connection with the Profun project, and multicore architectures, describing a cache coherence protocol within the Upmarc project. Both these projects are collaborations with other groups in the department.

4 Project Description

The project will build on the success of its predecessor and develop the framework and tools in directions to become more useful. In particular we intend to pursue:

- Development of a modal logic for expressing specifications, and model checkers for determining that processes meet specifications.
- Development of modal transition systems to enable more readable specifications, together with a refinement relation to determine that processes implement specifications.
- Development of a semantics that captures resource usage, time, probabilities etc in a general way by ascribing metrics to actions.
- Further development of the workbench to make it more accessible, both for analyzing process behaviors and for defining new psi-calculi as part of the framework.
- Continuing the formalization effort in Isabelle of new developments and migrating to a more modern version of the Nominal datatype package.

These directions are not independent. The workbench and the formalization can be regarded as integral parts of the first three, with theory and implementation conducted

concurrently. In the following we shall expand on these directions, and point out further areas of interaction.

4.1 Theory developments

4.1.1 Modal logics

A process in a psi-calculus is a definition of behavior. It is then natural to consider logics where properties of such behaviors can be defined and verified. We have already conducted preliminary investigations on what a counterpart of Hennessy-Milner logic would be for a psi-calculus. Clearly the modalities are derived from the actions, and the interplay of binding actions, modalities and fixed-points should not represent a major hurdle since these issues have been well studied in the pi-calculus. The intriguing new aspects have to do with the logical assertions already present in the psi-framework. A process in a psi-calculus can assert a formula to hold, meaning that it introduces new logical facts; thus a specification in a HML logic should reflect this ability.

An interesting question is what further aspects to include. Logics for many applications in cryptography use epistemic elements for reasoning about what principals can deduce what facts (eg. [22, 16, 24] and many more). Other logics include spatial elements to directly reason about the structure, as in who performs what action (eg. [15, 12, 13, 32]). Integration of such ideas with psi-calculi constitute a more advanced development which will be considered in the later half of the project. Linear time logic also has distinct advantages and should be considered eventually, though it is probably harder to integrate in the framework than HML which builds directly on the existing process syntax.

To be of use there should be a procedure for determining whether a given property holds for a given process, the so called model checking problem. The principles behind the original model checker for the pi-calculus [10] is described by Dam [19]. Subsequent work has developed the method [40, 17] and opened for model checking in extensions with arbitrary data domains [31, 33]. The problem in the general case is that model checking will certainly not be decidable. Thus efforts will proceed along two lines: identifying decidable fragments of psi-calculi, and providing heuristic support or incomplete methods (which are not guaranteed to terminate) for more advanced cases.

4.1.2 Modal transition systems

A related way to specify properties of behavior is in so called modal transition systems. Introduced by Larsen and Thomsen in the late 1980's [29, 28] they provide a natural way to consider processes as implementations of specifications. Rather than using two separate formal languages (a process calculus and a HML logic) there is a single formalism, akin to a process calculus but where actions are indexed by modalities like "may" and "must". A modal process refines, or implements, another modal process if it contains all the must actions and does not stray outside the may actions. Applications, complexity issues and tool support for modal transition systems are summarized in [6]. For more applied calculi with arbitrary data terms, modal transition systems have been considered in e.g. [21, 8], showing promise in both more realistic applications and tool support. We are not aware of modal transitions for the pi-calculus or similar formalisms with name bindings in the actions.

The idea of modal transitions is simple

enough to be directly transferrable to both the pi-calculus and the psi framework. Action prefixes would need to contain modalities, but the prefixes already have a rich structure and possibly the modality could be embodied in an existing feature, e.g. the sort for the prefix subject. The main problems are the implications for bisimulation and definition of a new refinement relation. Our aim is to establish theorems for modular construction (e.g., if P refines Q then $P|R$ refines $Q|R$), hopefully reusing some elements of the corresponding proof for bisimulation.

There is an interesting trade-off between modal logics and modal transition systems. Modal logics are more general and can in some sense encode modal transitions, while the other direction does not hold, for example modal transition systems cannot directly represent negation. Modal transitions constitute a much more readable format for large applications and is more accessible for non-specialists (recursive HML formulas for the pi-calculus are hard also for specialists!), and requires less changes to the syntax. A decision which to focus on in this project should be taken after preliminary investigations and experiments with both. Important questions that need to be answered include how readable and expressive the formalisms are for practical applications, and what the consequences are for tool support.

4.1.3 Semantic weights

The semantics in the psi framework currently does not admit performance measures such as time, energy, resource usage, or measures such as reliabilities and probabilities of transitions. These properties are often important and several formalisms and tools deal explicitly with them. Process algebras incorporating such measures abound (several chapters of [9] give overviews) and have also been considered for the pi-calculus since the mid 1990's [37]. Extending the psi framework in this direction will mainly be a challenge in keeping the extensions general and uniform. We shall develop such measures in a parametric way, meaning that several different measures are obtained by instantiating parameters appropriately. The advantage is that we can then verify the meta-theory of many different measures in one go.

