South African
Computer
Journal
Number 20
December 1997

Suid-Afrikaanse
Rekenaar-
tydskrif
Nommer 20
Desember 1997

Computer Science and Information Systems

Rekenaarwetenskap en Inligtingstelsels

The South African Computer Journal
An official publication of the Computer Society of South Africa and the South African Institute of Computer Scientists

Die Suid-Afrikaanse Rekenaartydskrif
'n Amptelike publikasie van die Rekenaarverenging van Suid-Afrika en die Suid-Afrikaanse Instituut vir Rekenaarwetenskaplikes

Editor
Professor Derrick G Kourie
Department of Computer Science
University of Pretoria
Hatfield 0083
dkourie@dos-lan.cs.up.ac.za

Subeditor: Information Systems
Prof Lucas Introna
Department of Informatics
University of Pretoria
Hatfield 0083
lintrona@econ.up.ac.za

Production Editor
Dr Riël Smit
Mosaic Software (Pty) Ltd
P.O.Box 23906
Claremont 7735
gds@mosaic.co.za

World-Wide Web: http://www.mosaic.co.za/sacj/

Editorial Board

Professor Judy M Bishop
University of Pretoria, South Africa
jbishop@cs.up.ac.za

Professor Richard J Boland
Case Western Reserve University, USA
boland@spider.cwrv.edu

Professor Ian Cloete
University of Stellenbosch, South Africa
ian@cs.sun.ac.za

Professor Trevor D Crossman
University of Natal, South Africa
crossman@bis.und.ac.za

Professor Donald D Cowan
University of Waterloo, Canada
dcowan@csg.uwaterloo.ca

Professor Jürg Gutknecht
ETH, Zürich, Switzerland
gutknecht@inf.ethz.ch

Professor R Nigel Horspool
University of Victoria, Canada
nigelh@csr.csc.uvic.ca

Professor Fred H Lochovsky
University of Science and Technology, Hong Kong
fred@cs.ust.hk

Professor Kalle Lyytinen
University of Jyvaskyla, Finland
kalle@cs.jyu.fi

Doctor Jonathan Miller
University of Cape Town, South Africa
jmiller@gsb2.uct.ac.za

Professor Mary L Soffa
University of Pittsburgh, USA
soffa@cs.pitt.edu

Professor Basie H von Solms
Rand Afrikaanse Universiteit, South Africa
basie@rkw.rau.ac.za

Subscriptions

	Annual	Single copy
Southern Africa:	R50,00	R25,00
Elsewhere:	$30,00	$15,00

An additional $15 per year is charged for airmail outside Southern Africa

to be sent to:
Computer Society of South Africa
Box 1714 Halfway House 1685

Phone: +27 (11) 315-1319 Fax: +27 (11) 315-2276

Editorial Notes

For two reasons, this edition of SACJ is far later than it ought to have been. The first reason is that there have been some personnel changes in the editorial team. Lucas Introna, having continued for some time as IS editor after transferring to London, asked to be relieved of his duties. Niek du Plooy has kindly agreed to fulfill this role in a temporary capacity until a suitable replacement for Lucas can be found. Due to work pressure, Riël Smit has also withdrawn as production editor, and has been replaced by John Botha. SACJ owes the two retired members a huge debt of gratitude. During his period of tenure, Lucas did sterling work in setting and maintaining a solid standard for IS contributions. Riël put SACJ on a LaTeX path, and has laboured diligently to produce an aesthetically pleasing product. Thanks are also due to Niek and John for their willingness to take over in their respective roles. Until further notice, IS contributors may forward their submissions directly to Niek at his address given on the front inside cover. I shall put successful authors in touch with John for further instructions regarding final preparation of their manuscripts.

The second reason for a delay in this edition has to do with authors who have not scrupulously followed guidelines for producing their final submissions. There have been a variety of problems ranging from missing citations and inappropriate production of figures to incompatible electronic file submissions. All of this, coupled with our new production editor (who—despite an extremely busy schedule—has valiantly climbed a steep LaTeX learning curve) has resulted in an edition that should have been out to press several weeks earlier.

The editorial team will be giving attention to the general matter of format and submission procedures in future. SACJ's citation and reference methods are somewhat archaic and will probably be revised. All the necessary information will be provided on the new SACJ web site at `www.cs.up.ac.za/sacj/`. The site will also contain abstracts of articles in this and future editions.

