Review report on EPSRC Grant GR/L62290/01 (01/07/98 – 30/06/01)

Calculi for Interactive Systems: Theory and Experiment

Philippa Gardner, Robin Milner and Peter Sewell

1 Overview

The objectives of the project were, briefly: (1) To study and design a foundational calculus and prototype programming language, based on the π-calculus, for describing and analysing migratory distributed systems; (2) To further develop the theory of action calculi, a graphical formalisation of interactive systems, to compare disciplines and study common concepts; (3) To test this framework upon existing and new calculi.

The proposal was organised in two strands, specific and general, essentially (1) and (2) above. The work has proceeded in this manner, with continual exchange of ideas on topics shared between the strands, such as: the behavioural equivalence of processes; the interplay between *connectivity* ('wiring', channels or naming) and *locality* (machine, scope of name, ambient) in modelling migratory systems; implementation of a calculus by an abstract machine; labelled transition systems; and many others. The next two sections of this report give details of the work done on the two main strands, named exactly as in the proposal.

The principal successes of the project have been in two regions:

- In the design, analysis and implementation of the calculus *Nomadic π-calculus* and associated programming language *Nomadic Pict* for migratory systems. These make it possible—we believe for the first time— to express both the specification and the implementation of protocols for migratory systems within the same frame, and thus to verify them formally, employing novel techniques and semantic concepts. Complementary work has been done on modularity and versioning, secure encapsulation and failure semantics.
- In theoretical models, using new concepts such as *explicit fusion* and *bigraph*, for the behaviour of a very wide range of interactive systems—including migratory ones such as Nomadic π-calculus. Explicit fusions improve both the behavioural theory of certain calculi and the algebraic theory of graphical models; bigraph theory, supported by new categorical notions, yields uniform behavioural congruence results which have been sought for several years. The categorical foundations of bisimilarity have also been advanced.

Thus good progress has been made on the design, mathematical modelling and formal verification of interactive systems—especially those which are distributed and migratory. While the technology of these systems is still advancing it is vital to coordinate the advance of theoretical models and verification methodology, as the project has done. The authors believe that the outcome of this project establishes their research programme in a prominent position internationally in this difficult topic. It also strengthens the claim that the family of calculi grouped around the π-calculus contributes centrally to the understanding of mobile systems.

PhD Dissertations Four PhD Dissertations have been written in close association with the work of the project. Those by James Leifer, Asis Unyapoth and Pawel Wojciechowski have been awarded PhD, and Lucian Wischik will submit by November 2001.

2 The specific strand: Distributed migratory systems

Our work in the specific strand has addressed four aspects of distributed computation, described below under **Mobility**, **Versioning**, **Secure encapsulation**, and **Failure**. The work on mobility deals specifically with migratory systems, while the other three are more broadly relevant to migratory and non-migratory distributed systems alike. For each, we have designed calculi that capture the relevant phenomena, equipped them with operational semantics, and used them for analysis and design. For the first and last aspects we have corresponding programming language implementations, closely paralleling the theoretical work.

Mobility: Nomadic Pict Our work on distributed migratory systems took on a specific focus early in the project: that of understanding how to express, design, and reason about the distributed infrastructure that is required for mobile computations to interact. High-level programming often requires *location independent* communication primitives, for example as provided by the Distributed Join Calculus of Fournet et al, and the the dpi calculus of Sewell [46] (introduced to study type-theoretic control of co-located resources). The underlying networks, however, provide *location dependent* primitives. Implementing the delivery of location independent messages to migrating computations is therefore challenging: it requires highly-concurrent distributed algorithms, and there is a complex design space.

In [33, 34] Sewell, Wojciechowski and Pierce designed the *Nomadic π-calculus* for expressing these algorithms. It has a low-level subcalculus, with migration and location-dependent communication, that has a straightforward distributed implementation. The full calculus extends this with location-independent messaging, enabling implementation algorithms to be expressed as an encoding of the full calculus into the low-level subcalculus. Our experience suggests that the latter provides a good level of abstraction, enabling details of concurrency and locking to be seen clearly.

Based on this, Wojciechowski and Sewell designed and implemented the *Nomadic Pict* programming language, and explored the algorithm design space [50, 51, 53] (the last of which was completed after Wojciechowski took up a position at EPFL). Full discussion can be found in Wojciechowski's PhD thesis [52]. Our implementation is available electronically [54].

