Proceedings

of the

1996 National Research and Development Conference

Industry meets Academia

Interaction Conference Centre, University of Natal,
Durban.
26 & 27 September

Edited by
Vevek Ram
FOREWORD

This book is a collection of papers presented at the National Research and Development Conference of the Institute of Computer Scientists and Information Technologists, held on 26 & 27 September, at the Interaction Conference Centre, University of Natal, Durban. The Conference was organised by the Department of Computer Science and Information Systems of The University of Natal, Pietermaritzburg.

The papers contained herein range from serious technical research to work-in-progress reports of current research to industry and commercial practice and experience. It has been a difficult task maintaining an adequate and representative spread of interests and a high standard of scholarship at the same time. Nevertheless, the conference boasts a wide range of high quality papers. The program committee decided not only to accept papers that are publishable in their present form, but also papers which reflect this potential in order to encourage young researchers and to involve practitioners from commerce and industry.

The organisers would like to thank IBM South Africa for their generous sponsorship and all the members of the organising and program committees, and the referees for making the conference a success. The organisers are indebted to the Computer Society of South Africa (Natal Chapter) for promoting the conference among its members and also to the staff and management of the Interaction Conference Centre for their contribution to the success of the conference.

On behalf of the Organising Committee
Vevek Ram
Editor and Program Chair
Pietermaritzburg, September 1996
Organising Committee

Conference General Chairs
Mr Rob Dempster and Prof Peter Warren (UNP)

Organising Chair
Dr Don Petkov (UNP)

Secretariat
Mrs Jenny Wilson

Program Chair
Prof Vevek Ram (UNP)

Program Committee
Prof Peter Wentworth, Rhodes
Dr Milan Hajek, UDW
Prof Derek Smith, UCT
Prof Anthony Krzesinski, Stellenbosch
Dr Don Petkov, UNP
Mr Rob Dempster, UNP
Prof Peter Warren, UNP
# Table of Contents

Foreword i  
Organising Committee ii  
List of Contributors vi  

**Keynote Speaker**

*The Role of Formalism in Engineering Interactive Systems*  
M D Harrison and D J Duke  

**Plenary**

*Industry-Academic-Government Cooperation to boost Technological Innovation and People Development in South Africa*  
Tjaart J Van Der Walt  

*Checklist support for ISO 9001 audits of Software Quality Management Systems*  
A J Walker  

*The IS Workers, they are a-changin'*  
Derek Smith  

**Research**

*Examination Timetabling*  
E Parkinson and P R Warren  

*Generating Compilers from Formal Semantics*  
H Venter  

*Efficient State-exploration*  
J. Geldenhuys  

*A Validation Model of the VMTP Transport Level Protocol*  
H.N. Roux and P.J.A. de Villiers  

**Intelligent Systems**

*Automated Network Management using Artificial Intelligence*  
M Watzenboeck  

*A framework for executing multiple computational intelligent programs using a computational network*  
H L Viktor and I Cloete  

*A Script-Based prototype for Dynamic Deadlock Avoidance*  
C N Blewett and G J Erwin  

*Parallelism: an effective Genetic Programming implementation on low-powered Mathematica workstations*  
H. Suleman and M. Hajek  

*Feature Extraction Preprocessors in Neural Networks for Image Recognition*  
D Moodley and V Ram  

iii
Real-Time Systems

The real-time control system model - an Holistic Approach to System Design
T Considine

Neural networks for process parameter identification and assisted controller tuning for control loops
M McLeod and VB Bajic

Reference Model for the Process Control Domain of Application
N Dhevcharran, A L Steenkamp and V Ram

Database Systems

The Pearl Algorithm as a method to extract information out of a database
J W Kruger

Theory meets Practice: Using Smith's Normalization in Complex Systems
A van der Merwe and W Labuschagne

A Comparison on Transaction Management Schemes in Multidatabase Systems
K Renaud and P Kotze

Education

Computer-based applications for engineering education
A C Hansen and P W L Lyne

Software Engineering Development Methodologies applied to Computer-Aided Instruction
R de Villiers and P Kotze