These are times of conflicting stresses on both the academic and industrial IT communities. They are being felt somewhat more acutely in Southern Africa (and presumably in other developing countries) than in the developed world. Internationally there is tendency to cut back on state financing of universities and a seemingly insatiable demand for IT graduates. Many companies snap up new graduates at attractive salaries, positively discouraging full-time postgraduate studies. International recruitment agencies scour the South African scene for qualified candidates, luring some of our most promising young professionals out of the country. Job-hopping, a drift from academia to industry and from local industry to USA or European industry seems to be the order of the day. Despite the availability of private colleges and institutes, virtual or otherwise, there is a rush of students to university and technikon IT departments, all hoping to get at the IT honey-pot. University administrations are struggling to correct the structural deficiencies of the past and to provide IT departments with sufficient resources to cope with demand. As editor of SACJ, I have no particular competence authoritatively to sum up or analyze these tendencies, but it does seem to me desirable that someone ought to do so. Bodies such as SAICSIT, the CSSA, university authorities, IT industry and state representatives ought actively to pursue joint strategies to ensure that our IT departments are properly resourced and that (non-Zuma) measures are taken to retain graduates in the country. It seems almost redundant to attempt to spell out the consequences of inactivity.

Derrick Kourie
EDITOR

Mobile Agents at ISADS 97

I Vosloo
Object Technology Expertise Centre, Computer Science, Pretoria University, Pretoria, Hatfield 0083

Abstract

This communication gives an overview of the field of mobile agents as presented at various events held in conjunction with a recent international symposium in Germany. The nature of mobile agents and the key concepts involved are discussed in broad terms. Technical themes, such as migration, communication and security, are described in more detail, and a synopsis of an assessment presented by GMD FOKUS researchers of current mobile agent technologies is provided. The status quo on standardisation in this field concludes the communication.

1 Introduction

The relative immaturity of mobile agents as a field of study is evidenced by the uncertainty surrounding definitions as to what agents are and by the disparate views on their benefits and weaknesses. The International Symposium on Autonamous Distributed Systems (ISADS) was held in Berlin this year. It was preceeded by a tutorial and a workshop on mobile agents hosted by the German Informations- und Kommunikationstechnologie Verbund e. V. (IKV) and the Gesellshaft für Informatik. These events gave a broad overview of the current status of mobile agents and also served as a useful introduction to the concepts germane to the field.

What follows is a brief summary of the overview presented in Berlin, augmented by some insights gained there.

2 Agents: An Overview

There is wide-ranging dispute about the precise definition of an agent, but it is safe to say that agents are generally seen as software artefacts to which certain tasks can be delegated. An agent should then be able to tend to this task with a sizeable degree of autonomy. The somewhat philosophical motivation for the agent concept is that such entities can help people and other software artefacts cope with an overflow of information and, in general, with the complexity that these people and software artefacts have to deal with. This could allegedly be accomplished by agents since agents enable one to delegate responsibilities.

The delegation of responsibility to software artefacts places significant importance on security and correctness; both of these are problems which are not yet satisfactorily solved in the field of mobile agents (in fact, no-one even made mention of correctness or the modelling of agent-based systems).

The events in Berlin focused on the mobility aspect of agents. This is, however, just one characteristic of an agent. Intelligence is another. Agents can have these characteristics to greater or lesser degrees and the mobility focus of these events does not mean that the discussion exclude agents with other characteristics.

Other basic mobile agent (MA) concepts include the ideas of agent platforms and agencies. An *agent platform* is the software providing the functionality needed to manage, transfer and execute mobile agents. An *agency* is a domain or locality on such a platform where the platform provides certain capabilities.

There are two types of code mobility. *Remote execution* is a kind of mobility where the agent code and data are transferred to the target system before it starts to execute. It will then be started on the target system, complete its task and stop execution there. *Migration* is the term used when agents are transferred while they are being executed. This implies that the agent can have control over its mobility. (It can choose when and where to migrate.) Migration also requires the execution state of the agent to be transferred along with the agent code and data. The scope for application of the remote execution technique is limited to dynamic software distribution, whereas migration is a much richer mechanism that can be applied to process transactions in a distributed fashion. By and large, the MA community assume that truly mobile agents would have migrating functionality instead of merely being remotely executed.

The tutorial hosts expect to derive the following benefits from using mobile agents:

- asynchronous task execution;
- reduction of network traffic and client processing power;
- robustness (especially with regard to network or server availability);
- decentralisation of processing, control and management;
- flexibility;
- network-centric computing; and

- adaptation to and ability to combine with existing software.