Meanwhile, Unyapoth and Sewell studied the semantic definition and proof techniques for Nomadic Pict, demonstrated with a correctness proof for one such algorithm. This required new semantic theory – especially labelled transitions and *translocating equivalences* for reasoning about the behaviour of migrating agents and low-level messages, that can fail if their target agent has moved away. The work was reported in [47], with full details in Unyapoth's PhD thesis [48].

Versioning The design of Nomadic Pict infrastructures drew attention to the problem of distributed versioning – a large distributed system typically has many different versions of software coexisting and interacting with each other. In [35, 36] Sewell introduced a sound model of separate compilation and execution of modular programs that interact by communicating values (possibly of abstract types), exploring how one can statically enforce coherence between different versions.

This work made use of type-level **new**-binders, scoping over system configurations that include compiled object code. It smoothly integrates these with a singleton-kind type system, as used by Harper and others for expressing type sharing in ML-style module systems.

Secure encapsulation In a wide-area network one must deal with potentially-malicious software components. Work of Sewell and Vitek developed precise notions of what it means to securely encapsulate an untrusted component with a *wrapper*, in terms of a *Box-π* calculus for describing components and wrappers [37, 38].

There is a tension between providing liberal communication primitives, which permit easy communication between components, and strong encapsulation primitives, which can enforce limitations on the communications of untrusted components. Box-π is an experiment at the latter end – it allows a wrapper complete control over the interactions between an encapsulated component and the outside world. Implementing wrappers is then straightforward enough – the main focus of the work was on the precise specification of what properties they guarantee. To express these the calculus was equipped with a lightweight causal semantics. For proving them, a type system was developed for controlling causal flow in Box-π, enabling simple proofs of information flow properties as corollaries of a subject reduction theorem [39, 40, 41].

Failure The Nomadic π semantics addressed only a limited form of failure, of messages being lost due to their target agent not being present on the correct site (though some of our algorithms go beyond, dealing with disconnection). In work begun near the end of this grant (and continuing, funded by EPSRC grant GRN24872), Serjantov, Sewell and Wansbrough are studying more accurate failure semantics.

In [30, 31] they gave a model for the sockets interface to the standard UDP and ICMP protocols, choosing this level of abstraction as it is ubiquitous and provides a clear view of the failure behaviour. This has required a novel *experimental semantics* approach: experimentally determining the behaviour of particular OS implementations of the sockets code. It includes also a semantics for a fragment of OCaml and an implementation of the exact socket operations that are modelled, making it possible to reason about distributed programs that are written in a general-purpose programming language and use standard communication primitives. It marries (albeit with a shotgun wedding?) the now well-developed concurrency theory and actual network programming.

Sewell leads a new EPSRC project, *Wide-area programming: Language, Semantics and Infrastructure Design* (GRN24872) concerned with application of theoretical models to real-world distributed systems.

3 The general strand: Action calculi and their dynamics

For this strand, the proposal focused on a general understanding of the dynamics of *action calculi*, an evolving graphical process framework in which to unite calculi for describing concurrent distributed systems. The framework has two key concepts: agents can be collected *locally* using controls (analogous to ambients and the wrappers of Box-π), and agents can be cross-referenced (*connected*, wired together). Our research has indeed evolved in ways that we did not predict, motivated both by specific calculi – which are much more numerous than when the project began – and by a concern to identify the correct primitive notions. In particular, we have explored different notions of *locality* and *connectivity*, both for individual calculi and for general models, and have found that our choices have strong impact upon the general theory.

In our first subsection, **Action calculi**, we explain how action calculi as previously defined have led to the developments reported in the ensuing subsections, which are as follows: the **Explicit fusion calculus**—a calculus introducing explicit fusions, a novel naming construct for connecting nodes in graphs; **Reactive systems**—an abstract way of uniformly describing labelled transition systems and guaranteeing behavioural congruences under certain abstract conditions; **Process graphs**—which emphasise the use of naming constructs to 'glue' graphs together; and **Bigraphical systems**— which evolved from process graphs and from

the decision to treat locality as orthogonal to connectivity, and which meet the abstract conditions identified for reactive systems. Further subsections describe additional work on **Categorical models of processes**, and on **Tools and abstract machines** associated with our theoretical work.