COBIE: A Cobol Integrated Environment
N Pillay

The Design and Usage of a new Southern African Information Systems Textbook
G J Erwin and C N Blewett

Teaching a first course in Compilers with a simple Compiler Construction Toolkit
G Ganchev

Teaching Turing Machines: Luxury or Necessity?
Y Velinov

Practice and Experience

Lessons learnt from using C++ and the Object Oriented Approach to Software Development
R Mazhindu-Shumba

Parallel hierarchical algorithm for identification of large-scale industrial systems
B Jankovic and VB Bajic
Information Technology and Organizational Issues

A cultural perspective on IT/End user relationships
A C Leonard

Information Security Management: The Second Generation
R Von Solms

Project Management in Practice
M le Roux

A Case-Study of Internet Publishing
A Morris

The Role of IT in Business Process Reengineering
C Blewett, J Cansfield and L Gibson

Abstracts

On Total Systems Intervention as a Systemic Framework for the Organisation of the Model Base of a Decision Support Systems Generator
D Petkov and O Petkova

Modular Neural Networks Subroutines for Knowledge Extraction
A Vahed and I Cloete

Low-Cost Medical Records System: A Model
O A Dami and T Seipone

A Methodology for Integrating Legacy Systems with the Client/Server Environment
M Redelinghuys and A L Steenkamp

Information Systems Outsourcing and Organisational Structure
M Hart and Kavatzandis

The relational organisation model
B Laauwen

The Practical Application of a New Class of Non-Linear Smoothers for Digital Image Processing
E Cloete

A Technology Reference Model for Client/Server Software Development
R C Nienaber

The Feasibility Problem in the Simplex Algorithm
T G Scott, J M Hattingh and T Steyn

Author Index
List of Contributors

Vladimir B Bajic  
Centre for Engineering Research,  
Technikon Natal,  
P O Box 953  
Durban 4000

Ruth de Villiers  
Department of Computer Science and  
Information Systems  
UNISA  
Box 392, Pretoria, 0001

C N Blewett  
Department of Accounting  
University of Natal  
King George V Avenue  
Durban 4001

G J Erwin  
Business Information Systems  
University of Durban-Westville  
Private Bag X54001  
Durban 4000

Justin Cansfield  
Department of Accounting  
University of Natal  
King George V Avenue  
Durban 4001

G Ganchev  
Computer Science Department  
University of Botswana  
PBag 0022  
Gaberone, Botswana

Tom Considine  
Apron Services (Pty) Ltd  
P O Johannesburg  
International Airport  
1600

J Geldenhuys  
Department of Computer Science  
University of Stellenbosch  
Stellenbosch  
7700

Eric Cloete  
School of Electrical Engineering  
Cape Technikon  
Box 652  
Cape Town

Louise Gibson  
BIS, Dept Accounting & Finance  
University of Durban  
Pvt Bag X10  
Dalbridge 4014

I Cloete  
Computer Science Department  
University of Stellenbosch  
Stellenbosch  
7600

Mike Hart  
Department of Information Systems  
University of Cape Town  
Rondebosch  
7700

O A Daini  
Department of Computer Science  
University of Botswana  
Gaborone  
Botswana

M. Hajek  
Department of Computer Science  
University of Durban-Westville  
Pvt Bag X54001  
Durban 4000

Nirvani Devcharan  
Umgeni Water  
Box 9  
Pietermaritzburg  
3200

A C Hansen  
Dept of Agricultural Engineering  
University of Natal  
Private Bag X01  
Scottsville 3209

P J A de Villiers  
Department of Computer Science  
University of Stellenbosch  
Stellenbosch  
7700

J M Hattingh  
Department of Computer Science  
Potchefstroom University for CHE  
Potchefstroom 2520
Boris Jankovic
Centre for Engineering Research
Technikon Natal
P O Box 953,
Durban 4000

Paula Kotze
Department of Computer Science and
Information Systems
UNISA
Box 392
Pretoria, 0001