Agent mobility requires support for things like the naming of agents, the notion of locality (possibly virtual locality), execution of dynamic code, persistence and security (for agents and hosts alike). These capabilities could be implemented as part of the agents or as part of the supporting platforms. Putting the functionality in the agents makes them 'fat' (large in terms of code and therefore costly to transfer); putting the functionality in the supporting platforms requires stronger standardisation needs. (The benefits expected from agents assume a truly open environment.)

There are a number of technical problems still to be solved. These concern security (of agents and platforms), task execution control (or distributed transaction processing), fault management (e.g., orphan detection and management), performance, heterogeneity of the environments, transfer overhead trade-offs, and access to and integration with existing services like the Internet and CORBA services.

The directions currently important to the MA community concern the integration of MA concepts and capabilities into the existing distributed object computing world, dominated by CORBA. The community also targets the standardisation of enabling platforms for agents. To this end a number of companies have submitted a proposal to OMG. (More on this will follow below.)

3 Migration

There are different methods to migrate an agent with its execution state to another machine. The ideal method, called *pre-emptive migration* (not a widely used term as yet), is where the underlying execution environment has thorough knowledge about the agent and its complete execution state. It can then suspend the agent process, transfer its code, data and complete execution state to the other machine, where its execution state can be restored and where it can be rescheduled. All of this can happen transparently to the programmer.

This sort of specialised agent-enabling support only exists in execution environments that are specifically developed for agents. Other platforms are more widely available, but generally do not have the functionality to make this possible.

Something similar to program state is object state. It is easy to transport the state of objects if they are based on explicit state machines. The state of the state machine can then be transported.

In general purpose environments like Java a totally different approach is taken: it can be required of the objects themselves to have the necessary intelligence to save and restore their own execution state (whatever that may mean to them). (Java only supports remote execution.) An example of this type of migration is the implementation of Aglets Workbench. It uses an event-driven object model and expects all its agents (Aglets) to have a number of predefined methods customised for each object. These include methods like: onCreation(), onArrival(), onDispatching(), and so forth. Aglets, being Java applets, each have a run() method. Migrating an object involves a request to the underlying system which will in turn call the onDispatching() method of the object. This method should be used by the object to collect enough of its own execution state information in such a way that it can be sent to another system, using Java's serialisation capabilities. On the receiving side, the underlying system will call the object's onArrival() method. This method should then rebuild the execution state from the saved information and call the run() method of the object to start it up again.

4 Communication

One of the areas where there are still widespread differences among the various agent platforms is that of communication. A number of different communication models exist. Some platforms allow agents to communicate with other agents without regard for the locality of these other agents. Other platforms only allow agents to communicate with other agents in the same agency. Telescript introduced the notion of *meetings*. Meeting-based communication requires of agents wishing to communicate to first engage in a meeting. Such meetings could be restricted to apply only to agents in a single location. (Although locations can be virtual: consisting of more than one physical location.)

Agents also need to communicate with the host systems that they visit. The best model here is to have a number of *services* in an agency. Services look like agents, but they are not mobile and agents can be allowed access to local resources via these services. It is generally regarded as a security risk to let agents have direct access to resources, even simple ones like the terminal. Services can be utilised to do security checks, to prioritise requests and so forth.

During the course of the workshop, it was quite clear that there are myriad ideas about communication afoot and also nuances to those ideas. Surprisingly, no-one saw fit to assess and compare all these approaches. It is important that these different models of communication should be enumerated and assesed, and that unanimous decisions about them should be reached in order to achieve standardisation.

5 Security

Although no papers were submitted on the topic of security, a panel discussion was held instead. As stated earlier security is regarded as a high-priority problem because the very nature of agents implies that they will carry some responsibility, making them prime candidates for malicious attacks. The panel members were: Andreas Pfitzmann (University of Dresden), Joachim Posegga (Deutsche Telekom AG, Darmstadt), Lars Rasmusson (SICS, Stockholm) and Fritz Hohl (University of Stuttgart).

Agent security consists of security in three areas: the agent has to be transferred securely; the agent platforms and the resources on them have to be protected against possible malicious agents; and the agents have to be protected against possible malicious platforms.

