A Reconfigurable Context-Aware Protocol Stack for Interplanetary Communication

Cathryn Peoples, Gerard Parr, Bryan Scotney and Adrian Moore

University of Ulster
School of Computing and Information Engineering,
Coleraine, Northern Ireland
cathryn@infc.ulst.ac.uk; {gp.parr; bw.scotney; aa.moore}@ulster.ac.uk

Abstract **– This paper presents an approach to improve transmission success in delay-tolerant networks. The Context-Aware Broker (CAB) grants networking autonomy when communicating in challenging environments, which suffer from conditions which are variable and exceed the limits for which terrestrial protocols were designed. Such environments currently require human intervention and the manual configuration of each communication - a seemingly simple decision of when to transmit becomes an issue in deep space due to planet movement. However, manual configuration is becoming unrealistic, given the scale on which communications occur. CAB automates the process by making intelligent decisions before transmission begins, and reconfigures as it progresses. It recognises the dynamic environments through which a transmission may pass and matches protocol capabilities with environmental constraints.**

Index Terms **- Context-awareness, Autonomy, Delay-Tolerant Networking (DTN), Interplanetary backbone, Quality of Service (QoS)**

I. INTRODUCTION

Autonomy empowers the network to be fully responsible for communication decisions, thus removing the need for human intelligence and intervention. This is becoming increasingly important to allow boundaries restricting networking efforts in the 21st Century to be overcome. The most extreme environment is deep space, where long and variable propagation delays are a limiting factor. Similar communication challenges also exist in the Arctic: we are currently involved in an effort to network this environment [1], being one expected to benefit from deployment of the Context-Aware Broker (CAB). Autonomic functionalities are required in such environments, being beyond the reach of human help when unexpected and communication hindering events occur. Solving the problems for these inhospitable environments will allow the next generation of networking to be achieved.

A reconfigurable protocol stack is under development at the University of Ulster, UK. After collecting application and environmental information, it applies contextual knowledge to make intelligent protocol choices within the transport layer. Intelligent choices match characteristics of the operating environment to the transmission protocol, to provide an alternative approach to quality of service in comparison to [2], for example. A single TCP flow over a fast link results in an inefficient use of network resources. The reconfigurable protocol stack aims to use such information in advance to remove the need for human understanding of contextual information, and to automate the process when communicating in extreme environments.

Context-aware configuration of the protocol stack is important where real-time decisions must be made. Our research involves an anticipated deep space scenario; here, we envisage network congestion, more than one communication route, and a human presence in deeper space. More data flows, coupled with limited network architecture and a desire to just push out a signal without prior planning dictates a need for real-time decision-making.

II. CURRENT USE OF CONTEXT-AWARENESS IN DTN

There are several autonomic missions currently under development [3], and NASA's Autonomic Computing initiative will officially be launched between 2020 and 2030 [4]. These missions typically use a number of individual components which communicate with each other, and reconfigure using real-time information. Due to the number of components and decisions, their location, and the need for immediate reaction to unexpected events and real-time configuration, human control of the mission is not an option. Context-awareness which enables autonomic decision-making is therefore empowering a new type of independent mission.

In addition to the development of autonomic missions at Goddard, NASA's Jet Propulsion Labs is involved in another branch of networking in space, the Delay Tolerant Networking effort. This describes the development of techniques for communicating over the interplanetary backbone and is the environment for which CAB has been initially designed. Communication between Earth and deep space could benefit from the integration of autonomy: a DTN bundle [5], for example, has limited choice of what to do when node resources are less than it requires [6]. Autonomic capabilities could be used in advance of reaching the under-resourced node to identify and bypass the constraint. We are therefore confident that this development fills a key research gap.

III. CONTEXT-AWARE BROKER OVERVIEW

We are developing an approach to enable communication autonomy, initially proposed in [7]. The intention is for its deployment in all hardware involved in a communication, including source and destination nodes, and intermediary bundle nodes. The deployment may exist in a base station

sending telemetry commands to a spacecraft approaching Mars; in a node on the surface of Mars; or in a craft orbiting Mars, interfacing with components on the surface of the planet and with humans on Earth. Therefore, while one of the aims is enabling long-distance communication, CAB must also possess capabilities for transmission over typical terrestrial distances. The broker (Fig. 1) will be integrated into the protocol stack between the application and transport layers. Working in a top-down approach, it combines application information with dynamic environmental data to make choices in the transport layer. Choosing the transport protocol in real-time, CAB maximises the opportunity that QoS requirements will be met, given the current network situation. The broker performs a series of evaluations using static and dynamic information: the initial evaluation is initiated on reception of a transmission request from the application layer. 'On-the-fly' evaluations continue on a per-port basis, as the environmental information is dynamically updated.