J W Kruger
Vista University
Soweto Campus
Box 359
Westhoven 2124

A C Leonard
Dept of Informatics
University of Pretoria
Pretoria, 2000

Ben Laauwen
Laauwen and Associates
P O Box 13773
Sinoville
0129

Mari Le Roux
Information technology, development: project leader
Telkom IT 1015
Box 2753
Pretoria 0001

P W L Lyne
Dept of Agricultural Engineering
University of Natal
Private Bag X01
Scottsville 3209

Rose Mazhindu-Shumba
Computer Science Department
University of Zimbabwe
Box MP167
Harare, Zimbabwe

Meredith McLeod
Centre for Engineering Research,
Technikon Natal,
P O Box 953
Durban 4000

D Moodley
Computer Management Systems
Box 451
Umhlanga Rocks
4320

Andrew Morris
P O Box 34200
Rhodes Gift
7707

R C Nienaber
Technikon Pretoria
Dept of Information Technology
Private Bag X680
Pretoria 0001

E Parkinson
Department of Computer Science
University of Port Elizabeth
Box 1600
Port Elizabeth 6000

Don Petkov
Department of Computer Science and
Information Systems
University of Natal
PBag x01
Scottsville 3209

Olga Petkov
Technikon Natal
Box 11078
Dorpspruit 3206
Pietermaritzburg

N Pillay
Technikon Natal
Box 11078
Dorpspruit 3206
Pietermaritzburg
V Ram
Department of Computer Science and Information Systems
University of Natal
PBag x01
Scottsville 3209

H. Suleman
Department of Computer Science
University of Durban-Westville
Pvt Bag X54001
Durban 4000

Melinda Redelinghuys
Department of Computer Science and Information Systems
UNISA
Box 392
Pretoria, 0001

A Vahed
Department of Computer Science
University of Western Cape
Private Bag X17
Bellville 7530

Karen Renaud
Computer Science and Information Systems
UNISA
Box 392
Pretoria, 0001

A Van der Merwe
Computer science and Information Systems
UNISA
PO Box 392
Pretoria, 0001

H N Roux
Department of Computer Science
University of Stellenbosch
Stellenbosch
7700

Tjaart J Van Der Walt
Foundation for Research and Development
Box 2600
Pretoria, 0001

T G Scott
Department of Computer Science
Potchefstroom University for CHE
Potchefstroom
2520

K Vavatzandis
Department of Information Systems
University of Cape Town
Rondebosch
7700

T Seipone
Department of Computer Science
University of Botswana
Gaborone
Botswana

Y Velinov
Dept Computer Science
University of Natal
Private Bag X01
Scottsville 3209

Derek Smith
Department of Information Systems
University of Cape Town
Rondebosch
7700

H Venter
Department of Computer Science
University of Port Elizabeth
Box 1600
Port Elizabeth 6000

Anette L Steenkamp
Department of Computer Science and Information Systems
UNISA
Box 392
Pretoria, 0001

H L Viktor
Computer Science Department
University of Stellenbosch
Stellenbosch
7600

T Steyn
Department of Computer Science
Potchefstroom University for CHE
Potchefstroom 2520

R Von Solms
Department of Information Technology
Port Elizabeth Technikon
Private Bag X6011
Port Elizabeth 6000
A J Walker
Software Engineering Applications
Laboratory
Electrical Engineering
University of Witwatersrand
Johannesburg

P Warren
Computer Science Department
University of Natal
P/Bag X01
Scottsville 3209

Max Watzenboeck
University of Botswana
Private Bag 0022
Gaberone
Botswana
THEORY MEETS PRACTICE:
USING SMITH'S NORMALIZATION IN COMPLEX SYSTEMS

Alta van der Merwe & Profs. Willem Labuschagne
Department of Computer Science and Information Systems
University of South Africa
Pretoria

Abstract

In the middle 80s Smith introduced a new way of deriving fully normalized tables. His method was based on the use of a dependency list and a dependency diagram to produce fifth normal form directly. We introduce the use of end-line, begin-line and in-line bubbles to simplify the diagram for large-scale applications.

