G. Rozenberg (Ed.)

Advances in Petri Nets 1992

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Series Editors

Gerhard Goos
Universität Karlsruhe
Postfach 69 80
Vincenz-Priessnitz-Straße 1
W-7500 Karlsruhe, FRG

Juris Hartmanis
Department of Computer Science
Cornell University
5149 Upson Hall
Ithaca, NY 14853, USA

Volume Editor

Grzegorz Rozenberg
Department of Computer Science, Leiden University
P. O. Box 9512, 2300 RA Leiden, The Netherlands

CR Subject Classification (1991): F.1-3, C.1-2, D.4, I.6

ISBN 3-540-55610-9 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-55610-9 Springer-Verlag New York Berlin Heidelberg

Printed in Germany

Typesetting: Camera ready by author/editor
Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
45/3140-543210 - Printed on acid-free paper

Preface

The main aims of the series of volumes "Advances in Petri Nets" are:
(1) to present to the "outside" scientific community a fair picture of recent advances in the area of Petri nets, and
(2) to encourage those interested in the applications and the theory of concurrent systems to take a closer look at Petri nets and then join the group of researchers working in this fascinating and challenging area.

The ESPRIT Basic Research Action DEMON (DEsign Methods based On Nets) has been a focus of developments within the Petri nets community for the last three years. The research done within DEMON spans many areas of Petri nets; it has certainly helped to consolidate and unify the knowledge about Petri nets, and it has led to many new developments. Hence, it fits the aims of "Advances" to have a special volume presenting some of the achievements of DEMON.

The papers presented in this volume have been selected from 24 papers submitted to "Advances" in response to a call for papers directed to participants of DEMON. The papers went through the refereeing process, and those that were accepted appear in this volume in a revised form.

This volume contains technical contributions giving insight into a number of major achievements of the DEMON project. It also contains four survey papers covering important research areas: basic net models and modular net classes, structural techniques in performance analysis of Petri net models, recognizable languages of infinite traces, and equivalence notions for net based systems. These surveys certainly help the reader to get an overview of a broad range of research taking place in the area of Petri nets and related models of concurrent systems. The volume begins with a description of DEMON given by its coordinator E. Best.

I want to thank E. Best for his help in preparing this volume. Special thanks go to the referees of the papers in this volume who very often are responsible for considerable improvements. The referees were: L. Aceto, M. Ajmone Marsan, E. Astesiano, J. Baeten, J. Billington, W. Brauer, M. Broy, A. Corradini, F. De Cindio, Ph. Darondeau, P. Degano, J. Desel, R. Devillers, C. Diamantini, V. Diekert, H. Ehrig, J. Esparza, U. Goltz, R. Gorrieri, J. Hall, A. Heise, R. Henderson, R. Hopkins, B. Keck, A. Kiehn, E. Kindler, M. Koutny, H.J. Kreowski, M. Kwiatkowska, M. Latteux, G. Mauri, A. Mazurkiewicz, M. Nielsen, L. Petrucci, M. Pinna, A. Poigné, L. Pomello, W. Reisig, G. Reggio, B. Rozoy, C. Simone, E. Smith, P. Starke, P. Taylor, W. Thomas, W. Vogler, K. Voss, D.N. Yankelevitch, W. Zuberek. The editor is also indebted to Mrs. M. Boon-van der Nat and Dr. A. Deutz for their help in the preparation of this volume.

Leiden, April 1992

G. Rozenberg
Editor

Table of Contents

Technical Contributions

Surveys

Esprit Basic Research Action 3148
(Design Methods Based on Nets)
— Aims, Scope and Achievements —

Quote of the Initial Statement of DEMON (1989)