One way is to use a so called weighted semantics [20]. The idea is to use a commutative semiring $(S, 0, 1, +, \cdot)$ with carrier S representing the measure under consideration. Addition $+$ with unit 0 is the operation of combining two measures disjunctively (such as when there are alternate ways to derive a transition), and multiplication $\cdot$ with unit 1 combines them then conjunctively (eg. when several are needed simultaneously to derive a transition). The theory of the psi-framework can then be developed uniformly for an arbitrary semiring. Time, probabilities, resource usage etc will be defined by choosing an appropriate semiring for each. It is even conceivable that the modalities of modal transitions systems (Section 4.1.2 above) can be represented in a semiring. The corresponding notion of refinement might then be an instance of a more general simulation preorder.

4.2 Implementation

4.2.1 The Workbench

The workbench has two types of users. Type I is the ordinary user of a verification tool, analysing systems in an existing instance of the framework. Type II is the implementor of new such instances, defining

domain specific formalisms and putting together the necessary parts.

For Type I users many improvements are needed. A better (possibly graphic) interface, increased functionality, inclusion of model checkers for modal logics or refinement checkers, resource usage analysis, etc. This work is contingent on the theoretical developments of the preceding sections, and can be partly conducted in parallel to guide it towards areas of practical interest.

More interesting is the support for Type II users, where the workbench is unique (other tools such as mCRL [18] and ProVerif [11] are parameteric to a much lesser extent). In order to build a new instance of the framework, a Type II user need only specify the parameters, i.e. data structures and logic to reason about data, verify a few properties required for psi-calculi, to get a language with a formally verified meta-theory. An important research area is to determine general operators on these parameters that preserve the properties of a psi-calculus. Examples could include increasing the data terms by new constructors, or adding new factors to the logical assertions. Our initial results in this direction are ad-hoc but suggest there might be a pattern. The ultimate goal is to provide a user with building blocks for psi-calculi and ways to combine them in the workbench. In this way the framework becomes easier to use and more available to non-specialists when constructing domain specific models.

The workbench is dependent on external constraint solvers in order to compute transitions. A Type II user thus needs to plug in a constraint solver for the logic in the particular calculus. The underlying so-called symbolic semantics is currently not available for general pattern matching, and extending the symbolic semantics in that direction would enable more high-level calculi. Our preliminary investigations indicate that this would require higher-order unification algorithms and not only constraint solvers.

4.2.2 Formalization in Isabelle

Sections 4.1.1 through 4.1.3 are all natural candidates for extending the formalization. We shall continue our strategy to formalize the theories during the development (as opposed to post-hoc efforts) since it maximizes the benefits and leads to less backtracking.

Modal logics have already been implemented in Isabelle several times. Of particular interest here are implementations of Lamport's temporal logic of actions, e.g. [3], where parts can conceivably be re-used, though since the name binding scheme is different this is far from certain.

An ongoing and important task is migration to Nominal 2, the new nominal package. It is much more efficient and allows construct for multiples name bindings such as sets or sequences of binders, greatly reducing the effort in the psi-framework where these occur frequently. During this work we will have some influence over the developments in Nominal 2, since we are its by far largest customer.

We have found that many of the proofs are partly similar. Although formally independent, large parts are obtained by copy-paste from existing proofs. This is particularly true when a new version of the framework is established, such as the extension to higher-order processes. It is therefore conceivable that there are automatic tactics that could guide and simplify the formalization effort. We now have enough experience to begin development in this direction.

An independent extension concerns the symbolic semantics used by the Workbench.

A formalized proof of how it relates to the standard structural operational semantics would increase confidence in the output. The formalization will also be connected to the workbench as a backend checker of analysis results. For example, if the workbench produces a transition sequence or even a bisimulation, these could be formally verified in Isabelle, thereby eliminating the risk of a workbench bug producing erroneous results. This connection requires a common format for psi calculi in the formalization and the Workbench.

5 Originality

The preceding project was highly original in two respects: a parametric framework to unify a large set of advanced process calculi, and formalization of the meta-theory in an interactive theorem prover. The proposed project is highly original also in other respects. Hennessy-Milner logics have not been studied for a parametric framework, and not for calculi with advanced features such as the psi-calculi assertions where processes can dynamically declare new facts. Modal transition systems have not been studied even for the pi-calculus or other calculi with binders in the actions. Different kinds of weighted semantics have been studied for simple process algebras, but a generic treatment is missing. The formalization of modal transition systems and weighted semantics in a theorem prover also represent new advances. The concurrency workbench will be original in its support for a Type II user, facilitating formally verified domain specific formalisms.