Introduction

It is usual to follow a procedural methodology when designing a schema for relational database management systems (RDBMS). During the design phase an entity relationship (ER) model is used to represent the entities and relationships. The designer then transposes the entities' associated data fields into a set of candidate tables and, finally, all candidate tables are transformed step-by-step to fifth normal form by applying normalization guidelines (Date, 1986). The apparent simplicity of this ER-model approach is deceptive inasmuch as the model uses the semantics of entities and not the fields. The designer of the system must have a complete understanding of the normal-form guidelines and any real-world information applicable to the system when transposing the ER-model to candidate tables (Codd, 1970).

In 1985 Henry C. Smith introduced a method to compose fully normalized tables from a rigorous dependency diagram. This method is based on field-related semantics. Because the semantic meanings of its dependency chains are specifically tailored to normal-form guidelines, the model directly and immediately synthesizes into fully normalized tables with completely specified primary keys (PK), foreign keys (FK), and join paths (Smith, 1990).

Because the dependency diagram plays the key role in Smith's method, it is essential that the diagram should be readable. Readability tends, of course, to diminish in proportion to the size and complexity of the application. In this paper we adapt Smith's method for application to complex large-scale systems.

Basic conventions

When designing a dependency model using Smith's method (SM), three documents are used: a dependency model, a schema synthesized from the model and an enterprise data dictionary. We focus on the dependency model and the schema synthesized from the model.

The dependency model is developed in accordance with the following rules:

- Every field name used in the dependency model must be atomic or non-decomposable and may appear only once in the dependency model.
- A single valued dependency (SVD) is diagrammed by drawing a single bubble around the field and using a single-headed arrow to the field dependent on it. See figure 1 for a single dependency A→B (B is determined by A). The originator (A) is called the determinant and each attribute dependent on it is called a singleton.
- Multi-valued dependencies (MVD) arise when a field A determines many values of field B. An MVD is diagrammed by drawing bubbles around field A and B and using a double-headed arrow from bubble A to B (Figure 2). A is called the determinant and B is called the end-multiple bubble. B is called an end-multiple bubble if it contains a field whose multiple values are determined by A; a double-headed arrow points to B and no arrows point from it. Values of A and B must be not null and the combination A,B must be unique.
If MVDs exist between A and B and A and C (with A the determinant) they are diagrammed as in Figure 3. If MVDs exist from A to B and from B to C they form an MVD chain and are modelled as in Figure 4. Values of A, B and C must be not null and the combination A, B, C must be unique. A is called the uplink-determinant bubble, B the determinant bubble and C the end-multiple bubble.

SVDs can be uplinked to an MVD or MVD chain.
- No transitive dependencies are allowed.
- A double bubble is used when a field is a determinant of two different or conflicting dependencies. E.g. if A determines B but A also determines C where C and B do not have anything to do with each other, a double bubble are used to diagram these facts (figure 5).
- \([n]\) at the bottom of fields is used to show that these fields share the same domain values (\(n=1\ldots\)).
- \(<m>\) at the bottom of a singleton field means that more singleton fields can be found at number \(m\) in the dependency list.
- A singleton chain is a series of interlinked bubbles that ends with one or more singleton bubbles, i.e. a bubble with one or more single-headed arrows pointing to it and no arrows pointing from it.
- An end-multiple chain is a series of interlinked bubbles that end with an end-multiple bubble, i.e. a bubble that has one or more double-headed arrows pointing to it and no arrows pointing from it.

Smith’s algorithm to create 5NF relations

1. Consider all the singleton chains. All the fields within all singleton bubbles linked to the same determinant bubble are the non-key attributes of the table. The field within that determinant bubble plus field(s) within all uplink-determinant bubbles form the table’s primary key. Mark the line between the last determinant and all the singletons linked to it, as worked (√).
2. For an end-multiple chain, work upwards from the end-multiple bubble: the fields within the end-
multiple bubble, the determinant bubble and all uplink-determinant bubbles form the table's primary key. Mark the line between the last determinant and the end-multiple bubble linked to it, as worked (√).