Traditional encryption schemes can be applied to cover the secure transferral of agents. The problem of protecting platforms is also manageable since platforms can control the access agents have to the resources provided (and the senders of agents can be verified). The problem of protecting an agent against a malicious platform, however, is not solvable, according to Pfitzmann. The essential interpreted nature of agents aggravates this problem. The technologies discussed during the tutorial and workshop provided different levels of support for the 'solvable' security problems, but no work has been presented (not even purely academic work) to solve the last one. Pfitzmann is convinced that the problem is not solvable and is leading work aimed at proving his belief.

Security has always been a very specialised field and the extent to which it is necessary depends on the application. It also implies large computational overhead. Posegga and Hohl made it quite clear that the electronic marketplace requires absolutely robust, guaranteed security due to the speed and openness (especially the easy access to money) that characterise the electronic marketplace. Agent technology will never be able to provide this (according to Pfitzmann).

Rasmusson takes an entirely different approach to the security problem using the real world as a model. He proposes security by social control: building support into the agent model for notions such as reputation and trust. The security provided by this model redefines the problem to one of minimising fraud instead of the traditional aim of eliminating it. After his opening address a lively debate ensued about whether or not the social control metaphor of the real world can actually be extrapolated to the electronic world. To some the outcome of the debate was a realisation that this might be the only model we can hope for and that we would be forced by the elements it contains to deal with risk and uncertainty in the electronic world. Others still refuse to swallow this rather bitter, but potentially interesting, pill.

Whatever happens on the security front, it seems that security will play a major role in the level of applicability of agents in the different industries in the real world. Electronic commerce, as conceived of today, seems not to be such a good application domain for mobile agent technology in an open marketplace after all. It seems that agent technology would shift more into application domains like telecommunications where damage resulting from security breaks would be relatively unimportant and localised.

6 Mobile Agent Technologies

During the tutorials an overview of current mobile agent technologies was presented. A quick summary derived from the tutorial slides follows.

Apart from the general information provided about technologies, the technologies were specifically compared and assessed with regard to architecture and the functionality provided.

The requirements considered for MA platforms included:

- Modelling requirements such as the adequacy of the agent model; applicability to more specific and more general problems; completeness; and simplicity.
- Generic requirements such as performance, efficiency, openness, scalability and maintainability.
- Functional requirements, falling into several categories. First is the creation, execution, control and termination of agents. Mobility functionalities include naming, addressing, migration and location. Communication functionalities include location of peers and communication facilities for various combinations of entities (inter-agent, agent/user, etc.) The functional requirements also covered security aspects.

Platforms covered:

- **Agent Tcl** is based on the Tcl/Tk language. It is lightweight, extensible and freely available. It has a number of deficiencies resulting mainly from the language used. It is assessed as being a good research system, but too weak for production environments. Development is still underway.
  It supports the pre-emptive migration, inter agent communication using TCP/IP (agents have to enter into a meeting or can just use asychronous messages), and some security features. Agents have access to dangerous functions once they are allowed into a system (like GUI functions etc). In principle it should be possible to replace the Tcl interpreter with that of another language.
- **Telescript** is a robust, secure, commercial agent platform which contains a number of valuable concepts. The trade-off as usual is that it is resource

intensive. The workshop presenters are of the opinion that it will not succeed because it is proprietary and closed, and has been poorly marketed. It supports a wide range of functionalities including an object-oriented language, pre-emptive migration, security, persistence and WWW integration. It has recently been repackaged as *'Tabriz'*.

- **Java** is also in the running because, even though it is not an agent technology, it can be used as an enabling technology and has relevant built-in functionality, namely: code mobility, network communications, strong coupling to the Internet and WWW, and security. The Java language is a good OO language and the language and its virtual machine are found everywhere. Its momentum and universal presence are the main factors that make it the technology of choice for MAs even though it does not support pre-emptive migration or transparent persistency.
- **Aglets Workbench (AWB)** is a commercial mobile agent platform developed by IBM and based on Java. The workshop presenters judged it the top platform contender at the moment. (However, one would have to bear in mind that many of them are working closely with IBM.) Among other things it comprises a Java framework for the agent programmer, called J-AAPI (Java Aglet API). Pre-emptive migration is not supported -- the transfer of execution state is left to the objects as explained in the earlier example. IBM has submitted J-AAPI and J-ATCI (Java Agent Transfer and Communications Interface) interfaces to OMG for standardisation.
- **OMG CORBA**. OMG has issued a Request for Proposal (RFP) for a mobile agent facility (MAF) to be added to the standard services of CORBA. It would need to add mobility and location awareness to CORBA objects, making use of the other basic CORBA services like security and persistence.
- **MAGNA** is an MA framework for telecommunication applications based on CORBA and Java. Developed by GMD FOKUS, the workshop presenters, its main goal is to provide a product conforming to the OMG MAF.