The broker progresses through four states during communication. States include learning, evaluation, configuration, and abort. During learning, the broker considers application QoS levels and the ability of the network to fulfill these requirements. Several management information bases (MIBs) are used to allow this to happen. The Application MIB is populated once at the beginning of a communication. It contains information on the required transmission characteristics (i.e. real-time/non real-time and synchronous/asynchronous), and acceptable worst-case performance metrics (i.e. maximum acceptable bit error rate and maximum acceptable latency) (Fig. 2). The Environment MIB measures the ability of the network to fulfill the application performance requirements. It records values on the end-to-end, node-to-node, and greater environment surrounding the communication path (Fig. 2). Node performance metrics are recorded on the current node (N), the previous node (N-1), and the next node (N+1). Statistics include performance metrics of the nodes (i.e. queuing delay and buffer capacity) and the link (i.e. propagation delay and packet loss rate). Propagation delay comes from ephemeris – pinging a link in deep space is not an option. The Environment MIB is also used to predict performance at nodes beyond N+1. In addition, sub-MIBs are associated with the Environment MIB. The Inferred Environment MIB retains information, for example, on the predicted number of retransmissions at nodes to be traversed in the future, derived from actual retransmissions and the current volume of traffic. It uses statistics collected to infer additional information. A Historical Environment MIB records a running total of values from the Environment and Inferred MIBs, and is used to indicate network trends as the communication progresses. Finally, the Ephemeris MIB determines occurrences of loss of line-of-sight connectivity between nodes.

The Environment MIB will be standardised and updated as part of the network management system. The broker will access the MIB during traversal of the link to identify changes in performance and take action. The Application MIB will be

