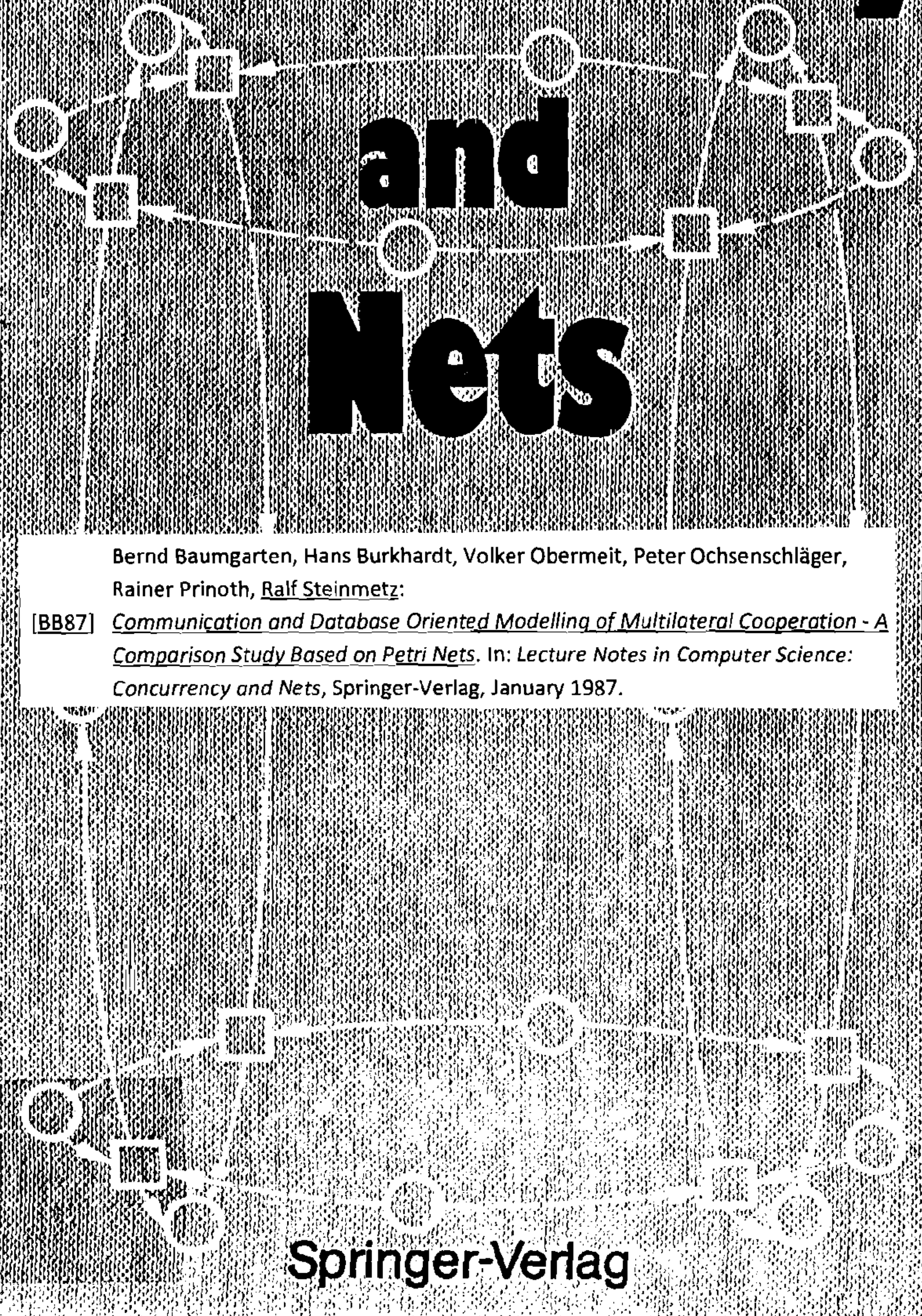
K.Voss H.J.Genrich
G.Rozenberg (Eds.)

Concurrency and Nets



Bernd Baumgarten, Hans Burkhardt, Volker Obermeit, Peter Ochsenschläger, Rainer Prinoth, Ralf Steinmetz:

[BB87] *Communication and Database Oriented Modelling of Multilateral Cooperation - A Comparison Study Based on Petri Nets*. In: *Lecture Notes in Computer Science: Concurrency and Nets*, Springer-Verlag, January 1987.

Springer-Verlag

COMMUNICATION AND DATABASE ORIENTED MODELLING OF MULTILATERAL COOPERATION - A COMPARISON BASED ON PETRI NETS

Volker Obermeit, Ralf Steinmetz
Technische Hochschule Darmstadt

Bernd Baumgarten, Heinz-Jürgen Burkhardt,
Peter Ochsenschläger, Rainer Prinoth
GMD Darmstadt

Abstract

The design of distributed sytems requires integrating viewpoints of various disciplines into a common modelling approach. Distributed applications are characterized by the need to correlate different processing states and data, a task which comprises communication and database aspects.