Some lesser-known agent implementations were introduced at the workshop itself. These were not part of the assessment given. However, references to information about them are supplied below.

7 Standardisation

As mentioned earlier, standardisation is singled out as one of the primary aims of the MA community at the moment. There is consensus that without uniform, open standards agents would not be very successful. Three bodies are involved in the process: OMG, FIPA and the Agent Society.

OMG has issued an RFP for a mobile agent facility in November 1995. The OMG expects a specific set of requirements from the MAF. Its focus is on providing the concepts of locality and migration on top of the hitherto location-transparent CORBA environment. It assumes that other CORBA services (like security and persistence) would be used by the MAF to meet all requirements of an MA environment. In January 1997 IBM, The Open Group and GMD FOKUS submitted a joint proposal which has not been accepted by the OMG. These three companies plan a new submission based on the feedback from OMG in collaboration with two more companies (Crystaliz and General Magic) in June 1997. This was the only response to the OMG RFP to date.

The Foundation for Intelligent Physical Agents (FIPA) is a relatively new body, formed in 1996. FIPA has members from the industry, similar to OMG. It is significantly smaller than OMG and has as goal the broader agent technology field, not just mobility. It places more emphasis on intelligence (as the name indicates).

The Agent Society was founded in 1996 with the aim of facilitating the transfer of information between the players in the agent field. They publish and promote information and technologies concerning agents.

During the workshop a number of people voiced concern about the fact that FIPA and OMG could develop competing standards similar to what happened in the networking field between the IETF and ISO. It is interesting to see that a number of the main players tend to belong to more than one of these bodies. IBM, being one of the most active in the field, belongs to all three and plays a leading role in all three. Some believe that this fact (that the companies belong to more than one organisation) will ensure that the emerging standards would eventually be similar, if not identical. (For example: IBM does not develop its proposals for the different bodies separately; they develop one proposal and submit the relevant subsets of that proposal to FIPA and OMG.) Others believe that this in not enough and are already making provision for functionality that would help their agent platforms to interact with other agent platforms. One paper presented at the workshop explored techniques to do just that.

8 Conclusion

The field of mobile agents is at an interesting juncture: a number of different people, with diverse points of view on implementation, conceptual definitions and functionality, have developed enabling software for mobile agents. However, these are mostly laboratory-like environments that

only enable the very basic functionality needed to create agent systems. None of these systems could solve the additional real-world problems like the need for agent security and robust distributed transactions. Unfortunately, without these the applicability of mobile agents seems to be limited and therefore the industry support for mobile agents is also limited. The people involved are nevertheless enthusiastic and although the advantages of mobile agents are not yet convincing in real-world terms, the enthusiasts claim to be working diligently towards two main goals. The first goal is standardisation of all the concepts and platforms, and the second is to try and solve the more complex needs demanded by real-world applications. Java also has very wide visibility in the MA community, whose members use it mainly because of its sizeable momentum which they hope to utilise to secure more general support for mobile agents.

Acknowledgements

Large parts of this communication are derived from the tutorial presented by Thomas Magedanz and Sven Krause from GMD FOKUS. Thanks to OTEC at the University of Pretoria for financial support making the trip to Germany possible.

References

1. K Rothermel and R Popesco-Zeletin (eds). *Lecture Notes in Computer Science 1219: Mobile Agents '97.* Springer Verlag, Berlin/Heidelberg.

The list of World Wide Web references below have been extracted as-is from the materials supplied at the events in Berlin for further reference.

Overview Pages:
On mobile code:
http://www.w3.org/pub/WWW/MobileCode/
University of Aberdeen:
http://www.csd.abdn.ac.uk/~pedwards/agents.html
UMBC (Tim Finin):
http://www.cs.umbc.edu/agents
The Software Agents Mailing List:
http://www.cs.umbc.edu/agentslist/

Specific technologies and projects:
Telescript/Tabriz:
http://www.genmagic.com/Telescript
AWB:
http://www.trl.ibm.co.jp/aglets
Java-to-go:
http://ptolemy.eecs.berkeley.edu/dgm/
javatools/java-to-go/