Action calculi In previous work (published or invited during the grant period and cited here because of its relevance to the project's progress), Gardner with co-authors [1] connected action calculi with known concepts from type theory and category theory, to provide a better understanding of their character. This work led to the dissertation of her student Hasegawa, given a Distinguished Dissertation award [19]. She clarified the action calculus concept in two further papers [11, 12], in particular providing a deeper understanding of the transition from the π-calculus to the action calculus framework. This thread of work led both to the *explicit fusions* and to the *process graphs* discussed below.

Action calculi also prompted the abstract study of *reactive systems*, which in turn posed a challenge to action calculi—namely to demonstrate that they satisfy the conditions required for the behavioural congruence theorem. They do in a wide range of cases. This was shown for so-called *shallow* action calculi by Cattani, Leifer and Milner [3], and in more detail in Leifer's Dissertation [20]. Extension to *reflexive* action calculi was begun by Leifer and Milner [20, 22], and to *deep* calculi (i.e. with nested localities) by Milner in unpublished work; however, proving the conditions of the congruence theorem got harder. A liberation from this toil has been found in the *bigraphical systems* discussed below.

Explicit fusion calculus Gardner and her Ph.D student Wischik have developed the *explicit fusion calculus*, with a new notion of process interaction motivated by the need to simplify the treatment of names in graphical process models. In the π-calculus the primitive concepts are *processes* and *(channel) names* x, y, and its primitive dynamic step is the passing of names between processes. In the explicit fusion calculus, the primitive dynamic step is the symmetric 'fusing' of names [17, 16, 18, 49]. An explicit fusion (or alias) $x = y$ is a process existing concurrently a system, enabling the two names to be used interchangeably. This provides a direct way of merging named nodes in graphs, and removes a constraint present in action calculi as originally defined. The notion leads to a "small-step" account of the name-substitution occurring in the π-calculus, and in the *implicit* fusions of Parrow and Victor's fusion calculus and Fu's χ-calculus.

In his Dissertation [49], Wischik explores several equivalent bisimulations for the explicit fusion calculus (barbed congruence, ground congruence, an efficient characterisation), which provide a simpler story than the corresponding bisimulations for the implicit fusion calculi. The embedding of the fusion calculus in the explicit fusion calculus is fully abstract with respect to bisimulation; the embedding of the π-calculus is sound and complete with respect to open bisimulation. He also introduces a fully distributed abstract machine for the calculus, in which the explicit fusions play an essential role, as described below.

Reactive systems We unite process calculi and languages under an abstract categorical notion of *reactive system*. This is simply a family of *agents* $a, b, \ldots$, a category of *contexts* $C, D, \ldots$ with which to build agents, e.g. $C \circ a$, a set of *reaction rules*, and a subcategory of *active contexts* saying where reaction may occur. In this abstract setting we look for a uniform way of *deriving*, for a wide range of reactive systems, the labelled transition systems which are often taken as basic in specific process calculi. This leads to a uniform account of behavioural equivalences and preorders over agents; a criterion of success is that the derivation forces these behavioural relations to be congruences – i.e. preserved by context.

As a first step, Sewell [45] was able uniformly to derive satisfactory context-labelled transitions for parametric term rewrite systems with parallel composition and active/inactive controls, and showed bisimilarity

to be a congruence. It remained a problem to do it for reactive systems dealing with connectivity. Leifer and Milner [21, 20] achieved a uniform derivation of transitions yielding congruential behaviour relations (especially trace and failures preorders, and strong and weak bisimilarity) for all reactive systems satisfying two abstract conditions. We call this result the congruence theorem; it rests upon a simple categorical notion –the *relative pushout (RPO)*– apparently previously neglected by mathematicians and those who apply category theory. The properties needed of RPOs are derived in Leifer's PhD dissertation [20].

Process graphs In practical systems (e.g. the worldwide web) communication links cross locational boundaries. Even in the λ-calculus, free names can occur within the body of a λ-expression; this is a simple instance of the same general phenomenon. Though action calculi can model this boundary-crossing to a large extent, they do it indirectly and the theory becomes complex. By contrast, the process graphs of Gardner and the bigraphs of Milner (described below) adopt a more direct approach to connectivity and a simpler treatment of locality.