6 Feasibility

Our group has demonstrated beyond doubt the construction of a very general framework, the productive and essential use of a theorem prover, and applications to numerous existing calculi. For the extension to Hennessy-Milner logic we have preliminary results; also the principal investigator has worked actively on such logics for the pi-calculus. Modal transition systems represent a new area where we have preliminary contact with their original developer Kim Larsen. We believe that the main issues will be related to how they can be used. Here we will greatly profit from our work on and experience in applications. The same holds for a weighted semantics.

The workbench now has one PhD student and one research fellow (Gutkovas and Borgström) working on the implementation and semantics. The principal investigator and a senior faculty member (Victor) have personal experience of building similar tools. Another senior faculty member (Weber) has considerable experience of Isabelle, and we have excellent contacts with the leading groups in that area (Nipkow at TI Munich, Paulson in Cambridge, Urban at King's College London).

We intend to recruit two PhD students for this project, one starting 2015 on tool support, and one starting 2016 to replace Åman Pohjola who now works on theory development and Isabelle. The students will work within and be advised by members of our group.

7 Impact

In this project we shall pave the way in establishing how correct high-level specifica-

tion formalisms can be built systematically. Our general framework will be a platform for designing domain specific calculi and logics in a structured way. This will have impact for those who develop high-level formalisms, both those who invent new ones and those who want to certify the correctness of already existing ones. By extension there will be impact for those who use such formalisms to model and verify computer systems in various forms, and for those who develop modeling and verification techniques on high abstraction levels using domain specific formalisms. It already has a significant impact: We have discovered several errors in existing formalisms and demonstrated how to correct them.

The theories we will treat are among the most complex ever to have been subjected to proof mechanisation and will require new techniques to be handled efficiently. Our work has already had an impact on the development of the nominal datatype package in Isabelle. Being its largest customer we will continue to have impact, notably in strategies and tactics to handle structures of bound names. We may have a long range impact on the development of theorem provers and therefore also on commercial verification tools for software construction where theorem provers are integrated.

Our simultaneous participation in more applied projects such as Profun and Upmarc represent large collaborations within the department, on wireless sensor networks and multicore architectures respectively. We shall continue to apply our framework to concrete problems in these areas. The research group also participates in the Cost action Betty [1] on behavioral types. These can be likened with very high-level specifications, related both to processes and to logics. Betty will ensure a good spectrum of potential applications as well as a forum to discuss the relevance of our techniques.

References

[1] Behavioural types for reliable large-scale software systems (betty). `http://www.cost.eu/domains_actions/ict/Actions/IC1201`.

[2] Compcert: Compilers you can formally trust. `http://compcert.inria.fr/`.

[3] A definitional encoding of tla* in isabelle/hol. `http://afp.sourceforge.net/entries/TLA.shtml`.

[4] Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *Proceedings of POPL '01*, pages 104–115. ACM, January 2001.

[5] Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Fourth ACM Conference on Computer and Communications Security*, pages 36–47. ACM Press, 1997.

[6] Adam Antonik, Michael Huth, Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. 20 years of modal and mixed specifications. *Bulletin of the EATCS*, 95:94–129, 2008.

[7] Brian E. Aydemir, Aaron Bohannon, Matthew Fairbairn, Nathan J. Foster, Benjamin C. Pierce, Peter Sewell, Dimitrios Vytiniotis, Geoffrey Washburn, Stephanie Weirich, and Steve Zdancewic. Mechanized metatheory for the masses: The POPLmark challenge. In *International Conference on Theorem Proving in Higher Order Logics (TPHOLs)*, August 2005.

[8] Sebastian S. Bauer, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. A modal specification theory for components with data. *Sci. Comput. Program.*, 83:106–128, 2014.

[9] J. A. Bergstra. *Handbook of Process Algebra*. Elsevier Science Inc., New York, NY, USA, 2001.

[10] Fredrick B. Beste. The model prover - a sequent-calculus based modal μ-calculus model checker tool for finite control π-calculus agents. Master's thesis, Department of Computer Systems, Uppsala University, Sweden, March 1998. Available as report DoCS 98/97.

[11] Bruno Blanchet. Using Horn clauses for analyzing security protocols. In Véronique Cortier and Steve Kremer, editors, *Formal Models and Techniques for Analyzing Security Protocols*, volume 5 of *Cryptology and Information Security Series*, pages 86–111. IOS Press, March 2011.

[12] Luís Caires and Luca Cardelli. A spatial logic for concurrency (Part I). *Information and Computation*, 186(2):194 – 235, 2003.

[13] Luís Caires and Luca Cardelli. A spatial logic for concurrency—II. *Theoretical Computer Science*, 322(3):517 – 565, 2004.

[14] Muffy Calder, Savi Maharaj, and Carron Shankland. A modal logic for full lotos based on symbolic transition systems. *The Computer Journal*, 45(1):55–61, 2002.