The overall aim of the Action is to lay the foundation for the eventual development of a design calculus for concurrent systems based on the principles of Petri net theory. These principles concern basic characteristics of concurrent systems, such as decentralisation, distinctions between concurrency and nondeterminism, and an integrated view of the interplay between the states and the transitions of a system. (...) As the central net theoretic foundation for the envisaged calculus, structuring and modularity techniques are to be developed which include abstraction, refinement, composition, preservation of properties both within and across levels of abstraction, appropriate notions of equivalence and implementation, and analysis techniques, addressing the spectrum of techniques needing to be covered by a design calculus. The development of these techniques will involve original research, but shall also be guided by, and have an impact on, experience with the design of concurrent systems and existing other approaches to the modelling of concurrent systems. It shall be accompanied by the evaluation of case studies. Our proposal is thus not only meant as a further development located purely within net theory, but also as a widening of its scope as well as an exploitation of its generality, since we propose to take into account insights from complementary approaches in an important way. (...) Having identified the importance of making progress in investigating and developing the above issues (in particular, modularity), the Action proposes to concentrate on the following work:

- *The development of one or more classes of structured Petri nets which feature abstraction, refinement and composition operators, along with appropriate notions of equivalence, logic / algebraic calculi and analysis techniques. Such classes of nets would form the core of an eventual design calculus, bridging the gap which presently exists between applications of Petri nets as a non-formal graphical tool in the early stages of the design process, and applications of analytical techniques used in the final stages of the design process.*
- *The careful evaluation of the relationship to various complementary approaches aimed at similar goals, such as CSP (...), COSY, CCS, trace theory and event structures, with the aim of ensuring translatability, compatibility, and the transferral of useful techniques in both directions. (...)*
- *The investigation of the Petri net semantics of existing programming languages such as Occam, along with the transferral of techniques as described above. This investigation is done with two aims in mind: (i) to make explicit the connection to the programming of large and complex systems, and (ii) to study the possible definition of a Petri net based programming notation to provide a convenient way of expressing Petri net implementations.*
- *The study of a number of test cases to support the proposed conceptual techniques.*
- *The study of a selected number of existing design methods such as abstract data types, statecharts and logic formalisms. This is done with the aim of studying what new aspects are introduced in such methods by the consideration of true concurrency semantics; how they can be related to the Petri net framework to be developed; and how the resulting methods can be applied fruitfully in actual design. (...)*
- *The study of a number of specific and important issues which are of great practical importance but not yet theoretically fully understood. These concepts include priorities (Occam, Ada), exception handling (Ada), the compositionality aspects of buffered communication and dynamic system structure. The aim of this investigation is not just to study the semantics of these concepts in terms of the structured net classes, but also, conversely, to obtain criteria by which the usefulness of these classes can be evaluated.*

The above work of the Action will be structured into two main strands of activity, covering respectively the central Petri net theory work and the work on related areas, but strongly interrelated both in time and in contents. The first strand consists of a common effort to define new classes of modular Petri nets which are provided with means for composition, abstraction and refinement, thus establishing a common framework. In the second strand, more specialised work will be done both as motivation and input for work on the first strand, or as a result and application of it.
(End of quote)

In line with this work programme, the first, so-called 'core' strand of DEMON was intended to carry out Petri net based research that would lead to the required net classes, equivalence relations and proof methods. Strand I was accordingly organised into three Working Groups, WGs 1, 2 and 3. The second, so-called 'supporting' strand was intended to carry out work that would interface with the needs of practice and with related models. Originally, this strand was organised into seven Working Groups, but it soon became apparent (and was instigated by DEMON's reviewers) that the combination of some of these WGs would be sensible. Presently, Strand II consists of three Working Groups, WGs 4, 5 and 6:

Strand I: Working Group 1: **Structured Classes of Nets.**
Working Group 2: **Equivalence and Simulation Relations.**
Working Group 3: **Algebra and Proof Methods.**

Strand II: Working Group 4: **Abstract Models.**
Working Group 5: **High Level Nets, Case Studies and Specification.**
Working Group 6: **Programming and Related Issues.**

The DEMON Consortium and its boundary conditions have had the following shape:

DEMON Partner	Country	Rôle
Gesellschaft für Mathematik und Datenverarbeitung	D	Coordinator
Université Paris-Sud	F	Partner
Groupe Bull, Paris	F	Partner
Università degli Studi di Milano	I	Partner
Technische Universität München	D	Partner
Rijksuniversiteit te Leiden	NL	Partner
Universität Passau	D	Partner
University of Newcastle upon Tyne	GB	Partner
Université Libre de Bruxelles	B	Partner
Universidad de Zaragoza	E	Partner
Universität Hildesheim	D	Subcontractor.