Example

![Dependency Diagram]

Figure 6: A simple dependency diagram depicting a student system

**Dependency List**

1. For each student a Student Number (Student#), Name, Address and Telephone must be stored.
2. For each Lecturer a number, Name and Office Number (Office#) must be stored.
3. Each Subject is identified by an Subject_Code and described by a name (Sub_Name).
4. Each Lecturer can have more than one student and each student can take more than one subject.

**5NF relations**

- Student(Student#, Name, Address, Telephone)
- Lecturer(Professor#, Name, Office#)
- Subject(Subject_Code, Sub_Name)
- Class(Student#, Lecturer#, Subject_Code)

**Large-scale applications**

When one applies Smith's method on big complicated systems, the readability of the dependency diagram can be adversely affected in two principal ways, namely by lines that cross one another and by a proliferation of double-bubbles. We will use the MetAIS (Metrological Air Information System) system developed by Netsys International to illustrate the problems.

The MetAIS system was developed for a customer that provides operational flight information to the aviation community. The MetAIS system receives the data from various sources including satellite feeds and renders it available with a good interface in useful format. A typical use might be for a pilot to ask what the weather will be from point A to point B. The system must then gather all the information applicable to the route and draw the pilot's attention to potential problems in regard with meteorological phenomena as well as operational problems.

**Representing multiple dependencies**

The dependency list for the MetAIS system consisted of about 70 different dependencies of importance. To represent all the different dependencies explicitly in one diagram, and still keep it readable, was impossible. We used the rule: *<m>* at the bottom of a singleton field means that more singleton fields can be found
at number m in the dependency list to simplify the diagram. Although this helped, the readability of the diagram remained affected by the crossings of lines from one side of the diagram to another (figure 7).

Figure 7: Lines crossing to represent different relationships

An end-line bubble (ELB) and begin-line bubble (BLB) were created to eliminate the crossing of lines. Let's start with an example to illustrate the use of ELBs and BLBs. Consider dependency list number 7:

7. For Case A every value of field C will uniquely determine field D.

Our problem is that field C is positioned at the bottom lefthand corner and field D at the top righthand corner of the diagram. Linking the two requires crossing various other lines. By making use of an end-line bubble (figure 8) from C to link it to its begin-line bubble counterpart (figure 9), we simplify the diagram a great deal.

Figure 8: End-line bubble

Figure 9: Begin-line bubble

The following conventions govern the use of ELBs and BLBs:
- each ELB and BLB has a number written in it. This number is the same number as the dependency list number associated with it. The ELB and BLBs that are linked must have the same number.
- each ELB must have at least one BLB.

Figure 10: ELB and BLB dependencies
Figure 10 illustrates the use of an ELB and a BLB for the example given in figure 7. It is clear that using ELBs and BLBs to eliminate the crossing of lines renders the diagram more readable.

**Double-bubble representation**

Double-bubbles are used to represent different or conflicting dependencies (Figure 5). The following is an extract from the MetAIS system.

29. Each NSWC has a TTAAii with more than one YYGGgg. For each YYGGgg more than one CCCC and BBB exist and the combination of the four has one ValidTo and one set of Data.
30. Each SWC has a TTAAii with more than one YYGGgg. For each YYGGgg more than one CCCC and BBB exist and the combination of the four has one ValidTo and one set of Data.
36. A low level forecast has a combination of TTAAii, CCCC, BBB, YYGGgg that is unique and identifies a single ValidTo and Data field.
44. For each WC a TTAAii with more than one YYGGgg is stored. For each YYGGgg more than one CCCC and BBB exist and the combination of the four has one Flight_Level, ValidTo and one set of Data.

The diagram for the four dependencies in the dependency list is given by Figure 11.