[15] Luca Cardelli and Andrew D. Gordon. Anytime, anywhere: Modal logics for mobile ambients. In *Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '00, pages 365–377, New York, NY, USA, 2000. ACM.

[16] Rohit Chadha, Stéphanie Delaune, and Steve Kremer. Epistemic logic for the applied pi calculus. In David Lee, Antónia Lopes, and Arnd Poetzsch-Heffter, editors, *Proceedings of IFIP International Conference on Formal Techniques for Distributed Systems (FMOODS/FORTE'09)*, volume 5522 of *Lecture Notes in Computer Science*, pages 182–197, Lisbon, Portugal, June 2009. Springer.

[17] Taolue Chen, Tingting Han, and Jian Lu. A modal logic for pi-calculus and model checking algorithm. *Electronic Notes in Theoretical Computer Science*, 123(0):19 – 33, 2005.

[18] Sjoerd Cranen, JanFriso Groote, JeroenJ.A. Keiren, FrankP.M. Stappers, ErikP. Vink, Wieger Wesselink, and TimA.C. Willemse. An overview of the mcrl2 toolset and its recent advances. In Nir Piterman and ScottA. Smolka, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 7795 of *Lecture Notes in Computer Science*, pages 199–213. Springer Berlin Heidelberg, 2013.

[19] Mads Dam. Model checking mobile processes. *Information and Computation*, 129(1):35 – 51, 1996.

[20] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of weighted automata*. Springer, 2009.

[21] MiguelValero Espada and Jacovande Pol. An abstract interpretation toolkit for mucrl. *Formal Methods in System Design*, 30(3):249–273, 2007.

[22] Ulrik Frendrup, Hans Hüttel, and Jesper Nyholm Jensen. Modal logics for cryptographic processes. *Electr. Notes Theor. Comput. Sci.*, 68(2):124–141, 2002.

[23] Daniele Gorla. Towards a unified approach to encodability and separation results for process calculi. *Inf. Comput.*, 208(9):1031–1053, 2010.

[24] Dimitar P. Guelev and Mads Dam. An epistemic predicate ctl* for finite control - processes. *Electr. Notes Theor. Comput. Sci.*, 278:229–243, 2011.

[25] M. Hennessy and X. Liu. A modal logic for message passing processes. *Acta Informatica*, 32(4):375–393, 1995.

[26] Matthew Hennessy and Robin Milner. On observing nondeterminism and concurrency. In J. W. de Bakker and Jan van Leeuwen, editors, *ICALP*, volume 85 of *Lecture Notes in Computer Science*, pages 299–309. Springer, 1980.

[27] Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985.

[28] Kim Guldstrand Larsen. Modal specifications. In Joseph Sifakis, editor, *Automatic Verification Methods for Finite State Systems*, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 1989.

[29] Kim Guldstrand Larsen and Bent Thomsen. A modal process logic. In *LICS*, pages 203–210. IEEE Computer Society, 1988.

[30] KimG. Larsen. Proof systems for hennessy-milner logic with recursion. In M. Dauchet and M. Nivat, editors, *CAAP '88*, volume 299 of *Lecture Notes in Computer Science*, pages 215–230. Springer Berlin Heidelberg, 1988.

[31] Huimin Lin. Model checking value-passing processes. In *Software Engineering Conference, 2001. APSEC 2001. Eighth Asia-Pacific*, pages 3–10. IEEE, 2001.

[32] Etienne Lozes and Jules Villard. A spatial equational logic for the applied pi-calculus. *Distributed Computing*, 23:61–83, 2010.

[33] Radu Mateescu and Gwen Salaün. Pic2lnt: Model transformation for model checking an applied pi-calculus. In Nir Piterman and Scott A. Smolka, editors, *TACAS*, volume 7795 of *Lecture Notes in Computer Science*, pages 192–198. Springer, 2013.

[34] Robin Milner, Joachim Parrow, and David Walker. Modal logics for mobile processes. *Theoretical Computer Science*, 114(1):149 – 171, 1993.

[35] Joachim Parrow. Expressiveness of process algebras. *Electronic Notes in Computer Science*, 209:173–186, 2008.

[36] Michael Pedersen. Logics for the applied pi calculus, 2006. BRICS RS-06-19, Aalborg University.

[37] Corrado Priami. Stochastic pi-calculus. *Comput. J.*, 38(7):578–589, 1995.

[38] Frank P. M. Stappers, Michel A. Reniers, Sven Weber, and Jan Friso Groote. Dogfooding the formal semantics of mcrl2. In Jonathan P. Bowen, Huibiao Zhu, and Mike Hinchey, editors, *SEW*, pages 90–99. IEEE Computer Society, 2012.

[39] Christian Urban and Christine Tasson. Nominal techniques in Isabelle/HOL. In Robert Nieuwenhuis, editor, *CADE*, volume 3632 of *Lecture Notes in Computer Science*, pages 38–53. Springer, 2005.