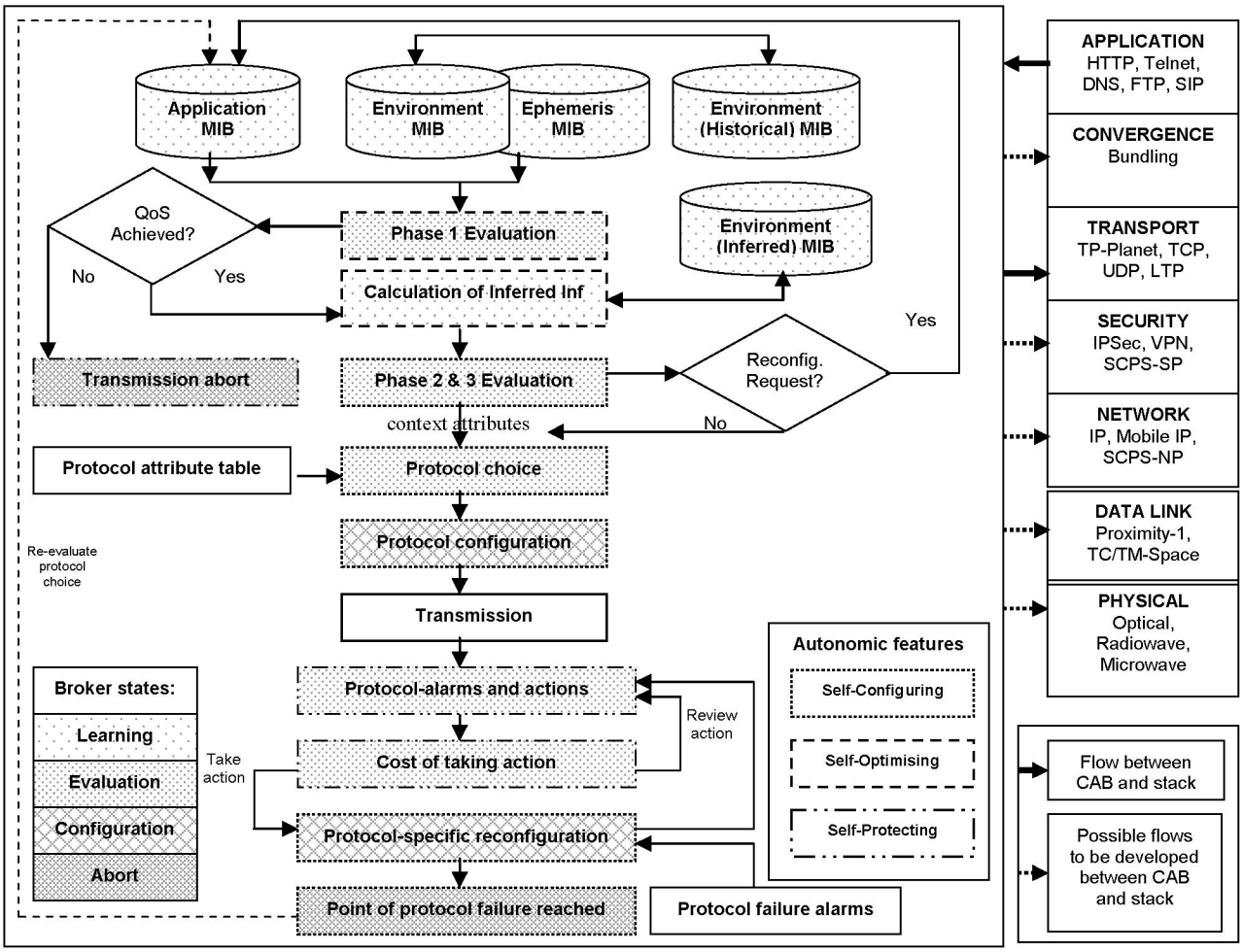

Figure 1. Architecture of Context Broker

retained on the end node. It is the responsibility of the application to define MIB data appropriate for the broker upfront. Where the information is incomplete, transmission will progress on a best-effort basis.

The Application and Environment MIBs are evaluated together to influence protocol choice in the transport layer. There are four phases of evaluation in the context-aware broker. The first phase occurs within the learning state, taking place at a high level on unprocessed MIB data. The aim of this evaluation is to quickly measure the network's ability to provide QoS and to halt transmissions early when this is not the case (Fig. 3). Decisions will be made based on: 1) the end-to-end propagation delay path (when traversed at the speed of light) in relation to the maximum acceptable application latency; 2) line of sight connectivity between end nodes; and 3) node battery power. Subsequently, context-aware capabilities allow communication success or failure to be pre-empted at a high level.

The second evaluation phase uses all MIB data to evaluate the network's ability to support application requirements. Decisions are based on the current and predicted environment situation and the ability to achieve QoS. Evaluations ask, can the application cope with: 1) network bandwidth given transmission volume; 2) current and anticipated bit error rate of the network; and 3) actual and anticipated queuing delays at end and intermediary nodes. A dialogue with the application layer occurs when optimal application performance requirements cannot be achieved, with the aim of reconfiguring application traffic, where possible, to meet QoS requirements (Fig. 3). Reconfigurations include compression and switching sound or colour off. Transmission aborts are possible from this state if sufficient reconfigurations cannot be made.

The aim of Phase 3 evaluation is to further populate the contextual attributes and determine required transport protocol mechanisms (Fig. 3). Mechanisms include error-checking, retransmissions, and acknowledgements. Mechanism choice will be based on its ability to help or hinder the application in achieving QoS, and the operational performance of the network dictating a need for the mechanism.

Protocol choice is the final stage of the evaluation process. A protocol is selected based on its ability to match the required mechanisms as determined in Phase 3, and its ability to achieve application QoS (Fig.4 and Fig. 5). For example, if the point of protocol failure is beyond the one-way propagation delay of the network, the protocol will be unsuitable. Several transport protocols are available for consideration, which include TCP [8], UDP [9], LTP [10] and TP-Planet [11]. The protocols have been chosen due to their believed capabilities in a number of operating environments.