The focus of process graphs is the use of naming constructs to connect directly between locations [13]. In fact, the variation using explicit fusions (fusion graphs [18]) evolved from Gardner and Wischik's symmetric action calculi [15], which conservatively extend reflexive action calculi. A specific application of process graphs is to model semi-structured data. With Cardelli and Ghelli, Gardner is exploring a spatial logic for reasoning about labelled directed graphs [2], adapting Cardelli and Gordon's ambient logic for reasoning about trees. The logic will be used to design a query language for analysing and manipulating semi-structured data, with the naming constructs used to 'glue' graphs together. The aim is to provide a smooth integration between the data model, the logic and the query language, as found in relational databases but not yet in languages associated with XML.

Bigraphical systems The innovation of the *bigraphical* model [27, 28, 25, 26], compared with action calculi, is to treat connectivity as orthogonal to locality. A *bigraph* consists of a *topograph* for locality and a *monograph* for connectivity, and the two need only to agree on their control nodes, not in the structure that they impose on those nodes. This new model not only increases expressive power (e.g. the *redex* of rule for remote interaction can consist of parts at widely separate locations), but also makes it much easier to establish labelled transitions and behavioural congruences for a wide range of deep bigraphical systems, including versions of the ambient calculus and distributed π-calculus and combinations of them.

Gardner's fusion graphs contributed strongly to this development; the idea of fusion is essential to bigraphs. Sewell's work on multi-hole contexts [45] is guiding the development, especially in the treatment of distributed agents. In current work Milner and a PhD student are examining the transitions and congruences generated for various π-calculi, for comparison with the original theory of the calculus.

Categorical models of processes Cattani and Sewell investigated the mathematical structure of interleaving and causal models for the π-calculus, introducing syntax-free notions of *Indexed Labelled Transition System* and *Indexed Labelled Asynchronous Transition System*, that smoothly generalise models for CCS-like calculi, and relating denotational and operational semantics [5, 6]. These functor-category models make explicit certain renaming-structure that one would expect to arise from our general work on labels.

Cattani completed work with Winskel on presheaf models for CCS-like languages, showing that open-map bisimulations within a range of presheaf models are congruences for a general process language, in which CCS and related languages are easily encoded. The abstract results were applied to show that hereditary history-preserving bisimulation is a congruence for CCS-like languages extended with a refinement operator

[7]. Cattani also studied (with Fiore, Power and Winskel) the categorical treatment of (weak) bisimulation, in [10], [8] and [9], and gave a representation result for free cocompletions [29].

Tools and machines The general (or theoretical) strand of the project has given rise to exploratory abstract machines and implementations.

(1) Together with Cosimo Laneve of Bologna, Gardner and Wischik [49] investigate a fully distributed abstract machine for the explicit fusion calculus, and hence for the π-calculus and the fusion calculus, which removes the need for multi-way synchronisation. This contrasts with previous implementations, such as the language Pict of Pierce and Turner which runs on a single processor, Facile which relies heavily on synchronisation, and JoCaml and Nomadic Pict which adopt different communication primitives to deal with synchronisation. Wischik shows that the explicit fusion calculus and the π-calculus embed well in the machine [49], and proves correctness results for these embeddings. A prototype single-processor implementation is available online. Wischik will be developing a fully distributed implementation with Laneve in Bologna from January 2002.

(2) With Gardner, Norrish and Serjantov implemented a specification tool for action graphs, which allows the user to switch naturally between the syntactic and graphical presentations [14]. His algorithm is based on a notion of graph embedding, used also in the theoretical characterisation of action graph embeddings [3]. This work, though unpublished, has led to two undergraduate projects and lies on the critical path to future graphical tools to assist analysis of the graphical process models we are developing.

(3) In work for his PhD dissertation, Alistair Turnbull has devised a linear intermediate language –or abstract machine– aimed at the implementation of multithreaded languages such as Java. The machine, called MIN, has a graphical presentation. Turnbull has proved congruence for weak bisimilarity of programs, which will serve as a basis for compile-time optimisations.

4 Wider contribution

Gardner, Milner and Sewell play an active role internationally in the field of practically applicable models of interactive systems. **Gardner** holds an EPSRC Advanced Fellowship until 31st December 2001. She gave an invited presentation at MFPS 2000 in New York, and invited tutorials at Mathfit 1999 (London) and CONCUR 2000 (Pennsylvania), on explicit fusions and process frameworks. She was an invited participant at the Dagstuhl meeting on the Foundations of Semi-structured Data, September 2001. **Milner** led the team which originated the π-calculus, and published the first text book on the topic in 1999; he recently gave invited talks on bigraphical reactive systems at leading conferences including POPL 2001 (London), CONCUR 2001 (Aalborg) and Petri Nets 2001 (Newcastle). **Sewell** co-chaired the 4th International Workshop on High-Level Concurrent Languages (with PLI 2000, Montreal) and the 5th Mobile Object Systems Workshop (with ECOOP99, Lisbon). He gave an invited tutorial at Mathfit 1999 (London) and contributed a chapter on π-calculus to a volume on *Formal Methods for Distributed Processing*, to be published by CUP [43]. He took up a Royal Society University Research Fellowship in October 1999.

