An open peer-to-peer based platform for scalable multimedia communication

Frédéric Iterbeke, Stijn Melis, Bart De Vleeschauwer, Tim Wauters, Filip De Turck, Bart Dhoedt, Piet Demeester
Ghent University - IBBT - IMEC, Department of Information Technology
Gaston Crommenlaan 8 bus 201, 9050 Gent, Belgium

Bart Theeten, Thierry Pollet
Alcatel-Lucent Bell Research and Innovation, Service Platforms Division
Copernicuslaan 50, 2018 Antwerp, Belgium

Abstract

This article describes an open platform for multimedia applications on a peer-to-peer network. A Voice-over-IP application was developed on it as proof-of-concept. The application allows users to call each other, to publish and query presence information, and to specify whether to allow or deny communication with other users at any given time. For session setup and management the Session Initiation Protocol is used, while audio communication is handled by the Real-Time Transport Protocol. User data is persistently stored on a peer-to-peer network using a distributed hashtable. A software architecture is presented to implement these requirements by means of two highlevel components: the Node, handling all client-side aspects of the system such as calling other users, querying presence and setting preferences, and the SuperNode, handling all server-side aspects such as storage and management of persistent user data, user location, call routing, presence management and monitoring of the underlying peer-to-peer network.

Keywords: presence, peer-to-peer, distributed hashtable, Session Initiation Protocol, Voice-over-IP

1 Introduction

With the introduction of Skype [1], the telecommunication industry feared the end for traditional telephony, which is gradually being replaced by VoIP. VoIP presents a cheaper solution for telephony since its users already pay for their Internet connection, and no additional costs except bandwidth are attached to it. The success of this formula is enormous: as of 2006, Skype has attracted a user base of more than 50 million users [2].

However, Skype uses a proprietary peer-to-peer (P2P) protocol, thus making it difficult for similar programs to interoperate with the Skype network. This is possible by using gateways between these applications' networks, but this approach lacks scalability. To provide an even better and more global service, an open Skype-like system is necessary. This system should use standardised technologies allowing multiple (VoIP) applications to share the same user base, eliminating the need for gateways to cross-connect application networks. Another shortcoming of the Skype system is its limitation with regards to customizing user presence and preferences. Only basic presence such as user status (away, busy, offline, etc.) and a buddy list is provided. It would be desirable to have added functionality, for example allowing users to block certain users at certain times (e.g. colleagues cannot call a user outside of working hours).

With regards to VoIP, the Session Initiation Protocol (SIP)[3] stands out amongst other protocols due to its open character. SIP is a standardised protocol used to set up sessions between users. Currently, SIP-based communication networks rely on central servers to guarantee their functionality. The combination of SIP and P2P is an active area of research [4], the goal of which is to create an open P2P SIP-based system.

In this article, we investigate the possibility of a completely decentralised and structured P2P SIP-based system. The architecture should provide an open, extensible P2P platform on which to run high-level applications such as VoIP, filesharing and instant messaging. We also examine how this platform could be used to support advanced presence capabilities. As a proof of concept, we implemented a VoIP application on top of the presented system.
After a summary of related work, we sketch a small-scale use case where the above system can be used, and the requirements involved. Next we zoom in on the technologies used to create the platform as well as the application. We then present its software architecture and relate it to the mentioned used case. Finally, we analyse the required amount of resources the participants of the system should possess to guarantee their reachability.

2 Related work

Combining SIP with P2P is a hot topic. Namely the P2PSIP working group [4] is aimed at researching and creating a SIP-based DHT overlay and providing proof-of-concept applications for it. In particular, [5] describes a SIP/SIMPLE-based P2P communication system that uses SIP messages for intra-DHT traffic based on the Chord algorithm. EarthLink SIPShare [6] provides a file-sharing architecture with fully distributed content search based on P2P over SIP. Ongoing research is also being conducted on several aspects of integrating SIP and P2P [7]. The industry also shows a lot of interest in this area, and is providing mobile P2P applications that also use SIP, such as Dakama [8]. Many companies are working on similar projects, using P2P to leverage traditional server-based download architectures. Joost[9] and Zattoo[10] use a combination of P2P and server-based streaming to deliver video content to end users. The long-term objective of these projects is to offer low-cost alternatives to cable TV. They are trying to achieve for video and television what Skype achieved for telephony. BitTorrent has recently started a service called BitTorrent DNA[11] that offers client companies a content delivery network based on the BitTorrent P2P protocol to reduce the cost of serving large files on standard webservers. Contrary to the platforms mentioned above, our solution is based on an architecture that is open as well as generic, so that multiple service components for multimedia communication and distribution can be easily built in.