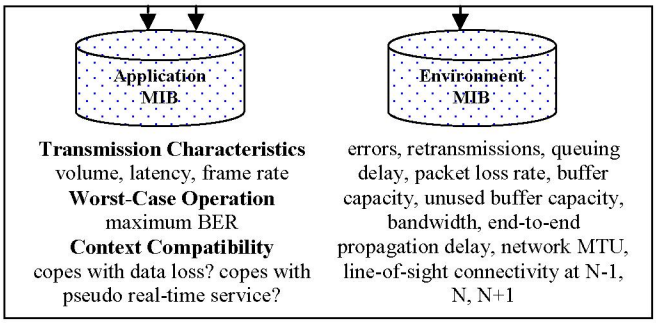

Figure 2. MIB Data used by CAB

Before transmission, protocol-specific re-configuration may occur. This stage represents the refinement of default protocol configurations to more accurately match the operating environment and to maximise the chances that application QoS will be achieved. The operational performance of TCP, for example, can be extended by adapting its timeout mechanisms, as identified in [12]. This stage recognises that, while one protocol may be more suitable in a particular communication, it is possible to further improve functionality.

Once transmission begins, the context-aware function does not end. Autonomic decisions continue in response to alarms which identify when network performance falls below an acceptable level. Subsequent action will be taken to maintain network performance at the necessary level. For example, an alarm will identify that the bit error rate is higher than that accepted by the application. In response, the broker will take action. It may fail the transmission if retransmissions cannot be made within the maximum application latency. Or, if it detects that a meteorite shower is causing a burst of errors, the transmission will be temporarily halted until the shower is over, providing the application can cope with this unexpected delay.

Where a disparity grows between application QoS and network performance, and the protocol fails to be able to bridge this gap, the point of protocol failure has been reached. At this stage, CAB will re-enter the evaluation stage and either continue transmission with a new protocol or abort.

IV. AUTONOMIC FEATURES

A system must possess four properties to be truly autonomic [3]: properties include self-configuring, self-optimising, self-healing, and self-protecting. CAB allows autonomy in the transport layer of the protocol stack (and can be extended to the other layers). Its autonomic aspects are indicated in Fig. 1. According to the definition, CAB is not completely autonomic. It self-configures by choosing between transport protocols to match the environmental constraints. Learning allows self-optimisation, by using past responses to a set of MIB attributes when the same scenario re-occurs. It self-protects using an emergency mode of operation and alarms to indicate when performance drops below an acceptable level. Self-healing, however, is not a feature of our autonomic system. If the node is damaged, CAB does not repair it, although it can inform ground-stations of the occurrence. If MIB information is absent, CAB will attempt to cope by transmitting on a best-effort basis, but will not request additional information. It is not appropriate to include human intelligence as part of the self-healing function of CAB, as in [2]. In doing so, an application cannot be truly autonomic.

V. DEMONSTRATION OF CAB OPERATION

To demonstrate CAB operation, consider application and environmental information in the following scenario.

```
Transmission volume: 100,000 bytes
Propagation delay: 0.06 seconds (or 17,987
kilometres)
Maximum application latency: 0.2 seconds
Network bandwidth: 512,000 bytes
Can cope with data loss: yes
```

Once the application and environmental information has been collected, the broker passes into Phase 1 of the evaluation process. The broker evaluates if, under ideal circumstances, it

is possible for the application to transmit within its maximum latency given the one-way propagation delay. With a total delay of 0.255 seconds (propagation and node de-queuing delay) between source and destination nodes, the maximum acceptable application latency of 0.2 seconds is exceeded. In Phase 2, the broker attempts to resolve the reasons for application QoS failing. In the example above, compression will be applied iteratively until transmission within the maximum application latency is possible or further compression is not. Transmission within 0.2 seconds is possible after 50% compression is applied. In Phase 3, required protocol mechanisms are determined. As there is little surplus time between time taken to transmit and the maximum application latency, retransmissions are turned off. This is acceptable because the actual network bit error rate is less than the bit error rate accepted by the application. If network errors were high and transmission accuracy important, 80% compression could be applied and retransmissions turned on. A hold state is not required due to the strict latency requirements of the application. Further decisions on the necessity of link probing, store and forward operation, error checking, retransmissions, and an unreliable mode of transport, enable the most suitable transport protocol to be chosen.

Performance of the broker can be demonstrated by its ability to choose between TCP and UDP. A series of evaluations assess protocol suitability, given the QoS requirements of the application and operational capabilities of the protocols (Fig. 4). Application QoS is defined using the attributes Type of Service (ToS), real-time requirements, and ability to cope with data loss, among others. ToS allows distinction between applications: a ToS between three and six represents audio and video applications. Those with a lower ToS may have less stringent latency requirements (e.g. email), but higher reliability requirements (e.g. telemetry).