[40] Ping Yang, C. R. Ramakrishnan, and Scott A. Smolka. A logical encoding of the pi-calculus: model checking mobile processes using tabled resolution. *STTT*, 6(1):38–66, 2004.

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Name of applicant
Parrow, Joachim

Date of birth
561225-1453

Title of research programme
A Framework for Parallel Programming Specifications

Appendix B

Curriculum vitae

Vetenskapsrådet, Box 1035, SE-101 38 Stockholm, tel. +46 (0)8 546 44 000, vetenskapsradet@vr.se

Joachim Parrow Curriculum Vitae

April 2014

Higher education qualifications

BSc (högskoleexamen från Matematikerlinjen), Uppsala University, November 1980. Major subjects: Mathematics and Computer Science.

Degree of Doctor

PhD (Tekn. Doktor) in Computer Science, Uppsala University, January 1986. Thesis title: Fairness Properties in Process Algebra. Advisor: Björn Pehrson

Postdoctoral positions

Visiting Researcher at the University of Edinburgh, Department of Computer Science, for in all 18 months 1986–1989. Employed on a grant from the Science and Engineering Research Council (grant holder: Robin Milner).

Qualification required for appointment as a docent

Uppsala University 1990.

Present position

Professor (Chaired Professor) in Computing Systems (Datalogi) from May 2010 at the Department of Information Technology, Uppsala University. There is no fixed amount of research associated with the position. The last few years I have been doing approximately 60% research and graduate (PhD level) education, 20% undergraduate education, and 20% departmental service.

Previous positions

Professor in Computer Systems (Datorteknik) from January 2002 until May 2010 at the department of Information Technology, Uppsala University.

Professor in Distributed Systems from October 1994 until June 2002 at the department of Microelectronics and Information Technology (until 31/12/2000 the department of Teleinformatics) at the Royal Institute of Technology (KTH), Stockholm.

Researcher from February 1986 until September 1994 at the Swedish Institute of Computer Science, leading the research group on formal methods.

Interruptions in research

Dean of Education 1 July 2005 – 30 June 2008 at the Faculty of Science and Technology, Uppsala. The volume of the undergraduate education was around 4300 full time equivalent students among around 25 programmes, and the yearly turnover over 300 million SEK.

Head of Undergraduate Education 1 July 2002 - 30 June 2005 at the department of Information Technology.

Head of Department (Prefekt) at the department of Teleinformatics, KTH, April 1997 – October 2000. The department employed around 60 persons and the turnover 1999 was 55 million SEK.

Deputy Head of Department 1996-1997.

Director of Graduate Studies, 1996–1997.

Supervision

PhD Students: I have advised nine students for a PhD:
Peter Sjödin 1991, Fredrik Orava 1994, Björn Victor 1998, Gunnar Övergaard 2000, Jose Vivas 2000, Lars-åke Fredlund 2001, Oskar Wibling 2008, Jesper Bengtson 2010, Magnus Johansson 2010 (where I was a very active co-supervisor).

I have advised seven students for a Tekn. Lic. I currently advise two PhD students.

Other information

I am an ISI highly cited researcher `http://isihighlycited.com/`.

Invited speaker on many occasions, most recently DisCoTec 2014 (Berlin).

External grants: I have been principal investigator for the Swedish contribution of several EU-projects (CONCUR, CONCUR2, CONFER, EXPRESS), and was the co-ordinator of the EU-project PROFUNDIS (1.2MEuro, 2002–2005). I have led several nationally funded projects (by VR, TFR and NUTEK). I am a co-proposer of The VR Linneaus center UPMARC and the SSF-funded project ProFun, both starting 2008.

Other Professional Services (highlights) Member of the Editorial Board for *Formal Aspects of Computing*. Editor or guest editor of seven collections or special issues. Conference chair of five prestigious international conferences, eg ICALP 2003. Member of 14 other programme committees for international conferences. Board member of the Swedish Institute of Computer Science 1999 – 2002. Board member and one of the originators of the NUTEK Competence Center ASTEC (Advanced Software Technologies), 1994–1998.

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Name of applicant
Parrow, Joachim

Date of birth
561225-1453

Title of research programme
A Framework for Parallel Programming Specifications

Appendix C

List of publications

Vetenskapsrådet, Box 1035, SE-101 38 Stockholm, tel. +46 (0)8 546 44 000, vetenskapsradet@vr.se

Joachim Parrow: Publications 2007–2014

All articles can be obtained from my homepage `http://user.it.uu.se/~joachim/cv.html`. Citation count is from Google Scholar, 1 April 2014. The five publications most relevant for the project are labeled *. All publications are peer-review original articles in journals or conferences.