The present paper deals with the question of how the modelling of distributed applications is influenced by the viewpoint taken by the modeller. To answer this question we model an example from the area of human cooperation first from the standpoint of a communication systems specialist and then from the standpoint of a database systems expert. In both cases, we begin by identifying substructures and their interrelations as suggested by the respective concepts. Expressing both models as higher level Petri nets we are then able to investigate the relation between the communication and database aspects.

Topologically, the two nets differ mainly in their degree of explicitly modelling local states and message passing. The tasks of coordination and synchronization between the participants, however, are so strictly determined by the problem that their net representations are very similar. The major difference lies in the field-specific interpretations of the nets, which induce dissimilar partitions into subnets, namely into participants and message types on one hand and into transactions on the other hand.

1. Introduction

Distributed applications generally include aspects of both communication and databases. In this paper, we investigate the relation between these aspects by comparing modelling concepts from the two fields. The latter are presented by means of an example from the area of human cooperation, namely the contract phase of a car purchase financed by a bank loan.

We start by structuring the application problem into application substructures and their interrelation. As is to be expected, this is done partly by reasoning at the general problem level and partly with particular regard to the modelling concepts of the communication and database fields. Then, by mapping these substructures and their relations on the concepts dedicated to the two areas, we obtain a database- and a communication-oriented model. As Petri nets are particularly suited for the modelling of complex concurrent systems /Petri73,77/, these models are both finally formalized as product nets in order to allow a systematic comparison of the concepts.

In section 2 a natural language description of the contract phase is given. In section 3 a communication oriented model is derived from this description and presented as a product net. Similarly, section 4 derives a database oriented model, which is then also described as a product net. Both nets serve in section 5 for a comparison of the structures and concepts applied under the aspects of communicating systems on one hand and databases on the other hand. Finally some hints are given how to bridge the gap between the two areas.

A more detailed version of this paper has appeared as /OSBBOP86/.

Our thanks are due to Mrs. Gallinat for preparing the illustrations.
For their encouragement and support we are indebted to Prof. Dr. Schek, head of the data base systems section in the Department of Computer Science of the Technische Hochschule Darmstadt, to Dr. Raubold, head of the Institute for Systems Technology in the GMD, as well as to all the other participants of our joint seminar on synchronization.

2. Negotiating the Contract of a Car Purchase Financed by a Loan - Natural Language Description

A customer who wishes to acquire a car contacts a licensed dealer of a certain manufacturer. He informs the dealer about his personal data, financial situation, and the model he is interested in.

The dealer is contractually connected not only to the manufacturer but also to a bank. He functions as a loan broker and applies on behalf of the customer for a loan with his partner bank. The latter checks the customer's credit rating. When the check proves positive, the bank notifies the dealer of its approval, making a specific loan offer. Otherwise the loan is refused. The manufacturer finds out whether the required model is available. If this is the case, he assures the dealer of delivery; otherwise he informs the dealer that delivery is not possible.

When both loan and supply are assured, the dealer makes the customer an offer; otherwise he gives the customer a negative reply. If bank and manufacturer give one positive and one negative reply then the dealer responds to the positive reply with a cancellation.

In the case he gets an offer the customer can either decline or he can accept by placing an order. The customer's order is transformed by the dealer into an order to the bank and another to the manufacturer. As soon as both orders are confirmed the dealer confirms the order to the customer.

In comparison with a real car purchase, the following simplifications are assumed:

The customer gives the dealer the full information, as to what kind of car he is willing to buy under which conditions, in one single step. There are no further steps of negociation between customer and dealer before the dealer's offer or his negative reply. Similar assumptions hold for the dialogues between the dealer on one side and his partner bank and the manufacturer, respectively, on the other side.

The bank disposes of unlimited financial resources and decides randomly whether it grants a loan or not. If it does, then the loan offer depends only on the customer's financial situation.

The manufacturer assures delivery whenever the wanted model is on stock. He has limited resources, which are, however, increased randomly.

3. Communication Oriented Model

The interplay described informally in section 2 is an example of a cooperation of autonomous partners for the purpose of reaching a common goal. Autonomy means in this context that the partners dispose of their own resources, that they are capable of concurrent actions and possess a certain freedom of decision. The cooperation between them is based on a common initial understanding, which is then perpetuated consistently by the actions of the single partners. Due to their spatial distribution and the independence of their memories, the only form of interaction between them is the exchange of messages. Thus we denote by a cooperation the establishment and perpetuation of a common understanding (context) of communication partners by the exchange of data.

The simplest form of cooperation is the bilateral one, where we have a communication association between two partners. A multilateral cooperation can be described as a correlation of bilateral cooperations, where the vehicle is a multitude of two-party communication associations.

In our example, the participating communication partners play the roles of clients and contractors. Each client/contractor-relation constitutes a two-party cooperation. Each client enters direct relations to one or more contractors. Each contractor on his part can be a (sub-)client, namely if he cannot satisfy the order alone. Consequently, the client/contractor-relations and thus the multilateral cooperation are structured as a tree.