3 Use Case

We investigated the use of the P2P platform in a partly or entirely mobile network. This network could incorporate mobile nodes such as laptops, cell phones and PDAs, as well as fixed nodes such as desktop pcs, gateways, access points and hotspots. The use of the system is aimed at communities: a relatively small number of users wishing to be reachable to each other during a certain event. Examples include a conference or a festival but also community happenings such as gaming tournaments and the likes. Users could then communicate freely with each other using their preferred device (assuming device capabilities are sufficient).

The proposed system should offer several features. Users should be able to invite each other to a VoIP session, publish presence information and obtain presence from other users. Furthermore, users should be able to define preferences, and thus have the ability to block or unblock certain (groups of) users. They should be able to login from different locations and devices, and the system should guarantee the consistency of the users' persistent data. These features should be reliably integrated in a P2P network, i.e. the system should be able to cope with peers joining and leaving the underlying network at unpredictable rates, more commonly known as 'churn'. It should also be an open system to facilitate interoperation with other standards-based applications and ease development of more advanced services.

4 Main protocols and datastructures

4.1 Session Initiation Protocol (SIP)

SIP is an application-layer control protocol for creating, modifying, and terminating sessions with two or more participants. It can be used to create two-party, multiparty, or multicast sessions that include Internet telephone calls, multimedia distribution, and multimedia conferences. Resources are identified by SIP URIs (i.e. sip:frederic@example.net). SIP is designed to be independent of the underlying transport layer; it can run over TCP, UDP, or SCTP. It is a standardized protocol defined in RFC 3261 [3]. SIP acts as a carrier for the Session Description Protocol (SDP)[12], which describes the media content of the session, e.g. what IP ports to use, the codec being used etc. In its typical use, SIP "sessions" are packet streams of the Real-time Transport Protocol (RTP)[13]. RTP is the carrier for the actual voice or video content itself.
The main SIP messages REGISTER, INVITE, BYE and their possible response messages handle user/resource registration, session setup and session teardown respectively. Further SIP messages such as PUBLISH, SUBSCRIBE and NOTIFY, typically related to presence information are defined in further RFCs [14, 15]. These messages and their responses handle the publication, subscription and notification of user resources, which will often be presence.

4.2 Distributed HashTable (DHT)

Distributed hash tables are a class of decentralized distributed systems that provide a lookup service similar to a hash table: (key, value) pairs are stored in the DHT, and any participating node can efficiently retrieve the value associated with a given key. The responsibility for maintaining the mapping from keys to values is distributed among the nodes, in such a way that a change in the set of participants causes a minimal amount of disruption. The actual mapping is based on consistent hashing of main resource attribute(s), or similar techniques. The obtained hash value, further called a resource ID, shares the same space as the identifiers of the DHT nodes, which we will refer to as node IDs. The DHT node with node ID nearest to a certain resource ID is responsible for holding the mapping for that key.

Most DHTs, such as Chord [16] and Pastry [17], guarantee a lookup complexity of $O(logn)$ where n is the number of participating nodes in the DHT overlay. To meet this guarantee, each node maintains a so-called finger table that holds $O(logn)$ pointers to other nodes. The lookup operation hashes the desired resource attribute to an identifier, and a lookup message for this resource ID is created. The lookup message is then routed to the responsible node as follows: each node forwards the message to the node in its finger table that is numerically closest to the target resource ID (see Figure 1). It can be proven that the message reaches its destination in $O(logn)$ hops.