1. Jesper Bengtson, Joachim Parrow. Formalising the pi-calculus using nominal logic. In H. Seidl (Ed) Proceedings of FOSSACS 2007, pages 63-77. Published as Springer Verlag LNCS 4423 (2007). An extended version is in *Logical Methods in Computer Science* 5(2:16) 2009. Number of citations: **35**

2. Jesper Bengtson, Joachim Parrow. A completeness proof for bisimulation in the pi-calculus using Isabelle. In the proceedings of SOS2007, *Electronic Notes in Theoretical Computer Science* **192**:1, pages 61-75 (2007). Number of citations: **5**

3. Joachim Parrow. Expressiveness of Process Algebras. *Electronic Notes in Theoretical Computer Science* **209**, pages 173–186 (2008). Number of citations: **29**

4. Magnus Johansson, Joachim Parrow, Björn Victor, and Jesper Bengtson. Extended pi-calculi. In Proceedings of *ICALP 2008*, LNCS 5126, pages 87–98, Springer Verlag 2008. Number of citations: **11**

5. Jesper Bengtson, Magnus Johansson, Joachim Parrow, and Björn Victor. Psi-calculi: Mobile processes, nominal data, and logic. In Proceedings of *LICS 2009*, pages 39-48. IEEE Computer Society (2009). Number of citations: **41**

6. Jesper Bengtson, Joachim Parrow. Psi-calculi in Isabelle. In Proceedings of *Theorem Proving in Higher Order Logics*, LNCS 5674, pp 99-114, 2009 Number of citations: **21**

7. Magnus Johanssom, Björn Victor, Joachim Parrow. A Fully Abstract Symbolic Semantics for Psi-Calculi. In Proceedings of *SOS'09*, EPTCS vol. 18 pp 17-31 2010. Number of citations: **13**

8. Magnus Johansson, Jesper Bengtson, Joachim Parrow, and Björn Victor. Weak equivalences in psi-calculi. In Proceedings of *LICS 2010* pages 322-331, IEEE Computer Society 2010 Number of citations: **11**

9. Jesper Bengtson, Magnus Johansson, Joachim Parrow, and Björn Victor. Psi-calculi: A framework for mobile processes with nominal data and logic. *Logical Methods in Computer Science* 7(1:11) pp 1-44 (2011). Number of citations: **24** *

10. Johannes Borgström, Shuqin Huang, Magnus Johansson, Palle Raabjerg, Björn Victor, Johannes Åman Pohjola, and Joachim Parrow. Broadcast Psi-calculi with an Application to Wireless Protocols. In Proceedingd of SEFM 2011, LNCS 7041 pp 74–89, 2011. A longer version is accepted for publication in *Software and Systems Modeling*, volume 13, Springer, 2014. Number of citations: **12**

11. Magnus Johansson, Björn Victor, and Joachim Parrow. Computing strong and weak bisimulations for psi-calculi. *J. Logic and Algebraic Programming*, 81.3 pp 162-180, 2012. Number of citations: **7** *

12. Joachim Parrow, Johannes Borgström, Palle Raabjerg, Johannes Åman Pohjola. Higher-order psi-calculi. *Math. Struct. in Comp. Science* Available online CJO 2013 Number of citations: **7** *

13. Johannes Borgström, Ramunas Gutkovas, Joachim Parrow, Björn Victor, and Johannes Åman Pohjola. A Sorted Semantic Framework for Applied Process Calculi. Proc. of *Trustworthy Global Computing (TGC 2013)* LNCS 8358 pp 103-118, 2014 *

14. Jesper Bengtson, Joachim Parrow, and Tjark Weber. Psi-calculi in Isabelle. Accepted for publication in *Journal of Automated Reasoning* 2014. *

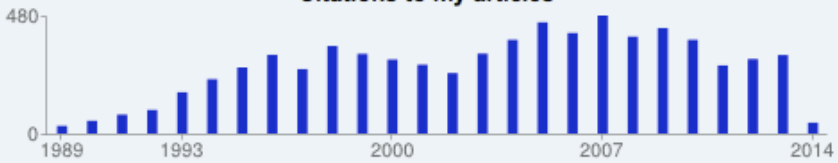
Joachim Parrow: Citation analysis and 5 most cited papers

Analysis according to Google scholar, 1 April 2014:

Citation indices

	All	Since 2009
Citations	7369	1756
h-index	27	13
i10-index	42	20

Citations to my articles



Show: 20 **1-20** Next >

Title / Author	Cited by	Year
A calculus of mobile processes, i R Milner, J Parrow, D Walker Information and computation 100 (1), 1-40	4164	1992
The Concurrency Workbench: A semantics-based tool for the verification of concurrent systems R Cleaveland, J Parrow, B Steffen ACM Transactions on Programming Languages and Systems (TOPLAS) 15 (1), 36-72	732	1993
The fusion calculus: Expressiveness and symmetry in mobile processes J Parrow, B Victor Logic in Computer Science, 1998. Proceedings. Thirteenth Annual IEEE ...	279	1998
An introduction to the-calculus J Parrow Handbook of Process Algebra, 479-543	253	2001
Modal logics for mobile processes R Milner, J Parrow, D Walker Theoretical Computer Science 114 (1), 149-171	241	1993