The *nodes of the tree represent clients and/or contractors. The* root of the tree represents the main client, while its leaves comprise those (sub-)contractors, who do *not also play the role of a client:*

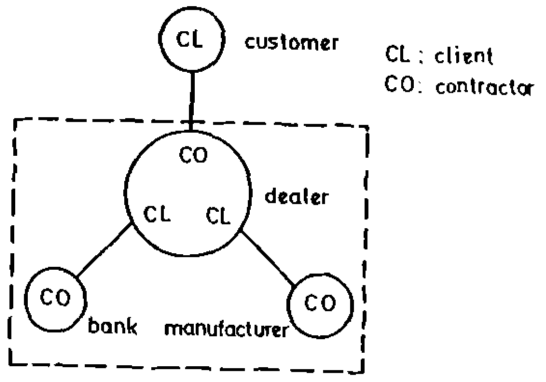


Fig. 1: Client-contractor-tree

In fig. 1, the dealer can be regarded as representing towards the customer the complete service to be rendered (dashed rectangle).

The common initial context and the consistent perpetuation of the common understanding can be considered and modelled as the correlation of local processing states to global cooperation states. We call this correlation synchronization.

The tool for maintaining the common context is the dialogue structure shown in fig. 2. In /OSBBOP86/, we show how this structure can be derived from an interplay of 'handshakes'.

Wanting to describe multitudes of car purchases, we base our model of this application on the notion of roles played by individual participants. In particular we introduce and interrelate the roles

customer	C,	dealer	D,
bank	B,	manufacturer	M.

In these roles the characteristical behaviour of the participants of this distributed application is laid down. In the context of a particuler *car purchase these roles will be played by particular* persons and institutions, e.g.

the customer	Will Knotpaigh,	the *dealer*	*Kay Owe & Co.,*
the bank	Lown & Sharques,	the manufacturer	Russ T. Tincan.

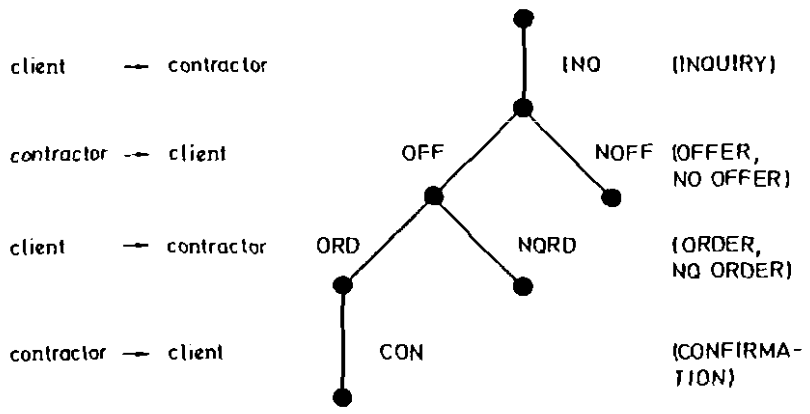


Fig. 2: Dialogue structure

The formal tool we use in our model are product nets, a form of higher level Petri nets with individualized tokens (whose identity is referred to in arc labels and transition inscriptions) as well as inhibitor arcs and erase arcs. Product nets were introduced *and* formalized in 1984 /EcPr84, EcPr85/ for the purpose of modelling communication systems in the PROSIT project /BEP85/.

In the course of our work with formal models for communication systems, under particular consideration *of standardization* activities regarding Open Systems Interconnection, certain patterns of communicative behaviour crystallized. *In constructing* our models, we treated these patterns both as building blocks and as strategies for connecting them, in the style *of a* design method.

On the Petri net level these building blocks are essentially subnets *bounded* (relative to the remainder) by transitions. Generally, building blocks obey the following demands: Omitting labels reduces *them (often)* to place-transition-nets modelling single activities in single roles. In this form they can only work for one activity at a time. Their state of progression in a single activitiy is recorded in interior places. Each of their transitions 'serves' (is neighbour of) at most one 'interface' (place outside the building block). Interface places are often drawn as triangles in order to emphasize the direction of information flow.

In our example, the entire net consists of four *subnets* (building blocks) describing the roles C, D, B, M, connected by 'communication' ('interface') places. CUSTOMER, BANK and MANUFACTURER essentially consist of a CLIENT or CONTRACTOR kernel topology (figs. 3, 4) which is responsible for producing the *dialogue* structure shown above.

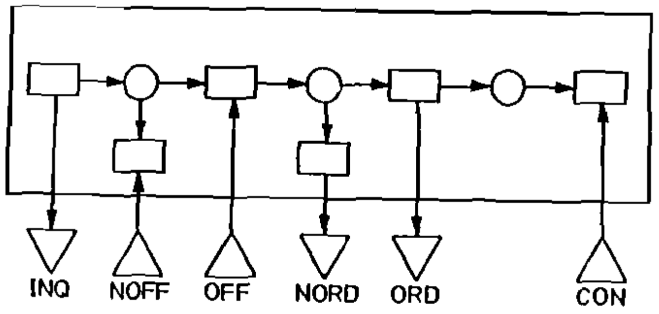


Fig. 3: Client

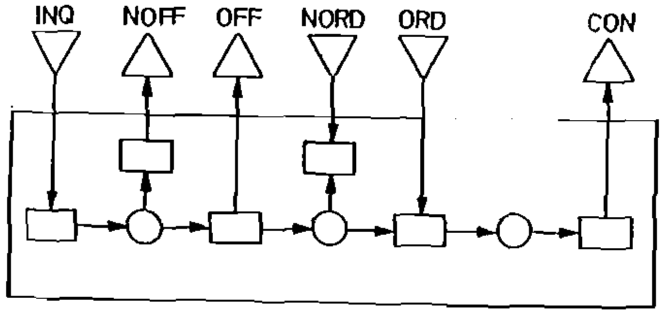


Fig. 4: Contractor

The DEALER building block models the dealer's task of coordinating the dialogues to the customer, bank and manufacturer. In a more detailed model the dealer contains additionally two CLIENT and one CONTRACTOR building blocks, such that concurrent activites within the dealer (-organization) may be represented, too /BBOP86/.