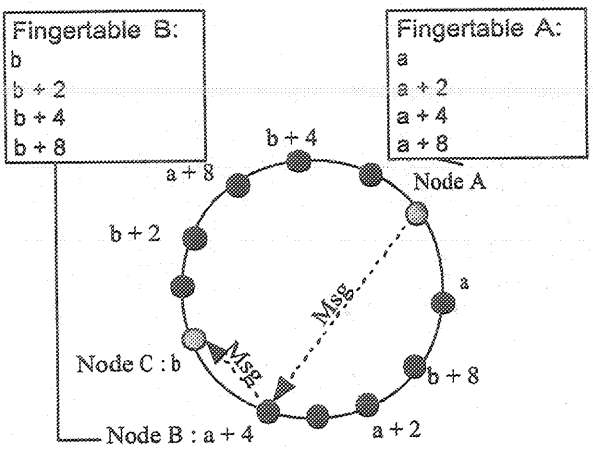

Figure 1. Routing via finger table in a DHT: a message is routed from node A to node C through an intermediate node B

5 Architecture details

The architecture consists of a client component, the Node, and a server component, the SuperNode. Typically, many Nodes will connect to one SuperNode (see Figure 2).

5.1 The Node

The Node is the part of the application the user will be working with. It enables the user to invite other users, maintain a contact list, subscribe to and publish presence information, and define preferences. These preferences provide a means to block or unblock individual users or groups during a period of time. The Node also manages the RTP[13] session once an invitation has been accepted. Preferences are published to the user's responsible SuperNode using a SIP PUBLISH message with a self defined XML content. Users can be added or removed from the contact list using SUBSCRIBE messages with another self defined XML content.

5.2 The SuperNode (SN)

The SuperNode is divided into three large components: the Communication Module, the DHT and the Content Management Module. The Communication Module is responsible for handling all SIP-related communication. A location service and a presence service are built on top of this component. The presence service acts as a Presence Agent (PA), and handles all presence-related SIP traffic (i.e. SIP SUBSCRIBE, PUBLISH and NOTIFY messages)[14]. The location service contains a SIP registrar, which handles SIP REGISTER messages. It is important to mention that it is not required that these services be closely linked to the Communication Module and that they could be implemented in various ways, as long as they are able to process the information contained within the SIP messages mentioned above.
The DHT component handles the distributed side of the application. It is responsible for routing SIP messages destined to users on other SuperNodes, and persistently stores the mappings of users' SIP URIs to their data. User data, such as contact list and preferences, is handled by the Content Management Module and stored in XML format in the DHT. To ensure no data is lost when SuperNodes leave the network, user data can also be redundantly stored on one or more other SuperNodes. If disk space is scarse, user data can also be kept exclusively in RAM memory. The SN itself manages users and redirects them to other SNs when necessary (when responsabilities for users change due to a SuperNode joining or leaving the network).
Various other services could also be supported by this architecture, for example it is possible to build a content service on top of the Management Module to provide advanced functionality related to the user data, i.e. blocking contacts at certain user-defined times. Other possibilities include instant messaging and file sharing services.

6 Scenario implementation

This architecture fulfills all the goals stated earlier in this document, and a preliminary version of this architecture was implemented as a proof-of-concept. In this implementation, users can invite other users to a VoIP session, using the INVITE SIP message with SDP content, which the caller's proxy routes to its destination. The proxy uses the DHT to find the callee's proxy if necessary. Each message will traverse 2 proxy hops at most, and once a session has been set up between caller and callee, all communication, typically consisting of RTP traffic, will flow point-to-point.