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Name of applicant
Parrow, Joachim

Date of birth
561225-1453

Title of research programme

Appendix D

Scientific report/account of scientific activity

Vetenskapsrådet, Box 1035, SE-101 38 Stockholm, tel. +46 (0)8 546 44 000, vetenskapsradet@vr.se

The project

The project *A Framework for Parallel Programming Models*, Dnr 2009-4543 has been supported by VR from 2010 and consumed the following resources: 2010 347 KSEK, 2011 411 KSEK, 2012 750 KSEK, 2013 536 KSEK, 2014 161 KSEK.

Summary of achieved results

We have developed the framework of psi-calculi, which provides machine-checked proofs that important meta-theoretical properties, such as compositionality of bisimulation, hold in all instances of the framework. The main idea is that a psi-calculus is defined by a number of parameters: what are the possible data structures that can be transmitted, what predicates are defined on them, when do different terms represent the same communication channels etc. Most importantly, the framework contains a means to define logical properties that hold for the data structures, so called *assertions* (for example that decrypting with the correct key always yields the encrypted data). Processes can use logical predicates to test properties, and also to assert new logical facts (akin to constraint programming) and scope such information locally through name bindings. The following have been the main developments conducted within the project:

Weak equivalences [4] We establish the theory of weak bisimulation, where the tau actions are unobservable. In comparison to other calculi the presence of assertions poses a significant challenge. We demonstrate that the complications mainly stem from psi-calculi where the associated logic does not satisfy weakening. We prove that weak bisimulation equivalence has the expected algebraic properties and that the corresponding observation congruence is preserved by all operators. These proofs have been machine checked in Isabelle. We also prove that weak barbed equivalence coincides with weak bisimulation equivalence

Decision procedures for bisimulation [5] We present a symbolic transition system with strong and weak bisimulation equivalences, and show that they are fully abstract with respect to bisimulation congruences in the non-symbolic semantics. A procedure which computes the most general constraint under which two agents are bisimilar is developed and proved correct. Symbolic semantics is necessary for an efficient implementation of the calculus in automated tools exploring state spaces, and the full abstraction property means the symbolic semantics makes exactly the same distinctions as the original.

Broadcast communication primitives [3] We add primitives for broadcast communication in order to model wireless protocols. The additions preserve the purity of the psi-calculi semantics, and we formally prove the standard congruence and structural properties of bisimilarity. We demonstrate the expressive power of broadcast psi-calculi by modeling the wireless ad-hoc routing protocol LUNAR and verifying a basic reachability property.

Higher-order calculi [6] We define higher-order psi-calculi through a technically surprisingly simple extension of the framework, and show how an arbitrary psi-calculus can be lifted to its higher-order counterpart in a canonical way. We illustrate this with examples and establish an algebraic theory of higher-order psi-calculi. Interestingly, replication and nondeterministic choice can be encoded using the higher-order constructs. The formal results are obtained by extending our proof repositories in Isabelle/Nominal.

Sorts [1] We extend our previous work on psi-calculi with novel abstract patterns and pattern matching, and add sorts to the data term language, giving sufficient criteria for subject reduction to hold. Our framework can directly represent several existing process calculi; the resulting transition systems are isomorphic to the originals up to strong bisimulation. We also demonstrate different notions of computation on data terms, including cryptographic primitives and a lambda-calculus with erratic choice. Substantial parts of the meta-theory of sorted psi-calculi have been machine-checked using Nominal Isabelle.

The Workbench [2] We present a generic tool for analyzing processes from any psi calculus instance, and for implementing new instances with the help of a supporting library. The Workbench implements symbolic execution and bisimulation algorithms for both unicast and wireless broadcast communication. We illustrate the Workbench by examples from the pi-calculus and the area of wireless sensor networks.

References

[1] Johannes Borgström, Ramunas Gutkovas, Joachim Parrow, Björn Victor, and Johannes Åman Pohjola. A sorted semantic framework for applied process calculi (extended abstract). In Martín Abadi and Alberto Lluch Lafuente, editors, *Trustworthy Global Computing*, volume 8358 of *Lecture Notes in Computer Science*, pages 103–118. Springer, 2013.

[2] Johannes Borgström, Ramunas Gutkovas, Ioana Rodhe, and Björn Victor. A parametric tool for applied process calculi. In *13th International Conference on Application of Concurrency to System Design*, pages 180–185. IEEE, 2013.