In the full model (fig. 5), the kernel topologies are enhanced with further topological details:

- counters x.REF which permit to distinguish locally between concurrent car purchases. These references accompany the messages exchanged between the participants (like 'our/your reference').
- B.LOANOFF models how the bank determines a possible loan offer from the customer's informations. References to loans actually granted are created by means of B.LOANREF.
- The available manufacturer's stock in M.STOCK is increased by the production of additional cars (M.PROD) whose potential model description is taken from M.MODELS. They are identified uniquely relative to their manufacturer by use of M.CARREF. M.NOFF will only occur if no suitable car is on stock (inhibitor arc from M.STOCK). Making an offer (M.OFF) will reserve the car, a cancellation (M.NORD) will make it available again.

On the dealer's part, D.BANK, D.MANUF and D.PRICE model the associations

dealer	→	partner bank,
dealer	→	manufacturer,
dealer, model	→	price.

When the answers from bank and manufacturer do not lead to an offer to the customer, the B- and M-tags in the D.NOFF-transitions indicate by whose negative reply this was caused.

Due to space limitations we confine the list of arc labels to the neighbourhood of the transitions M.INQ, M.OFF, M.NOFF. The interested reader will probably arrive at a labelling very similar to ours if he or she just fills in the informations to be passed around according to the problem description; otherwise he or she is referred to the full listing in /OSBBOP86/.

<u>partial list of arc labels:</u>

```
51: ‹D_INQ_M,  d_name, d_ref, m_name, car_spec›
52: ‹M_NOFF_D, m_name, m_ref, d_ref, d_name›
53: ‹M_OFF_D,  m_name, m_ref, d_ref, d_name›

61: ‹m_name, m_ref›
62: ‹m_name, m_ref+1›
63: ‹m_name, m_ref, d_ref, d_name, car_spec›
64: ‹m_name, car_spec, car_ref, FREE›
```

4. Database Oriented Model

The same scenario as in chapter 2 can also be modelled under the aspect of using a database management system. This leads to a radically different view of our model, as it is heavily influenced by the available tools:

- Static aspects will be modelled by the data base, consisting of raw data and its logical structure as defined in the conceptual schema. This schema describes the structure of the observed miniworld, and the data contents comprise the actual state.
- Dynamic aspects are modelled by transactions, embedded in application programs and executed by the database management system. These transactions implement the consistent state changes of the data base.

We will constitute our model by fitting our miniworld, consisting of customers, dealers, manufacturers and banks, into the framework of a conceptual schema and by defining transactions which simulate the actions in our scenario. Distribution and autonomy of the actors is of