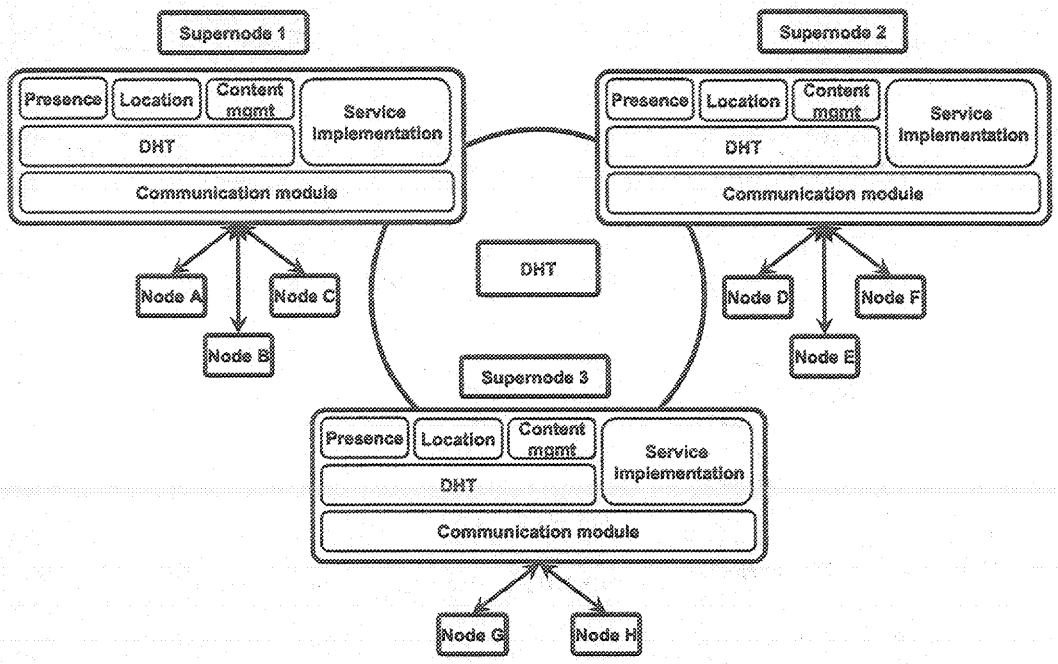

Figure 2. Main architecture: various Nodes connect to SuperNodes.
SuperNodes are linked together in a DHT; Nodes communicate with SuperNodes through the communication module, implemented here using SIP.

Thus, in this case, the SuperNode overlay network is only used for user location and session setup.

Presence information is published using the SIP PUBLISH message with a PIDF[18] content. PIDF is an XML format that represents presence information. Since PIDF only supports an "open" or "closed" presence status, we extended it to allow for other types of presence information (i.e. "Online", "Away", etc.). Users can subscribe to each other's presence information using the SIP SUBSCRIBE message. A subscriber is notified by the notifier's PA which sends a SIP NOTIFY message with the notifier's PIDF as content. This can be done on demand or at regular intervals. An example is shown in Figure 3.

To allow users to block or let through one another, we implemented a firewall-like rule-system. These rules can be defined by the user, and are sent to his responsible SN using a SIP PUBLISH message with a self defined XML document as content. The SN's Content Management Module then updates that user's XML data with the data it received. The content service enforces the rules by checking the sender of messages for that user against the defined rules.

Persistent user data consists of the contact list and preferences. Since a user receives his XML file whenever he logs in, a user can login from any location or device and still have the same contact list and preferences.

The system copes with changes in the network in the following way: whenever an SN loses responsability over a user (due to another SN joining or leaving the network), it redirects the user to its new responsible proxy. The address of this proxy is looked up by its DHT component. When an SN decides to leave the network, it attempts to redirect its users. Since the SN is still in the network (and thus still holds responsability over the users), it can't provide the nodes with their new responsible proxy. Instead, it provides the users with its own address. The users then check if the received address is equal to their current proxy's address. If so, the users perform a delayed redirect, which means they wait for a period of time, and then request the address of their (new) responsible proxy from the bootstrap SN (the SN which started the network). This way, the users can again log in on their new proxy. The redirection and bootstrap calls between components were implemented using RMI. For nodes that have no RMI support, alternatives can be explored, such as using modified SIP messages or web services to realize redirection.

Most parts of the presented system were implemented using free open-source components out of the box, as well as slightly adapted components. The communication module as well as the presence and location services are based on

the NIST implementation of a SIP proxy server [19]. These components all use the NIST SIP reference implementation of JAIN-SIP1.1. For the DHT, a slightly modified version of Bunshin DHT [20] was incorporated. Bunshin itself uses the FreePastry implementation of the Pastry protocol. The Java Media Framework (JMF) provides RTP support, while XML manipulations are implemented using the JDOM library.