Initially, a protocol is chosen which provides QoS, regardless of the environment. With a choice of TCP and UDP, TCP is chosen when the application demands reliability and accuracy. Protocol selection will be re-evaluated in relation to environmental information (Fig. 5). TCP's performance radius is 22.5 seconds [12] when the standard is implemented. Where delays are long and effects of the protocol visible, application analysis will be re-visited, with the aim of determining the ability of the application to cope with delays and data loss. Priorities between acceptable performance ranges allow protocol selection. Subsequently, protocol capabilities reflect application performance requirements. The protocol stack is thus empowered with the ability to make intelligent transmission decisions.

A. Experimental Results Driving Phase 3 Evaluation

Phase 3 evaluations become important as the number of protocols under consideration increase. In addition to key protocol mechanisms, performance over increasing distance has been quantified to aid protocol choice. Simulation of TCP has revealed relationships between key attributes and transmission performance (delay and goodput) (Fig. 6 and Fig. 7). The shape of the curves has been explained in [12], where an investigation of TCP's performance radii has been performed. These plots further confirm the effect of TCP on transmission delay as distance increases between nodes; this effect is independent of other factors which influence the speed at which a transmission takes place. Nonetheless, the effect of these factors can be seen, and should be taken into account. Attributes include file size (Fig. 6), link rate (Fig. 7), buffer size, and slow start threshold. Relationships between the attributes have been defined using multiple regression analysis to allow estimation of the total transmission time and

```
Evaluation: Phase 1
if (one_way_prop_delay > app_max_latency) {
        printf("One-way propagation delay is greater
than application latency - this cannot be overcome\n
ABORTING TRANSMISSION...\n");
        abort();}

Evaluation: Phase 2
compression_value = 0.8;
do {
        compressed_volume = (transmission_volume
        * compression_value);
        if (compressed_volume >
        recommended_transmission_volume){
        printf("Compressed volume greater than
        recommended volume … CONTINUE\n");}
        if (compression_value == 0.2) {
        printf("  It is not possible to further
        reduce transmission volume\n");
        break;}
        compression_value = compression_value -
        0.3;
} while (compressed_volume >
recommended_transmission_volume);

Evaluation: Phase 3
time_to_retransmit = (round_trip_prop_delay +
((cumulative_retransmissions) / network_bandwidth));
for (int p = 0; p < number_of_nodes_; p++) {
        cumulative_queuing_delay +=
        node_queuing_delay_[p];
}
total_time_to_retransmit = time_to_retransmit +
cumulative_queuing_delay
if (total_time_to_retransmit > app_max_latency){
        retransmissions_on = false;
} else {
        retransmissions_on = true;}
```

Figure 3. Pseudo-Code from Broker

```
Evaluation: Phase 4
if (type_of_service <= 3) {
  tcp_suitable = true;
  } else {
  tcp_suitable = false;}
if (real_time == true) {
  udp_suitable = true; tcp_suitable = false;}
if (can_cope_with_data_loss == false) {
  tcp_suitable = true; udp_suitable = false;
} else {
  udp_suitable = true; tcp_suitable = true;}
```

Figure 4. Protocol Assessment – Application Analysis

```
Evaluation: Phase 4
if (tcp_suitable == true) {
  if (distance > 22.5) {
  tcp_suitable = false;
  } else {
    if (distance < 1.5) {
    time_to_transmit = (-32.1 + 57.8 distance      +
0.000141 file_size);
      if (time_to_transmit > app_max_latency_){
      tcp_suitable = false;
      } else {
      tcp_suitable = true;}}}
```