[3] Johannes Borgström, Shuqin Huang, Magnus Johansson, Palle Raabjerg, Björn Victor, Johannes Åman Pohjola, and Joachim Parrow. Broadcast psi-calculi with an application to wireless protocols. In Gilles Barthe, Alberto Pardo, and Gerardo Schneider, editors, *Software Engineering and Formal Methods*, volume 7041 of *Lecture Notes in Computer Science*, pages 74–89. Springer, 2011.

[4] Magnus Johansson, Jesper Bengtson, Joachim Parrow, and Björn Victor. Weak equivalences in psi-calculi. In *Logic in Computer Science*, pages 322–331. IEEE Computer Society, 2010.

[5] Magnus Johansson, Björn Victor, and Joachim Parrow. Computing strong and weak bisimulations for psi-calculi. *J. Log. Algebr. Program.*, 81(3):162–180, 2012.

[6] Joachim Parrow, Johannes Borgström, Palle Raabjerg, and Johannes Åman Pohjola. Higher-order psi-calculi. *Mathematical Structures in Computer Science*, 24, 4 2014.

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Name of applicant
Parrow, Joachim

Date of birth
561225-1453

Title of research programme

Appendix N

Budget and research resources

Vetenskapsrådet, Box 1035, SE-101 38 Stockholm, tel. +46 (0)8 546 44 000, vetenskapsradet@vr.se

Bilaga N: Budget

Lön har beräknats för två doktorander 80 % med månadslön enligt gängse lönestege för doktorander, ökad med 2.5% per år. Av de nu anställda doktoranderna kommer åtminstone en att disputera ungefär när projektet börjar, och en något år senare. Vi räknar alltså med en nyrekrytering under 2015 och en under 2016. Den doktorand som fortsätter (Johannes Åman Pohjola) avses finansieras av det sökta projeketet.

Handledningskapacitet finns redan i forskargruppen, som ägnar delar av sin fakultetsfinansierade forskning år detta område. Vi har för närvarande inga ytterligare externa projektmedel för detta projekt.

Lönekostnaden inkluderar lönebikostnader 49.4% och övriga indirekta kostnader 28.5%. Resekostnaden motiveras av att projektet har en hög grad av samarbete med andra forskargrupper i Europa.

Utrustning är främst laptops med lite kringutrustning.

Lokalkostnader är enligt schablon.

Monthly salary 1	20 480	20 480	21 520	22 720	24 000
Yearly cost	483 603	495 693	533 886	577 748	625 555
Monthly salary 2	24 000	20 480	20 480	21 520	22 720
Yearly cost	566 722	495 693	508 085	547 234	592 192
Other costs					
Computer costs	30 000	30 000	30 000	30 000	30 000
Premises (offices rooms)	40 000	40 000	40 000	40 000	40 000
Travel costs	50 000	50 000	50 000	50 000	50 000
Travel costs (incl. indir. costs)	64 250	64 250	64 250	64 250	64 250
COSTS for VR application	**1 184 575**	**1 125 636**	**1 176 222**	**1 259 232**	**1 351 997**

VETENSKAPSRÅDET
THE SWEDISH RESEARCH COUNCIL

Kod
2014-20305-115114-19

Dnr
2014 -

Name of applicant
Parrow, Joachim

Date of birth
561225-1453

Reg date
2014-04-08 14:58:53

Project title
A Framework for Parallel Programming Specifications

NT

Research grants NT April 9, 2014
Project Research Grant
*Övrigt generellt
Uppsala universitet

Appendix S

Signatures

NT-2

2015	2016	2017	2018	2019	
1185	1126	1176	1259	1352	2015-01-01 -- 2019-12-31

Applicant | **Date**

Head of department at host University | **Clarifi cation of signature** | **Telephone**

A signature on the application is required not only from the applicant but also from the authorised representative of the administrating organisation/institution or equivalent (normally the head of the department or establishment where the research is to be conducted). The signature confirms that the department can accommodate the proposed research, position or equipment; that the costing in the application is approved for the department's part, that any proposed experimentation on human or animal subjects has been reported, and that the applicant has reported any secondary occupations and commercial ties (s)he may have, and nothing inconsistent with good research practice has thereby emerged. The applicant must have discussed these conditions with the representative of the administrating organisation/institution or equivalent before the latter approves and signs the application.

NOTE the exceptions!
Please note that a different procedure applies for the following types of grants:

- Infrastructure- Operation Grants
- Infrastructure- Grants for Expensive Equipment
- Infrastructure- Grants for Large Databases
- Framework Grant SIMSAM
- International Recruitment

For these types of grants the vice-chancellor, instead of the head of department, must sign Appendix S. Thereby (s)he confirms the commitments of the administrative organization. The details are specified in the Call for Proposals, published at www.vr.se.

Vetenskapsrådets noteringar
Kod

2014-20305-115114-19

Vetenskapsrådet, Box 1035, SE-101 38 Stockholm, tel. +46 (0)8 546 44 000, vetenskapsradet@vr.se