Review report on EPSRC Grant GR/L62290/01
(01/07/98 – 30/06/01)

List of Publications

Authors in italics are those not on the project as investigator, research associate or PhD student. *Luca Cardelli* was an industrial co-author, from Microsoft Research. International co-authors were *Giorgio Ghelli* [2] from Italy, *Benjamin Pierce* [33, 34] and *Jan Vitek* [37, 38, 39, 40, 41] from the USA, and *Glynn Winskel*[7, 8, 9, 10] from Denmark. (Professor Winskel joined our Laboratory in 2000.)

Three papers chosen as most significant are [17], [21] and [47].

Items are annotated as follows: **J** – Journal; **C** – Conference; **I** – Invited conference; **B** – Book or chapter; **T** – Technical Report; **D** – PhD Dissertation; **S** – Software.

Status of papers: **ref** – refereed; ***** – in preparation; ****** – submitted; ******* – invited submission.

References

[1] *A. Barber*, P. Gardner, *M. Hasegawa* and *G. Plotkin*. From action calculi to linear logic. Springer-Verlag, LNCS 1414, *Selected papers from the European Association of Computer Science Logic*, 1998.

ref J

[2] *L. Cardelli*, P.A. Gardner and *G. Ghelli*. A spatial logic for reasoning about graphs. Based on invited presentation at Dagstuhl Workshop, September 2001. To be submitted in 2001. *** C**

[3] G.L. Cattani, J.J. Leifer, and R. Milner. Contexts and embeddings for closed shallow action graphs. University of Cambridge Computer Laboratory, Technical Report 496, 2000. Available at `http://pauillac.inria.fr/~leifer` . **T**

[4] G.L. Cattani, J.J. Leifer, and R. Milner. Contexts and embeddings for closed shallow action graphs. Invited journal submission to Theoretical Computer Science, in association with MFCS, 2000. ***** J**

[5] Gian Luca Cattani and Peter Sewell. Models for name-passing processes: Interleaving and causal (extended abstract). In *Proceedings of LICS 2000: the 15th IEEE Symposium on Logic in Computer Science (Santa Barbara)*, pages 322–333, June 2000. **ref C**

[6] Gian Luca Cattani and Peter Sewell. Models for name-passing processes: Interleaving and causal. Technical Report 505, Computer Laboratory, University of Cambridge, 2000. 42pp. **T**

[7] Gian Luca Cattani and *Glynn Winskel*. Presheaf models for CCS-like languages. Technical Report 477, Cambridge University Computer Laboratory, 1999. To appear in *Theoretical Computer Science*. **T**,**ref J**

[8] Gian Luca Cattani, *A. John Power*, and *Glynn Winskel*. A categorical axiomatics for bisimulation. In *Proceedings of the 9th International Conference on Concurrency Theory, CONCUR '98*, volume 1466 of *Lecture Notes in Computer Science*, pages 581–596. Springer-Verlag, 1998. **ref C**

[9] Gian Luca Cattani, *A. John Power*, and *Glynn Winskel*. Towards a categorical axiomatics of bisimulation. Submitted, 1999. **** J**

[10] *Marcelo Fiore*, Gian Luca Cattani, and *Glynn Winskel*. Weak bisimulation and open maps. In *LICS '99, Proceedings of the Fourteenth Annual IEEE Symposium on Logic in Computer Science*, pages 67–76. IEEE Computer Society Press, 1999. **ref C**

[11] P. Gardner. Closed action calculi. *Theoretical Computer Science*, volume 228, 1999. Invited paper in association with the conference on Mathematical Foundations in Programming Semantics. **ref J**

[12] P. Gardner. From process calculi to process frameworks. Springer-Verlag LNCS 1877, *International Conference on Concurrency Theory (CONCUR)*, 2000. pp69–88. **ref I**

[13] P. Gardner. A compositional description of graphs using names. Journal paper to be submitted in September, 2001. *** J**

[14] P. Gardner, M. Norrish and A. Serjantov. Specification tool for action graphs. Unpublished manuscript, 2000. Available at `http://www.doc.ic.ac.uk/~pg/` .