![Figure 11: Double-bubbles used in the MetAIS system](image)

The fields TTAAii, YYGGgg, CCCC, BBB, ValidTo and Data are repeated for four different conflicting dependencies. We call this a repetitive dependency of degree 4. In general the double-bubble representation of repetitive dependencies of degree greater than 2 impedes the comprehension of the diagram. We propose the insertion of in-link bubbles as an alternative way to represent repetitive dependencies.

Consider the example in figure 12 with the following dependency list:

1. For Case 1 S# can have more than one D# and for D#,S# there exists only one O#.
2. For Case 2 S# can have more than one D# and for D#,S# there exists only one O#.
3. For Case 3 S# can have more than one D# and for D#,S# there exists only one O#.
4. For Case 4 S# can have more than one D#.

![Figure 12: Conflicting dependencies](image)  
![Figure 13: Diagram after 1 iteration](image)
An in-link bubble (ILB) is created between two fields to show the degree of the link. The value 3 in the in-link bubble between D# and O# means that, irrespective of any links to D# or O# and irrespective of any links that exit from D# and O#, this specific link must be used in exactly three iterations of the normalizing algorithm. Each iteration reduces the values inside the in-link bubbles by one. After the first iteration on the example in figure 12, we are left with the diagram in figure 13 and the relation CASE1 (S#, D#, O#) is derived. After the second and third iteration the following relations are derived: CASE2 (S#, D#, O#) and CASE3 (S#, D#, O#).

We are left with the diagram in figure 14.

![Figure 14: Example after third iteration](image)

Note that the link between D# and O# is marked off as worked (✓). A link is marked ‘worked’ when it was used in an iteration and no longer needs to be taken into account in the normalization process. Only the repetitive dependency between S# and D# (of degree 1) is of any further importance and the relation CASE4 (S#, O#) is derived according to the rules.

To make provision for in-link bubbles a small modification must be made to the way of deriving the normalised tables.

1. Consider all the singleton chains. All the fields within all singleton bubbles linked to the same determinant bubble are the non-key attributes of the table. The field within that determinant bubble plus fields within all uplink-determinant bubbles form the table’s primary key. If an in-link bubble exists between two fields, reduce the degree by one in the in-link bubble. In case of a degree of 0, mark the link as worked (✓).

2. For an end-multiple chain, work upwards from the end-multiple bubble: the fields within the end-multiple bubble, the determinant bubble and all fields in the uplink-determinant bubbles form the primary key. If an in-link bubble exists between any two fields, reduce the number by one in the in-link bubble. In case of a degree of 0, mark the link as worked (✓).

It should be borne in mind that any link to a singleton or end multiple without an in-link bubble is taken to be of degree 1. (Such a link must be used in one and only one iteration between the two fields that are linked).

The simplified diagram for the MetAIS system is given in figure 15.

![Figure 12: MetAIS system simplified using ILB.](image)
The relations derived for the system are the following:

NSWC \((TTAAii, YYGGgg, CCCC, BBB, ValidTo, Data)\)
SWC \((TTAAii, YYGGgg, CCCC, BBB, ValidTo, Data)\)
LLF \((TTAAii, YYGGgg, CCCC, BBB, ValidTo, Data)\)
WC \((TTAAii, YYGGgg, CCCC, BBB, ValidTo, Data, Flight\_Level)\)

Note that it is not strictly necessary to show that the degrees of the links between TTAAii, YYGGgg and CCCC are 4. TTAAii, YYGGgg and CCCC are uplink-determinants of BBB and according to Smith's algorithm will be repeated automatically for each determinant - end multiple (or singleton) repetition.

Conclusion

Many systems are designed over a long period, and new programmers may be added to the team at a late phase of the development. One of the most difficult things for these programmers is to understand the data processes and how the data interlink. The use of diagrams for the system designer may sometimes entail tedious effort, but as a documentation tool for the user and programmers added to teams, the advantages are worth the effort, provided only that the diagrams remain readable. By employing end-line and begin-line bubbles the crossing of lines can be eliminated, and by inserting in-line bubbles the representation of repetitive dependencies can be simplified, thus improving the readability of diagrams involved in large-scale applications.

References