7 Evaluation results

Clearly, the key components in the presented platform are the SuperNodes that form the P2P overlay and service the users of the system. We assumed a maximum of 1000 users in the network for the mentioned use case. To have an estimate of the memory footprint of a SuperNode, we modeled the size of user data and the size of the DHT state information per DHT node, depending on the number of DHT nodes in the network. User data size will generally not exceed 2 kB per user.

The Pastry state data (finger table) consists of a neighbour set and a leaf set plus the routing table. These sets are characterized by the base of the nodeIDs, 2^b, and the size of the leafset L, which are set to 16 and 24 respectively in the Bunshin DHT implementation. The leafset contains the $L/2$ numerically closest and larger nodeIds, and the $L/2$ numerically closest and smaller nodeIds. The neighbour set consists of the nodeIds of the L closest nodes according to the proximity metric. The routing table itself contains on average $log_{(2^b)}n$ entries for its prefix-based routing mechanism. The maximum size of the routing table is $2^b * log_{(2^b)}n$, for an entirely filled table.

The total state information kept for routing is the sum of the size of the neighbour and leafsets, augmented by the maximum routing table size. This yields $2^4 * log_{(2^4)}n + 2 * min(n-1, 24)$ as maximum value for the Pastry node routing state.

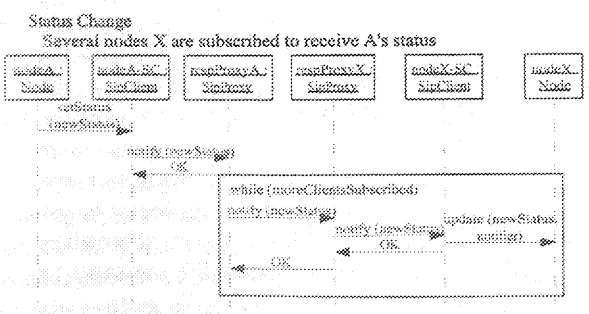

Figure 3. Scenario implementation: a user changes his presence status

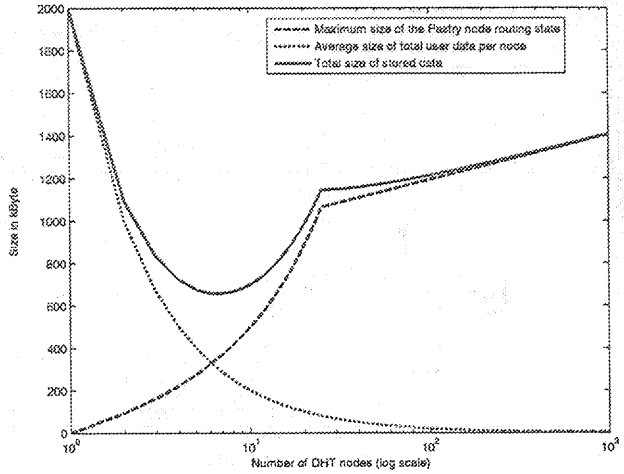

Figure 4. Maximum size of DHT routing state, average size of total user data, and total memory usage per DHT node based on experimental measurements of the prototype

The maximum size of the Pastry node routing state and the average size of user data kept per node, both depending on the number of participating DHT nodes, are depicted in Figure 4. The total amount of used memory is also shown. From the figure, we can derive the optimal number of DHT nodes with respect to memory usage. If 7 DHT nodes are present, the total memory footprint is minimal (657 kB). However, this number of nodes is too small for a smooth operation of the DHT: approximately 143 users would have to connect to each SuperNode. To guarantee DHT performance, there should be enough nodes to populate the neighbour set and the leaf set. If 20 to 25 DHT nodes are present, DHT performance is sufficient while also ensuring a good distribution of user data. Each SuperNode will then manage the user data for only 40 to 50 users, while the (maximum) Pastry routing state size will remain around 1 MB. Adding more DHT nodes will only increase the size of routing state, while the further distribution of user data will leave the SuperNodes underused. Therefore, no more then 25 DHT nodes are needed.