AgentTcl:
http://www.cs.dartmouth.edu/~agent/

Cyberagents:
http://www.wbs.com/cyberagent/index.html
Tacoma:
http://www.cs.uit.no/DOS/Tacoma
MA Projects:
http://www.informatik.th-darmstadt.de/~fuenf/
work/agenten/agenten_e.html

Standards organisations:
OMG:
http://www.omg.org
OMG/MAF:
http://www.omg.org/library/schedule/CF_RFP3.htm
FIPA:
http://www.networking.ibm.com/iag/iagfipa.html
Agent society:
http://www.agent.org

Notes for Contributors

The prime purpose of the journal is to publish original research papers in the fields of Computer Science and Information Systems, as well as shorter technical research notes. However, non-refereed review and exploratory articles of interest to the journal's readers will be considered for publication under sections marked as Communications or Viewpoints. While English is the preferred language of the journal, papers in Afrikaans will also be accepted. Typed manuscripts for review should be submitted in triplicate to the editor.

Form of Manuscript

Manuscripts for *review* should be prepared according to the following guidelines.

- Use wide margins and $1\frac{1}{2}$ or double spacing.
- The first page should include:
  - title (as brief as possible);
  - author's initials and surname;
  - author's affiliation and address;
  - an abstract of less than 200 words;
  - an appropriate keyword list;
  - a list of relevant Computing Review Categories.
- Tables and figures should be numbered and titled.
- References should be listed at the end of the text in alphabetic order of the (first) author's surname, and should be cited in the text in square brackets [1–3]. References should take the form shown at the end of these notes.

Manuscripts accepted for publication should comply with the above guidelines (except for the spacing requirements), and may be provided in one of the following formats (listed in order of preference):

1. As (a) LaTeX file(s), either on a diskette, or via e-mail/ftp – a LaTeX style file is available from the production editor;
2. As an ASCII file accompanied by a hard-copy showing formatting intentions:
   - Tables and figures should be original line drawings/printouts, (not photocopies) on separate sheets of paper, clearly numbered on the back and ready for cutting and pasting. Figure titles should appear in the text where the figures are to be placed.
   - Mathematical and other symbols may be either handwritten or typed. Greek letters and unusual symbols should be identified in the margin, if they are not clear in the text.

   Contact the production editor for markup instructions.
3. In exceptional cases camera-ready format may be accepted – a detailed page specification is available from the production editor;

Authors of accepted papers will be required to sign a copyright transfer form.

Charges

Charges per final page will be levied on papers accepted for publication. They will be scaled to reflect typesetting, reproduction and other costs. Currently, the minimum rate is R30-00 per final page for LaTeX or camera-ready contributions that require no further attention. The maximum is R120-00 per page (charges include VAT).

These charges may be waived upon request of the author and at the discretion of the editor.

Proofs

Proofs of accepted papers in category 2 above may be sent to the author to ensure that typesetting is correct, and not for addition of new material or major amendments to the text. Corrected proofs should be returned to the production editor within three days.

Note that, in the case of camera-ready submissions, it is the author's responsibility to ensure that such submissions are error-free. Camera-ready submissions will only be accepted if they are in strict accordance with the detailed guidelines.

Letters and Communications

Letters to the editor are welcomed. They should be signed, and should be limited to less than about 500 words.

Announcements and communications of interest to the readership will be considered for publication in a separate section of the journal. Communications may also reflect minor research contributions. However, such communications will not be refereed and will not be deemed as fully-fledged publications for state subsidy purposes.

Book reviews

Contributions in this regard will be welcomed. Views and opinions expressed in such reviews should, however, be regarded as those of the reviewer alone.

Advertisement

Placement of advertisements at R1000-00 per full page per issue and R500-00 per half page per issue will be considered. These charges exclude specialized production costs which will be borne by the advertiser. Enquiries should be directed to the editor.

References

1. E Ashcroft and Z Manna. 'The translation of 'goto' programs to 'while' programs'. In *Proceedings of IFIP Congress 71*, pp. 250–255, Amsterdam, (1972). North-Holland.
2. C Bohm and G Jacopini. 'Flow diagrams, turing machines and languages with only two formation rules'. *Communications of the ACM*, 9:366–371, (1966).
3. S Ginsburg. *Mathematical theory of context free languages*. McGraw Hill, New York, 1966.

South African Computer Journal

Number 20, December 1997
ISSN 1015-7999

Suid-Afrikaanse Rekenaar-tydskrif

Nommer 20, Desember 1997
ISSN 1015-7999

Contents