[15] P. Gardner and L. Wischik. Symmetric action calculi. Unpublished manuscript, 1998.

[16] P. Gardner and L. Wischik. A process framework based on the π_F-calculus. EXPRESS, Elsevier Science Publishers, 1999. **ref C**

[17] P. Gardner and L. Wischik. Explicit fusions. *Mathematical Foundations of Computer Science (MFCS)*, LNCS 1893, 2000. pp373–383. **ref C**

[18] P. Gardner and L. Wischik. Explicit fusions. Invited journal submission to Theoretical Computer Science, in association with MFCS, 2000. ***** J**

[19] *M. Hasegawa*. *Models of sharing graphs: a categorical semantics of let and letrec*. Springer, Distinguished Dissertation Series, ISBN 1-85233-145-3 , 1999. **ref B**

[20] J.J. Leifer. *Operational congruences for reactive systems*. PhD Dissertation, University of Cambridge Computer Laboratory, 2001. **D**

[21] J.J. Leifer, and R. Milner. Deriving bisimulation congruences for reactive systems. *CONCUR 2000, 11th International Conference on Concurrency theory*, pp243–258, 2000.
Available at `http://pauillac.inria.fr/~leifer` . **ref C**

[22] J.J. Leifer and R. Milner. Shallow linear action graphs and their embeddings. University of Cambridge Computer Laboratory, Technical Report 508, 2000. 16pp. **T**

[23] J.J. Leifer and R. Milner. Shallow linear action graphs and their embeddings. To appear in a festshrift in honour of R.M. Burstall, 2001. **ref B**

[24] R. Milner. *Communicating and mobile systems: the pi calculus*. Cambridge University Press, 1999. Reprinted 2001. **B**

[25] R. Milner. Bigraphical reactive systems. Invited paper, *CONCUR 2000, 12th International Conference on Concurrency theory*, LNCS 2154, pp16–35. Springer-Verlag, 2001. **I**

[26] R. Milner. Bigraphical reactive systems: basic theory. University of Cambridge Computer Laboratory, Technical Report, September 2001. 81pp. Available at `http://www.cl.cam.ac.uk/users/rm135/` . *** T**

[27] R. Milner. The flux of interaction. Abstract of invited lecture at *22nd International Conference on Applications and Theory of Petri Nets*, ed Colom and Koutny, LNCS 2075, pp19–22. Springer-Verlag, 2001. **I**

[28] R. Milner. Computational Flux. Abstract of invited lecture at *POPL 2001: 28th ACM SIGPLAN-SIGACT Symposium on principles of Programming Languages*, pp220–221. 2001. **I**

[29] *A. John Power*, Gian Luca Cattani, and *Glynn Winskel*. A representation result for free cocompletions. *Journal of Pure and Applied Algebra*, 151(3):273–286, 2000. **ref J**

[30] Andrei Serjantov, Peter Sewell, and Keith Wansbrough. The UDP calculus: Rigorous semantics for real networking. Technical Report 515, Computer Laboratory, University of Cambridge, July 2001. iv+70pp. **T**

[31] Andrei Serjantov, Peter Sewell, and Keith Wansbrough. The UDP calculus: Rigorous semantics for real networking. In *TACS'01*, 2001. To appear. **ref C**

[32] Peter Sewell, Paweł T. Wojciechowski, and *Benjamin C. Pierce*. Location independence for mobile agents. In *Proceedings of the Workshop on Internet Programming Languages (Chicago)*, May 1998. **ref C**

[33] Peter Sewell, Paweł T. Wojciechowski, and *Benjamin C. Pierce*. Location-independent communication for mobile agents: a two-level architecture. Technical Report 462, Computer Laboratory, University of Cambridge, April 1999. 31pp. **T**

[34] Peter Sewell, Paweł T. Wojciechowski, and *Benjamin C. Pierce*. Location-independent communication for mobile agents: a two-level architecture. In *Internet Programming Languages, LNCS 1686*, pages 1–31. Springer-Verlag, October 1999. **ref B**

[35] Peter Sewell. Modules, abstract types, and distributed versioning. Technical Report 506, University of Cambridge, 2000. 46pp. **T**