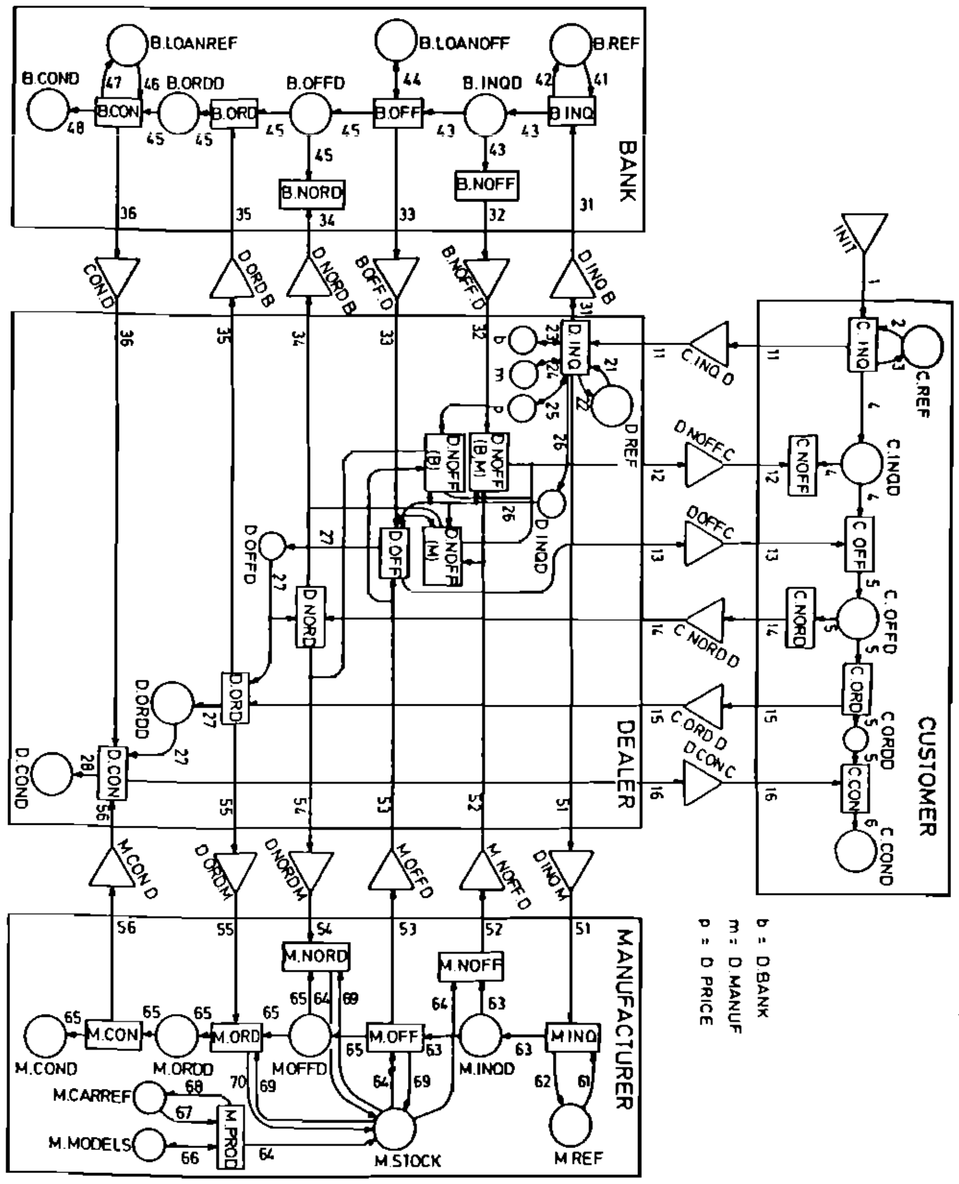


Fig. 5: The Communication Oriented Petri Net Model

minor importance, as the database management system acts as a global centralised coordination instance and hides the communication oriented aspects from the actual user. This can be done through artificial centralisation of the distributed process or through the use of a distributed data base management system.

Our first step consists of an informal problem description and a collection of the essential facts to isolate the miniworld concerned. Afterwards we model the relevant objects and their relationships as a conceptual schema based on the relational model of Codd /CODD70/. Here all data reside in tables (relations), which consist of identically structured records (tuples) consisting of atomic data fields (attributes). Relationships are implicitly contained in the data, e.g. through equality of corresponding attribute values. Our nomenclature will be 'relation-name (attribute-name-1, ..., attribute-name-n)'. When defining a conceptual schema one should take into account that certain aspects of the miniworld can possibly be modelled outside the data base (i.e. as external agents interacting with it). In our case we decided to use this open approach: The customer is actually modelled as the user of the database management system, and his interactions with the dealer comprise the I/O-interface of the system. The remaining data are partially grouped, corresponding to the actors to which they belong. This grouping could be used as a base for a partitioning in the case of a distributed database management system.

The following conceptual schema shows all the necessary relations, each one described by its name (in capital letters) and its list of attributes (enclosed in parentheses, attribute names in lower case). Attribute domains (e.g integer) are omitted for the sake of simplicity. The groups are separated by short explanations (enclosed in **).

CONCEPTUAL SCHEMA:

STOCK (manufacturer, model, status, car-id)
   ** Each tuple represents a car available from a certain manufacturer. **
LOANOFF (bank, price, finances, loan offer)
OFFERED LOANS (bank, customer, price, loan offer, loan-id)
   ** LOANOFF models an oracle which decides whether a loan is offered or not. Offers actually made are stored in OFFERED LOANS. **
PARTNER BANK (dealer, bank)
LICENSED DEALER (dealer, manufacturer)
PRICE CATALOG (model, price)

OFFER FILE (offer-id, dealer, customer, car-id, model, price, manufacturer, bank, loan offer, loan-id)

** All offers to customers are held in the OFFER FILE for further use.**

LOAN CONTRACT (dealer, customer, car-id, price, loan offer, bank)

PURCHASE CONTRACT (dealer, customer, car-id, model, manufacturer)

The last step is to define transaction programs to simulate the actual proceeding in our scenario. As the semantics of transactions guarantee atomicity, consistency, integrity-preservation and durability /HäRe83/, we see that our program must be divided into several (sub-) transactions, since in reality there exist observable intermediate states of the process (i.e. offer made to customer, but not yet answered). Our particular example is written in SQL /IBM83/ embedded in Pseudo-PASCAL. Due to space considerations only the main program's body is shown in table 1. The reader is recommended to consult /OSBBOP86/ for refinements of the sub-transactions and a full listing.

4.1 Representation of the Database Oriented Model as a Petri Net

For the purpose of comparing the two viewpoints of this 'car purchase' example (given in sect. 3 and 4) the database 'implementation' will be transformed into a Petri net of the type used for the communication style representation /EcPr84, EcPr85/. We distinguish two layers:

1. the user-interface including the user himself (customer),
2. the database-layer.

4.1.1 The User Interface including the Customer

The relevant attitudes and reactions of persons intending to buy a car have to be incorporated into the Petri net. This layer (fig. 6) is established for the very purpose of distinguishing all surrounding activities from the kernel, the database system.