Start date: June 19, 1989
End date: March 18, 1992 **Budget: ECU 731 500 . -**

Result Summary

This section is organised WG-wise. Its purpose is to provide a reasonably complete picture of the results of DEMON. However, the reader will *not* find all of the topics addressed in this section to be covered by the papers contained in this volume; information or further reading is provided by the list of references in the next section.

In addition to its original tasks (and not contained in the above quotation), DEMON has intended to produce survey and tutorial papers on some of the topics that are important to its goals. This volume contains three of these survey, while two earlier surveys (on refinements in Petri nets, and on structure theory of free choice nets, respectively) can be found in the Advances in Petri Nets 1990 (Springer Lecture Notes in Computer Science Vol.483).

Working Group 1: Structured Classes of Nets

Approach

The task of WG-1 was the definition of structured classes of nets starting from 'primitive' nets. More precisely, the aim of the group was to obtain an algebra in the mathematical sense, i.e., a carrier set (of nets) together with a set of operations (on nets). The pursuit of this goal has been constrained from two sides. The requirement to provide compositional semantics on the elementary Petri net level has tended to restrain the operators under consideration. The requirement that DEMON's net classes should form a domain for the semantics of languages with data, data sharing, communication etc. has tended to demand more, and more powerful, operators.

The WG has solved this problem by adopting a two–level approach. On the lower level, a small set of primitive operators indigenous to Petri nets have been defined: adding a place or a transition, and removing a place or a transition. On the higher level, more complex operations have been defined which, on the one hand, are based on the primitive operations of the lower level, but, on the other hand, are distinctly oriented towards the requirements of Strand II. This algebra has been christened the 'Petri Box Calculus' (PBC). WG-6 has built a finite model based on a class of high level nets.

Important cooperation has been with Working Groups 2, 3, 4 and 6. WGs 2 and 4 have provided motivation and basic work on equivalences; for instance, these WGs have paved the way for the notion of transition refinement that is presently adopted in the Box algebra. Both WGs presently continue to provide the ground work for two important further developments: the adoption of a weaker notion of equality between Boxes, and the definition of a compositional partial order semantics. WG-6 has strongly influenced the particular shape of our operators which are, partly, novel. In particular, multi-label communication, recursion and iteration have been influenced by WG-6. Conversely, WG-6 has profitted from the Box algebra by being able to define a small programming notation (with more expressiveness than occam) and its compositional Petri net semantics. Moreover, WGs 4 and 6 have done three nontrivial case studies specifically based on the notation. WGs 3 and 6 have provided the structured operational semantics for the Box algebra which is the main vehicle used so far in proofs of case studies.

Main achievements

- A survey of existing notions of net classes.
- Categorical setting at the lower level.
- A powerful algebra at the higher level, in particular as regards communication.
- A wealth of structural algebraic laws.
- Fixpoint semantics for general recursion.
- Indigenous operational semantics (in cooperation with WGs 3 and 6).

Working Group 2: Equivalence and Simulation Relations

Approach

The task of Working Group 2 was the definition of equivalence and simulation notions for models of concurrency with special emphasis on Petri nets. Important questions to be addressed were the study of equivalences based on partial orders, as well as the study of the practical relevance of the defined notions. The Working Group has approached its task by sampling existing equivalence notions (from the large existing set) and producing two surveys. Moreover, the WG has investigated the principles which underly the fact that certain ways of strengthening non-congruences (or non-behaviour preserving equivalences) would yield congruences (or behaviour-preserving congruences), while others would not, and has found a number of generally applicable methods to create one's favourite behaviour-preservation equivalence. Moreover, the WG has selected a few candidate congruences which are ready to be transported to the Box algebra framework. The transportation appears to be possible in a smooth way for two reasons: First, both WGs have based their investigations on labelled (rather than unlabelled) nets. Second, some preconditions of WG-2 theorems are actually theorems of WG-1.