8 Conclusion and future work

In this article we proposed an architecture to implement an open P2P based platform for scalable multimedia applications. We defined a use case where the proposed system can be used. We gave a brief overview of the main technologies and protocols we used. The software architecture consisting of a Node and a SuperNode was presented, and the functionality of both components was explained in more

detail. A more implementation-oriented view was also provided. Finally, we analysed the memory footprint of the proof-of-concept application related to the presented use case, which allowed us to extract the optimal number of SuperNodes for the use case scenario.

Future work will mainly focus on evaluating different DHT possibilities as well as using the presented architecture in a personal content sharing system with an incorporated instant messaging system. Another topic will be the development of thin P2P clients for smart phones.

Acknowledgement

Part of this work has been funded by the IBBT PecMan project.
Filip De Turck acknowledges the F.W.O.-V. (Fund for Scientific Research Flanders) for their support through a postdoctoral fellowship.

References

[1] "Skype. The whole world can talk for free," http://www.skype.com/.

[2] S. Guha, N. Daswani, and R. Jain, "An Experimental Study of the Skype Peer-to-Peer VoIP System," in *Fifth International Workshop On Peer To Peer Systems (IPTPS06)*, 2006.

[3] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," http://www.ietf.org/rfc/rfc3261.txt, June 2002, RFC 3261, Internet Engineering Task Force.

[4] "Peer-to-peer SIP," http://www.p2psip.org/.

[5] D. Bryan, B. Lowekamp, and C. Jennings, "SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System," in *Proceedings of the First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*, 2005.

[6] "EarthLink SIPShare," http://www.research.earthlink.net/p2p/.

[7] K. Dhara, V. Krishnaswamy, and S. Baset, "Dynamic Peer-To-Peer Overlays for Voice Systems," in *Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, 2006.

[8] "Dakama," http://www.dakama.com.

[9] "Joost," http://www.joost.com.

[10] "Zattoo," http://www.zattoo.com.

[11] "Bittorrent DNA," http://www.bittorrent.com/dna/.

[12] M. Handley and V. Jacobson, "SDP: Session Description Protocol," http://www.ietf.org/rfc/rfc2327.txt, April 1998, RFC 2327, Internet Engineering Task Force.

[13] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," http://www.ietf.org/rfc/rfc1889.txt, January 1996, RFC 1889, Internet Engineering Task Force.

[14] A. Niemi and Ed., "Session Initiation Protocol (SIP) Extension for Event State Publication," http://www.ietf.org/rfc/rfc3903.txt, October 2004, RFC 3903, Internet Engineering Task Force.

[15] A. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification," http://www.ietf.org/rfc/rfc3265.txt, June 2002, RFC 3265, Internet Engineering Task Force.

[16] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," in *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, February 2003.

[17] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, November 2001, pp. 329–350.

[18] H. Sugano, S. Fujimoto, G. Klyne, A. Bateman, W. Carr, and J. Peterson, "Presence Information Data Format (PIDF)," http://www.ietf.org/rfc/rfc3863.txt, August 2004, RFC 3863, Internet Engineering Task Force.

[19] "NIST SIP IP Telephony," http://snad.ncsl.nist.gov/proj/iptel/.

[20] R. Mondjar, P. Garca, and C. Pairot, "Bunshin: DHT for distributed applications," in *XIII Jornadas de Concurrencia y Sistemas Distribuidos (JCSD), I Congreso Espaol de Informtica (CEDI 2005)*, Granada, Spain, September 2005.

PROCEEDINGS OF
THE 2008 INTERNATIONAL CONFERENCE ON
PARALLEL AND DISTRIBUTED PROCESSING TECHNIQUES AND
APPLICATIONS

PDPTA'08

Volume I

Editors

Hamid R. Arabnia
Youngsong Mun

Associate Editors

Eltayeb Abuelyaman, Sanjay Ahuja
Jesus Carretero, Steve C. Chiu, Felix Garcia
Jose Daniel Garcia, Hiroshi Ishii, Kazuki Joe
Hyongsuk Kim, Mario Nakamori, Hiroaki Nishikawa
Zornitza Prodanoff, Ashu M. G. Solo

WORLDCOMP'08
July 14-17, 2008
Las Vegas Nevada, USA
www.world-academy-of-science.org

ISBN: 1-60132-082-5, 1-60132-083-3 (1-60132-084-1)
Printed in the United States of America