4.1.2 The Database Layer

From the entire model (fig. 7), we selected the transaction 'manufacturer inquiry', on which we demonstrate the transformation method as well as the results achieved.

The following rules characterize the transformation:

- A transaction (subtransaction) is transformed into a set of transitions, together with the corresponding places and arcs.
  E.g. the transaction 'manufacturer inquiry' is composed of: the transitions 'positive manufacturer' and 'negative manufacturer' the places 'manufacturer-input', 'manufacturer-output' and 'stock', the arcs no. 8, 30, 53, 54, 55 and 56.
- Every relation is represented by a place.
  E.g. the relation STOCK corresponds to the place 'stock'.
- The attributes of the relations constitute the components (in the Petri net context) of the tokens of the place.
  E.g. the components 'manufacturer, model, status, car-id' of the relation STOCK make up the components of the tokens at the place 'stock'.
- The manipulation of data will show up as the dynamics of the Petri net (the respective markings of the places and the occurrences of the transitions).

```
program   LOAN_PURCHASE ;
 begin
   read customer inquiry ;
   begin_transaction  OFFER ;
      select  bank, manufacturer, price
      from    partner bank, licensed dealer, price catalog
      where   "specifications of customer inquiry apply" ;
      call transaction  MANUFACTURER_INQUIRY ;
      call transaction  BANK_INQUIRY ;
      case concat(manufacturer_answer,bank_answer) of
        '++' : begin
                 insert  offer  into offer file ;
                 return ('positive')
               end
        '+-' : begin
                 call transaction CANCEL_BANK_OFFER ;
                 return ('negative')
               end
        '-+' : begin
                 call transaction CANCEL_MANUFACTURER_OFFER ;
                 return ('negative')
               end
        '--' :   return ('negative')
      end;
   end_transaction  OFFER ;
   if offer positive
   then begin
          display offer ;
          read customer decision ;
          if decision = 'buy'
          then begin
                 call transaction PURCHASE ;
                 print purchase contract ;
                 print loan contract ;
               end
          else call transaction CANCEL_OFFER ;
        end;
end.
```

Table 1: LOAN_PURCHASE (top level)

In performing these transformations, we observed that many sequences of operations on relations which often appear in real implementations can be transformed into simple transitions with adequately labelled arcs.

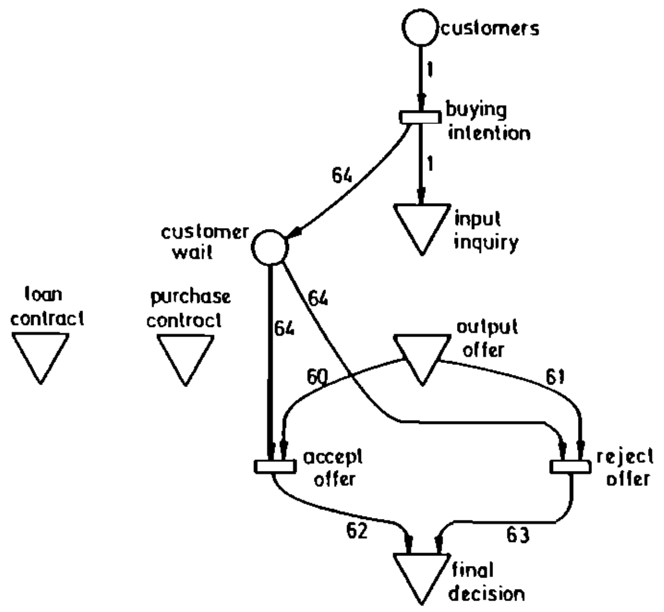


Fig. 6: Database oriented Petri net model - user layer

4.1.3 Places and arcs of the transaction 'manufacturer inquiry'

Stock: This is the place corresponding to the relation STOCK, which represents the supply of cars. Every car is identified by the 'car-id'. Its status can be 'sold', 'available' or 'reserved'. The place 'stock' can thus hold tokens of the following types:

⟨manufacturer,model,AVAILABLE,car-id⟩
⟨manufacturer,model,SOLD,car-id⟩
⟨manufacturer,model,RESERVED,car-id⟩

Variable names are written lower case while constants - here the different status values - are written in capital letters. At the beginning we need no tokens in this place, because the production of cars is explicitly modelled by a different transaction 'production'.

Manufacturer-input: This place and the manufacturer-output represents parameter passing between the dealer and the manufacturer:

⟨transaction-id,customer,dealer,manufacturer,model⟩

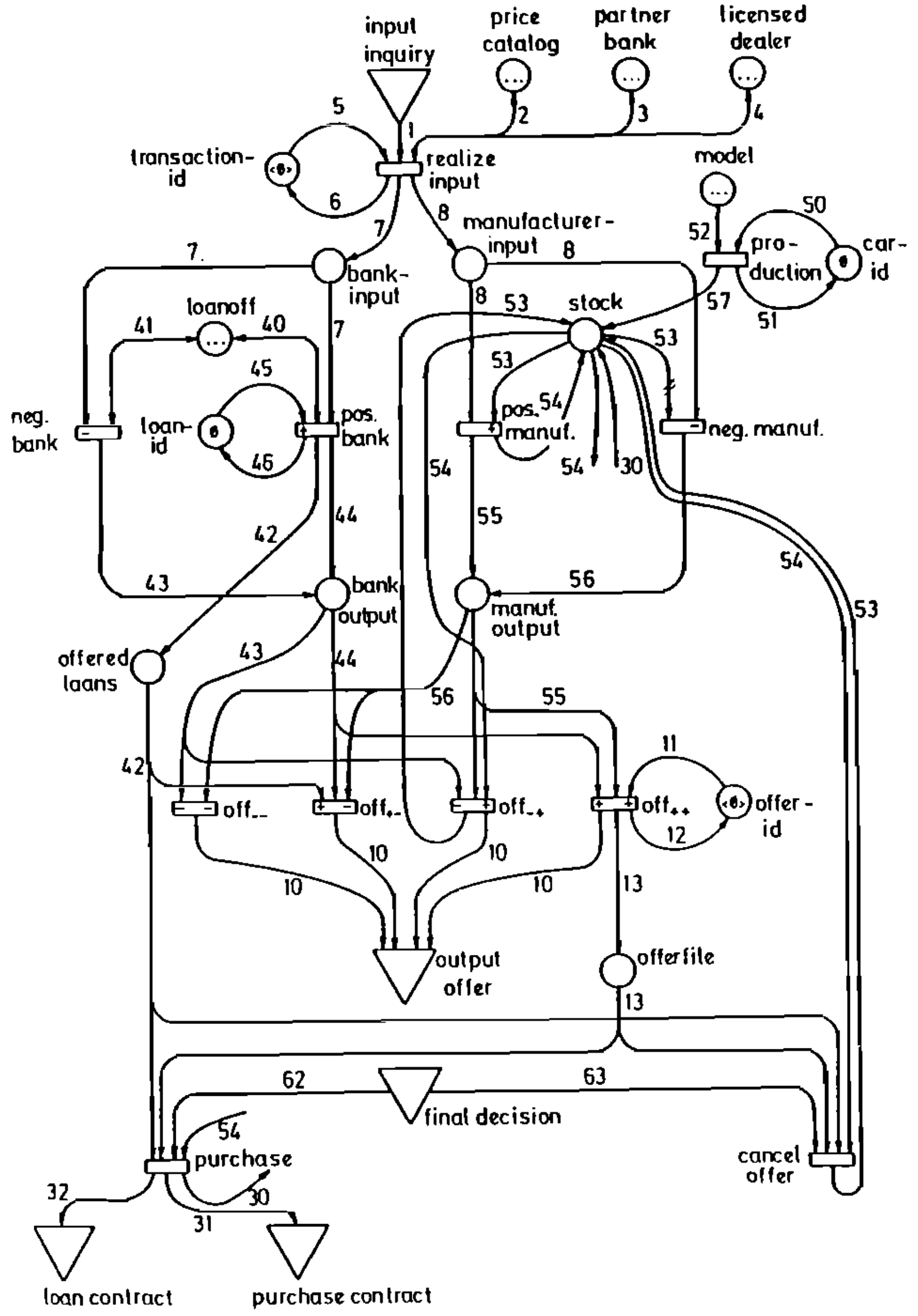


Fig. 7: Database Oriented Petri Net Model - Database Layer

<u>Manufacturer-output:</u> The outcomes of the inquiries to the manufacturers show up as the tokens in this place. They are of the types

‹transaction-id,NO-CAR-OFFER,dealer,customer,car-id,model, manufacturer›

‹transaction-id,RESERVED,dealer,customer,car-id,model, manufacturer›

The arcs attached to this transaction bear the following labels:

8: ‹transaction-id,customer,dealer,manufacturer,model›
30: ‹manufacturer,model,SOLD,car-id:
53: ‹manufacturer,model,AVAILABLE,car-id›
54: ‹manufacturer,model,RESERVED,car-id›
55: ‹transaction-id,NO-CAR-OFFER,dealer,customer,car-id,model, manufacturer›
56: ‹transaction-id,RESERVED,dealer,customer,car-id,model,manufacturer›

5. <u>A Comparison of the Models</u>

In the following, we compare the two net models, viz. the communication oriented one ('COMM') and the database oriented one ('DABA').

Both models decompose the total task given by the problem description (section 2) into subtasks of different extents (bilateral coordination steps, transactions). In both cases, the interplay of the respective subactivities in the context of each model achieves the full aim of the cooperation.

A comparison of the two models reveals that the database standpoint of the action constitutes a higher level (more abstract) view than the communication standpoint:

- an occurrence sequence of COMM beginning with D.ORD and ending with D.CON (including occurrences of B.ORD, B.CON, M.ORD, M.CON) corresponds to an occurrence of the transition 'purchase' in the DABA model.
- an occurrence sequence in the COMM model of D.NORD, followed by B.NORD and M.NORD in any order, corresponds to an occurrence of the 'cancel offer' transition in the DABA model.
- etc.

It turns out that the ways of modelling the synchronization mechanisms

- for the coordination of subactivities to a total task
- for the separation of different total tasks (several car purchases) during the concurrent access to common resources (e.g. manufacturer's stock)

essentially differ only in that the COMM model explicitly addresses the spatial separation of the systems.