[36] Peter Sewell. Modules, abstract types, and distributed versioning. In *Proceedings of POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (London)*, pages 236–247, January 2001. **ref C**

[37] Peter Sewell and *Jan Vitek*. Secure composition of insecure components. Technical Report 463, Computer Laboratory, University of Cambridge, April 1999. 44pp. **T**

[38] Peter Sewell and *Jan Vitek*. Secure composition of insecure components. In *Proceedings of CSFW 99: The 12th IEEE Computer Security Foundations Workshop (Mordano, Italy)*, pages 136–150. IEEE Computer Society, June 1999. **ref C**

[39] Peter Sewell and *Jan Vitek*. Secure composition of untrusted code: Wrappers and causality types. Technical Report 478, Computer Laboratory, University of Cambridge, November 1999. 36pp. **T**

[40] Peter Sewell and *Jan Vitek*. Secure composition of untrusted code: Wrappers and causality types. In *Proceedings of CSFW 00: The 13th IEEE Computer Security Foundations Workshop (Cambridge)*, pages 269–284. IEEE Computer Society, July 2000. **ref C**

[41] Peter Sewell and *Jan Vitek*. Secure composition of untrusted code: Box π wrappers and causality types. *Journal of Computer Security*. Invited submission for a CSFW00 special issue. To appear. **ref J**

[42] Peter Sewell. Applied π – a brief tutorial. Technical Report 498, Computer Laboratory, University of Cambridge, August 2000. 65pp. . **T**

[43] Peter Sewell. Chapter on Pi Calculus in the book Formal Methods for Distributed Processing, An Object Oriented Approach, edited by Howard Bowman and John Derrick. To be published by CUP. **B**

[44] Peter Sewell. From rewrite rules to bisimulation congruences. In *Proceedings of CONCUR '98: Concurrency Theory (Nice). LNCS 1466*, pages 269–284. Springer-Verlag, September 1998. **ref C**

[45] Peter Sewell. From rewrite rules to bisimulation congruences. *Theoretical Computer Science*, 2001. Invited submission for a CONCUR 98 special issue. To appear in Volume 272/1–2. **ref J**

[46] Peter Sewell. Global/local subtyping and capability inference for a distributed π-calculus. In *Proceedings of ICALP '98: International Colloquium on Automata, Languages and Programming (Aalborg). LNCS 1443*, pages 695–706. Springer-Verlag, July 1998. **ref C**

[47] Asis Unyapoth and Peter Sewell. Nomadic Pict: Correct communication infrastructure for mobile computation. In *Proceedings of POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (London)*, pages 116–127, January 2001. **ref C**

[48] Asis Unyapoth. *Nomadic π-Calculi: Expressing and Verifying Infrastructure for Mobile Computation*. PhD thesis, University of Cambridge, June 2001. Also appeared as Technical Report 514, Computer Laboratory, University of Cambridge, June 2001. **D**

[49] L. Wischik, Explicit fusions: theory and implementation, Ph.D thesis to be submitted in September 2001. *** D**

[50] Paweł T. Wojciechowski and Peter Sewell. Nomadic Pict: Language and infrastructure design for mobile agents. In *Proceedings of ASA/MA 99: Agent Systems and Applications/Mobile Agents (Palm Springs)*, pages 2–12, October 1999. Best paper award. **ref C**

[51] Paweł T. Wojciechowski and Peter Sewell. Nomadic Pict: Language and infrastructure design for mobile agents. *IEEE Concurrency*, 8(2):42–52, April–June 2000. Invited submission for ASA/MA 99. ***** J**

[52] Paweł T. Wojciechowski. *Nomadic Pict: Language and Infrastructure Design for Mobile Computation*. PhD thesis, University of Cambridge, 2000. Also appeared as Technical Report 492, Computer Laboratory, University of Cambridge, June 2000. **D**

[53] Paweł T. Wojciechowski. Algorithms for location-independent communication between mobile agents. In *In the Proceedings of AISB'01 Symposium on Software mobility and adaptive behaviour*, 2001. Also appeared as Technical Report DSC-2001-013, Laboratoire de Systèmes d'Exploitation, EPFL, March 2001. **ref C**

[54] Paweł T. Wojciechowski. Nomadic Pict. programming language for global computation (implementation and tutorial). version 1.0, December 2000.
Available at `http://lsewww.epfl.ch/~pawel/nomadicpict.html`. **S**