In proving the desired congruence and behaviour-preservation results, WG-2 encountered (and overcame) severe technical difficulties. Apart from this work, WG-2 has also examined questions of decidability and state-based equivalences. Cooperation has been mainly with WGs 1 and 4.

Main achievements

- Categorisation and evaluation of equivalences.
- New results on congruences (for instance, for the first time taking into account systems with silent actions).
- Development of principles to define 'good' equivalences.
- New results on decidability.
- Ground work for research on state refinement and state-based equivalences.

Working Group 3: Algebra and Proof Methods

Approach

The task of Working Group 3 was the development of verification and synthesis techniques for structured nets. Classical Petri net techniques had to be adapted and extended, and their application to the verification of concurrent programs (through the net semantics defined by Working Group 6) investigated. The approach of Working Group 3 has been strongly centered upon efficiency: fast algorithms applicable only under certain constraints have been given more consideration than inefficient algorithms of wide applicability.

One line of approach has been to find very fast algorithms (polynomial time in the size of the *net*) for the checking of various important properties such as deadlock freeness. In accordance with known negative complexity results, this was expected to be possible only for subclasses of nets. The class of free choice nets has been the favoured choice for this WG, because a number of beautiful results could be found which establish strong links between the structure and the behaviour of a net, in all cases leading to the desired PTIME algorithms for a number of properties. WG-3 has developed this theory to the extent that a monograp on the subject is being currently written. WG-3 has also used and applied this theory to the problem of performance analysis of free choice systems (and their subclasses).

However, this theory, nontrivial though it is, has left WG-3 dissatisfied for several reasons. First, it is geared to unlabelled free choice nets and is thus at the same level as what has been called 'primitive' operations above (i.e. removing and adding of places and transitions – although these are obviously not the only structural operations employed in free choice theory). It seems very hard to adapt the theory to labelled nets in general, and to the operations of the PBC in particular. Indeed, WG-3 has also produced several 'impossibility' results of the nature that certain simple Box expressions cannot be simulated by free choice nets. A second reason for dissatisfaction was that only specific properties could be checked, rather than a class of properties.

WG-3 has sought and found two ways to overcome these limitations. Firstly, the WG has investigated the generalisation of its decision algorithms to semidecision algorithms which are applicable to more general classes of systems. Secondly, the generalisation from specific properties to classes of properties involves the definition of a logic and the consideration of model checking. WG-3 has approached model checking from two different angles.

(1) One of the developments has been to investigate under which circumstances model checking is feasible in polynomial time in the size of the *net* (rather than the transition system). Two recent results indicate that PTIME model checking is possible for a nontrivial class of nets (safe conflict free nets) and that the approach can be used to obtain fast model checking for persistent nets. This approach is beneficial not only by its generality, but also because persistency is a property that can be transferred immediately to labelled nets and to the PBC (as opposed to the free choice property which is much more indigenous to nets). In these results, the use of partial orders as defined in net theory (i.e. as alternations of local states and local transitions) plays a vital rôle, vindicating one of the presumptions of the project that concerns the exploitation of the strong points of Petri nets.

(2) A second development has concerned model checking in general. WG-3 has produced the notion of 'optimal simulation' which concerns the minimum amount of simulations that have to be carried out such that one still gets a correct model checker. This work is also based on concurrency semantics, albeit on the step sequences rather than on the processes of a system.

Almost all fast algorithms – be they decision algorithms, performance analysis algorithms, semidecision algorithms or algorithms for model checking of persistent nets – are based on linear algebraic and Linear Programming methods. This means that when the methods are eventually transferred to the Box expression level, they will rely on the latter's Petri Box semantics.

Main achievements

- Well-rounded structure theory of free choice nets.
- Application of structure theory to performance analysis.
- Judicious application of linear algebraic and Linear Programming methods to obtain fast algorithms.
- Various generalisations and extensions: semidecision algorithms; model checking.

Working Group 4: Abstract Models

The task of Working Group 4 was the study and extension of abstract models for concurrency – in particular Mazurkiewicz's traces – and the study of the relationships between these models and more 'concrete' models – several classes of Petri nets and process algebras.