The DABA model can be considered as a requirement specification for arbitrary implementations. If a centralized realization is intended, no further efforts are needed with respect to design logic. In the case of a distributed realization, non-local transactions may arise, which necessitate communication in order to ensure their atomicity.

The task treated in this paper nicely demonstrates an example of how these communication aspects are to be dealt with in such a case:

Table 2 contains corresponding partitions of the transitions of both models; it is obtained by superposing the DABA transaction structure given in section 4 and the COMM net (fig. 5) and DABA net (figs. 6, 7), respectively. In the net models each transaction consists of the subnet spanned by the transitions listed.

transaction .....sub-transaction	COMM transitions	DABA transitions
OFFER	D.INQ D.NOFF(B,M) D.OFF	realize input off-- off++
.....BANK_INQUIRY	B.INQ B.OFF B.NOFF	neg. bank pos. bank
.....MANUFACTURER_INQUIRY	M.INQ M.OFF M.NOFF	neg. manuf. pos. manuf.
.....CANCEL_BANK_OFFER	B.NORD D.NOFF(M)	off+-
.....CANCEL_MANUFACTURER_OFFER	M.NORD D.NOFF(B)	off-+
PURCHASE	D.ORD D.CON B.ORD B.CON M.ORD M.CON	purchase
CANCEL_OFFER	D.NORD B.NORD M.NORD	cancel offer
PRODUCTION	M.PROD	production

Table 2: Transaction structure in the net models

Obviously, in a realisation by a distributed database system there will be 'local' transactions (BANK_INQUIRY, MANUFACTURER_INQUIRY, PRODUCTION), as well as 'distributed' transactions (OFFER, CANCEL_BANK_

OFFER, CANCEL_MANUFACTURER_OFFER, PURCHASE, CANCEL_OFFER). The former are transactions of the type met in non-distributed databases.

Of course, different partitions are conceivable, including such that lead exclusively to local transactions. The particular choice of a partition will generally neither facilitate nor complicate the task of modelling: Different coverings by transactions will only shift the complexity of implementation between the database user and the database system.

Distributed transactions are user-friendly, as they guarantee per se the integrity of the distributed databases they manipulate. In their case it is up to the (invisible) interplay of the local database managers concerned to ascertain this integrity (e.g. by use of the CCR algorithm specified in ISO OSI standards for tree-structured transactions /BOP85b, BOP87/). If only local transactions are used, the user has to take care that the interplay of the transactions will produce the intended global results.

References

/BBOP86/ B.Baumgarten, H.J.Burkhardt, P.Ochsenschläger, R.Prinoth: The Signing of a Contract - a Tree-Structured Application Modelled with Petri Net Building Blocks, in: Advances in Petri Nets 1985, G. Rozenberg (ed.), Springer LNCS 222, 1986

/BEP85/ H.J.Burkhardt, H.Eckert, R.Prinoth: Modelling of OSI-Communication Services and Protocols using Predicate/Transition Nets, in: Protocol Specification, Testing, and Verification, IV, Y.Yemini et al.(ed.), North-Holland, 1985

/BOP85a/ B.Baumgarten, P.Ochsenschläger, R.Prinoth: Building Blocks for Distributed System Design, in: Protocol Specification, Testing, and Verification, V, M. Diaz (ed.), North-Holland, 1986

/BOP85b/ B.Baumgarten, P.Ochsenschläger, R.Prinoth: A Formal Model of the CCR Algorithm, Arbeitspapiere der GMD Nr.186, 1985

/BOP87/ B.Baumgarten, P.Ochsenschläger, R.Prinoth: Synchronization in Tree-Structured Transactions - a Case Study, to appear in: Protocol Specification, Testing, and Verification, VI, North-Holland

/Codd70/ E.F. Codd: A Relational Model for Large Shared Data Banks, Communications of the ACM Vol.13 , no.6, June 1970.

/EcPr84/ H.Eckert, R.Prinoth: Produktnetze - Definition eines PROSIT-Beschreibungsmittels, Arbeitspapiere der GMD Nr.92, 1984

/EcPr85/ H.Eckert, R.Prinoth: Grundsätzliche Betrachtungen und Bemerkungen zu Produktnetzen, Studien der GMD Nr.106, 1985

/HäRe83/ T.Härder, A.Reuter: Principles of Transaction-Oriented Database Recovery, ACM Computing Surveys Vol.15 , no.4, December 1983

/IBM83/ IBM Manual: SQL Application Programming, no.SH24-5018-2 , 1983

/OSBBOP86/ V.Obermeit, R.Steinmetz, B.Baumgarten, H.J.Burkhardt, P.Ochsenschläger, R.Prinoth: Datenbank- und kommunikations-orientierte Modellierung einer mehrseitigen Kooperation - ein Vergleich auf der Basis von Netzen, Studien der GMD Nr.115, 1986

/Petri73/ C.A.Petri: Concepts of Net Theory, in: Mathematical Foundations of Computer Science - Proceedings, High Tatras, 1973

/Petri77/ C.A.Petri: Communication Disciplines, in: Computing System Design - Proceedings, Newcastle upon Tyne, 1977