The Working Group has obtained results on a wide spectrum of abstract models. Not only trace theory has been studied, also models such as asynchronous automata, event structures, vector synchronized systems and transition systems have been considered. Trace theory has been very thoroughly studied; several extensions – context-sensitive traces, comtraces, infinite traces – have been considered, which bring traces closer to distributed languages and extend the range of phenomena that can be modelled by the theory.

There has been interaction with WG-6 and WG-1 on action refinement and on implementations of the handshake synchronisation. Some of the work items have been carried out in cooperation with other groups (in particular, connections between free choice nets and process algebras with WG-3, research on CCS/CSP-like structuring operations with WG-1 and WG-6).

Main achievements

- Detailed study of several extensions of trace theory.
- Characterisation of processes formalised through dependency graphs.

- Advances in the recognizability problem for classical and infinite traces.
- Research on the trace languages of net classes.
- Strong relationships between elementary nets, transition systems and event structures.
- Study of action refinement in event structures.
- Study of vector synchronisation mechanisms and their effects.

Working Group 5: High Level Nets, Case Studies and Specifications

Working Group 5 was dealing with two tasks:

- Case studies. Test cases should be studied as motivation for the development of proper structuring techniques.
- Development of Specification and Requirements Engineering Techniques. Techniques for expressing formal specifications, for deriving formal specifications from semi-formal requirements and for relating specifications to each other had to be developed. Work on statecharts and logical formalisms was planned.

The Working Group has focussed its efforts on the first two phases of software development, namely the specification of requirements and the design of an operational model (the third being the actual production of code). The Group has studied logical formalisms for the requirement specification and Petri net based formalisms for the operational design. In particular, since the latter had to support data structures, high-level nets have been considered. The work on case studies has partly been done in cooperation with WGs 4 and 6.

Main achievements

- Design and study of new partial order logics.
- Design and development of the specification languages OBJSA and CO OPN.
- Connections between high level nets and functional approaches.

Working Group 6: Programming and Related Issues

Approach

Working Groups 6 has addressed its tasks extremely pragmatically, one by one as mentioned in the DEMON work programme. In addition, WG-6 has also developed (in cooperation with WGs 1 and 3) the operational semantics for the Box calculus, and WG-6 has carried out two case studies using the calculus. Furthermore, WG-6 has defined a class of high level nets with compositionality properties and has given a semantics for the full calculus in it (the only restriction being that recursive calls must be surrounded by call/return actions).

Main achievements

- Operational semantics for the Box calculus.
- Two nontrivial case studies using the calculus. One of them includes the sematics of unbounded buffer communication by basic means (i.e. in terms of elementary Petri nets).

- Petri net semantics of occam using a version of the PBC, including all semantically significant features.
- Definition of a basic Petri net based programming notation and its denotational Petri Box semantics. This includes the semantics of handshake and bounded buffer communication.
- The development of an alternative semantic model in terms of FIFO nets for unbounded buffer communication.
- Petri net semantics of static and dynamic priorities, and development of a theory generalising causal partial orders which captures the concurrency in prioritised systems better than is otherwise possible.
- Compositional high level net semantics of general recursion.
- Investigations on dynamic structure, including the semantics of dynamic processor allocation and net modelling.

All Working Groups have made considerable progress in their respective tasks. This progress has shown a decisive tendency to converge; strong links can already be discerned. Combining their results into a single framework is within reach. The combination can have the form of a unified theory, and of a tool that implements the algebra and its semantics together with the analysis algorithms.

Acknowledgements

All DEMON-s have expended very considerable work on behalf of the project's goals.
May future readers appreciate their work.

Grzegorz Rozenberg, in his rôles both as a DEMON member and as the Editor of the Advances, has provided enormous support.

The reviewers of DEMON, Jaco de Bakker, Robin Milner, Joseph Sifakis and Glynn Winskel, have spent much effort in exerting a beneficial influence on the course of the project.

The Hildesheim Group consisting of Javier Esparza, Sabine Karmrodt, Hans-Günther Linde, Holger Schirnick and Thomas Thielke have helped in many ways, not least in preparing this overview.

Last and closing a cycle, Esprit Basic Research Management – in particular, Georges Metakides and Michel Bosco – are acknowledged for their financial support and unfailing encouragement.

Eike Best, Hildesheim, March 1992.

The Box Calculus: a New Causal Algebra with Multi-label Communication[1]

Eike Best

Institut für Informatik, Universität Hildesheim
Marienburger Platz 22, D-3200 Hildesheim, Germany

Raymond Devillers

Laboratoire d'Informatique Théorique, Université Libre de Bruxelles
Boulevard du Triomphe, B-1050 Bruxelles, Belgium

Jon G. Hall

Computing Laboratory, The University of Newcastle-upon-Tyne
Newcastle-upon-Tyne, NE1 7RU, U.K.

Abstract

A new Petri net calculus called the calculus of Petri Boxes is described. It has been designed to allow reasoning about the structure of a net and about the relationship between nets, and to facilitate the compositional semantic translation of high level constructs such as blocks, variables and atomic actions into elementary Petri nets. The calculus is located 'midway' in such a translation.

This paper first defines an algebra of Box expressions. A corresponding algebra of Boxes is then defined and used compositionally as a semantic model of Box expressions. The two algebras feature a general asynchronous communication operation extending that of CCS. Synchronisation is defined as a unary operator. The algebras also include refinement, iteration and recursion. It is shown how they can be used to describe data and blocks.

As the main results of this paper, it is proved that the Box algebra satisfies various desirable structural laws and enjoys two important behavioural properties. The paper also contains an example proof of a mutual exclusion protocol.

Keywords

Petri Nets, Process Algebras, Compositional Semantics, Multiway Synchronisation, Refinement, Recursion, Data.

Contents

[1] Work done within the Esprit Basic Research Action 3148 D-E-M-O-N (Design Methods Based on Nets).

1 Introduction

Petri nets [54,56] are a causal model of concurrency in that they allow one to reason about events between which there is no causal relationship. A major drawback of Petri nets is that they have very little basic structure in an algebraic sense. This may have adverse implications for the systematic construction and verification of systems. An important alternative to Petri nets are process algebras [1,43,50]. These approaches were originally based on a single observer (interleaving) model in which the occurrence of events is serialised even for independent processes[2]. A disadvantage of this approach is that causality information may be lost.

Part of the objectives of the Esprit Basic Research Action DEMON has been the search of a good way of bringing together the two models. We quote from [13]:

> 'The long term objective for which this Action lays the foundation is the development of a design calculus based on Petri net theory and supporting modular structuring, as well as specification and analysis techniques. (...) In outline this involves: Detailed analysis of existing variants of the basic model ('net classes'), and distillation of their features into a small number of tractable, well behaved and modular net classes of wide applicability ...'

This paper describes part of the results of this endeavour. We produce a new net-based model for specifying concurrent systems called the *Box algebra*, which has as its natural semantic domain a simple but versatile form of structured Petri nets. The Box algebra has been developed,

- to serve as the semantic domain of a compositional semantics of high level concurrent programming languages such as occam [48];
- to provide a compositional semantics in terms of Petri Nets and their associated partial order (true concurrency) behaviour.

In a sense, the Box algebra is located midway between such user-friendly high-level constructs as blocks, variables, recursion, critical sections etc., and such theoretically important primitive concepts as local states and transitions. It is designed to serve as a bridge between (Petri Net) theory and (Concurrent Programming) applications. Its aim is also to build a superstructure on top of Petri nets, which allows one to reason about the relationship between nets and about the structure of a net – rather than just about a net's immediate constituents such as places, transitions and arcs – while retaining the possibility of exploiting indigenous analysis methods known from net theory. This kind of superstructure is not achieved by the High Level net model [29,45,55], since the inscriptions of a high level net provide information neither about the net's structure nor about its relation to other nets.

The Box algebra consists of two parts:

(i) A syntactic domain of *Box expressions*, and

(ii) A semantic domain which we christen the domain of *Petri Boxes*.

There exists a semantic homomorphism from Box expressions to Boxes. Boxes are equivalence classes of Petri Nets equipped with an interface. They behave like black boxes extended with a communication facility and can be combined appropriately at their interfaces. The most important technical requirement we impose on our semantics is that of strict *compositionality*, i.e., the requirement that operators on one domain and operations on another domain should match each other. This requirement, together with the requirement of being able to model data and atomic actions, has had an interesting influence on the types of events we allow. As we have moved away from an interleaving semantics we can now give a true implementation of atomic actions as single events without the use of semaphore type constructs as in [50]. For instance, an atomic action of the form

[2] As a rule, process algebra formalisms depend on the interleaving view. For instance, the well known expansion theorem holds only in the presence of interleaving; the 'head normal form of processes' which is an important proof technique depends on the expansion theorem.

$$\langle x := y + 1; c?y; x := x + 1 \rangle$$

(where the brackets $\langle \ldots \rangle$ enclose 'virtually atomic' actions – see [3] and others) involves a multi-way synchronisation of the action's constituent parts, namely the variables x and y, the channel c and the process in which the action is contained. However, since the variables (and the channel c) may be declared in different blocks, the compositionality principle makes it necessary to build this multi-way synchronisation incrementally.

The stepwise construction of multi-way synchronisations is achieved by the technical device of *multi-labels*, that is, to allow the transitions of a net to be labelled by multisets of action names. We have found the CCS approach (with conjugate matching labels) to be more conducive to compositionality than the TCSP or COSY approach. The operators we have defined on multi-labels are, therefore, extensions of those defined on the CCS single label algebra. The generalisations allow various unifying observations to be made (such as a silent action being the empty multi-label, i.e., literally the absence of any visible action, and the use of simple disjoint union for concurrent composition). SCCS and related formalisms [18,58,50] achieve a similar generalisation. However, our approach differs fundamentally from SCCS in that it is asynchronous while SCCS is synchronous[3]. It is for this reason that commutativity of synchronisation is a nontrivial issue in our approach; the commutativity result is central to the present paper.

The paper is structured as follows. Section 2 defines the Box expression algebra. Section 3 introduces the domain of the Petri Box model. Section 4 defines the Box algebra, i.e., the set of operations on the domain. In Section 5, elementary structural properties such as the commutativity of synchronisation are stated and proved. Section 6 defines the semantic homomorphism from Box expressions to Boxes, and Section 7 describes some elementary behavioural properties of the semantics. Section 8 contains miscellaneous comments. It is described how data can be translated into Box expressions; an example is given; and some discussion on literature and on further work can be found. Section 9 contains some concluding remarks and an outlook, as well as acknowledgements.

This paper gives the reader an intermediate 'snapshot' on the development of a Petri net indigenous algebra. It is not to be viewed as a self-contained finished piece of work. Indeed, several problems have been left open here. They are being taken up in other work that is in progress, which will be reported upon in future papers. We indicate such open ends throughout the text.

2 The Box Expression Algebra

2.1 Action names

Before defining the syntax of the Box expression algebra, we describe its basic constituents. Action names are defined axiomatically; in the model, they will reappear as transition labels.

Definition 2.1

We assume a countably infinite set of action names, A, to be given. On A, we assume a bijection $\hat{}$ with the following properties to be defined:

$$\hat{}: A \rightarrow A$$
$$\text{with} \quad \hat{a} \neq a \text{ and } \hat{\hat{a}} = a \text{ for all } a \in A.$$

This bijection is called conjugation, and $a, \hat{a}$ are called conjugates of each other[4]. ■ 2.1

In applications, we shall choose the elements of A freely. Whenever a name a is chosen, it must be specified what its conjugate, $\hat{a}$, looks like. For instance, the conjugate of $c!1$ ('output the value 1 on channel c') could be $c?1$ ('input the value 1 from channel c'), but not $c?2$. Or, the conjugate of $C?$ could be $C!$; or, the conjugate of $P_2!$ (in P_1) could be $P_1?$ (in P_2).

Communications are based on multisets; Appendix A gives some definitions relating to multisets.

[3] The difference can be seen on the example $a\|\overline{a}\|a\|\overline{a}$ which is discussed in Section 4.5.

[4] We use the symbol $\hat{}$ instead of CCS's $\bar{}$, because we reserve the overbarring symbol for the operational semantics of Box expressions [17].