Figure 5. Protocol Assessment – Environment Analysis

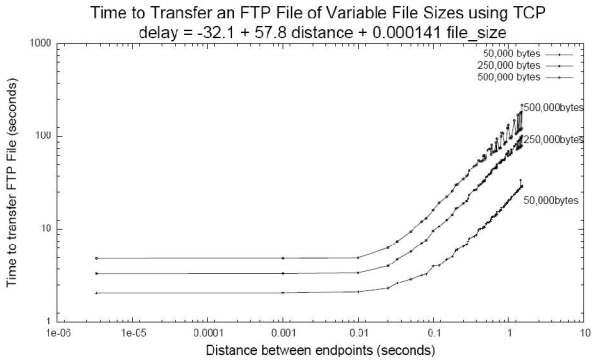

Figure 6. Influence of File Size on Delay

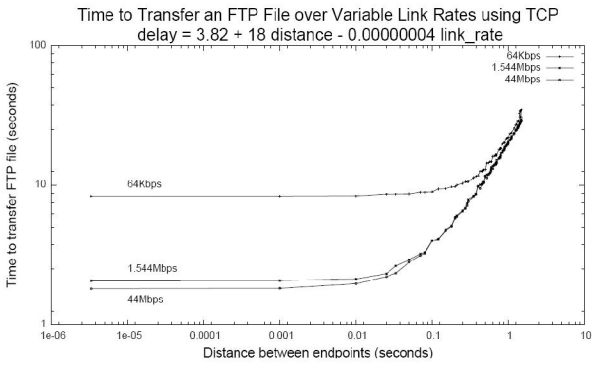

Figure 7. Influence of Link Rate on Delay

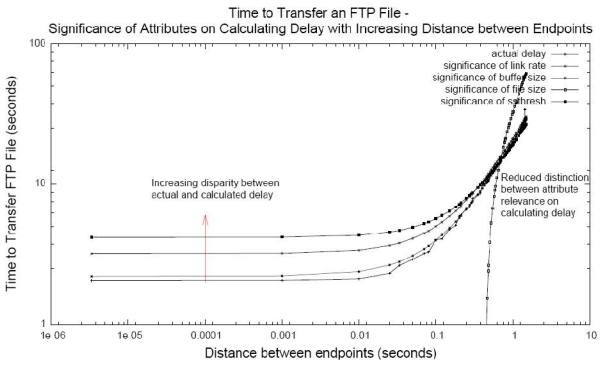

Figure 8. Significance of Attributes on Calculating Delay

the ability to decide if the protocol allows QoS. Performance is considered up to a distance of 1.5 seconds only. Beyond this, goodput drops off in relation to throughput [12]. CAB uses this information in advance of the communication to maximise success.

The quality of results produced when using the relationships has been measured. Fig. 8 identifies the accuracy of calculated delay by comparing disparity from actual delay. In order of decreasing importance, the attributes buffer size, link rate, ssthresh, and file size have a declining relevance on the calculation of delay when nodes are closest to each other. Buffer size can therefore be used effectively in advance to determine transmission delay. With distance, distinction between attribute relevance declines. File size has little relevance on delay at any distance: its effect is shown through performance of other attributes. This information is important to CAB, empowering it with the ability to use the most relevant attribute for the propagation distance.

VI. FURTHER WORK

Figs. 4 and 5 represent a high-level view of the evaluations. With incorporation of more protocols, decision-making becomes increasingly complex. Future work involves defining radii of protocols noted in Section III, characterising performance with distance between nodes, and defining relationships between attributes to extend decision-making.

Furthermore, we aim to manipulate the protocols to extend performance over greater distances. This involves, for, example improving TCP's performance by extending timeouts to a distance which reflects actual delay. CAB will therefore be imbued with intelligence to allow re-configuration for this, and other protocols. Once the protocol selection functionality has been fully incorporated, alarms will be introduced to identify drops in network performance. By empowering CAB with the ability to act when alarmed, its self-configuring and self–protecting capability is ensured.

VII. CONCLUSIONS

Protocols provide levels of service suitable for different environments when transmitting different applications. As one of the aims of next generation computing is autonomy of decisions and, given the diversity of transport protocols available, we consider it imperative to include a decision-making functionality with regard to protocol choice. The context-aware evaluation broker aims to allow application QoS to be achieved in all networking environments, without the need for human intervention. Autonomy of decisions has already enabled a new type of mission and we anticipate that our application of autonomy will not only achieve delay-tolerance in challenging links, but allow a uniform approach to communicating in the future.

REFERENCES

[1] R. Beck, K. Hinkel, C. Peoples, G. Parr, et al, "GPSDTN: Predictive Velocity-Enabled Delay-Tolerant Networks for Arctic Research and Sustainability", accepted for presentation at the IEEE First International Workshop on Tracking Computing Technologies, July 2007.

[2] S. Blake, D. Black, et al,"An Architecture for Differentiated Services", Network Working Group, RFC2475, December 1998.

[3] W, Truszkowski, J; Rash, et al; "Some Autonomic Properties of Two Legacy Multi-Agent Systems – Logos and ACT", Proceedings of the 11th IEEE Int. Conference and Workshop on Engineering of Autonomic Systems, 24 – 27 May 2004, pp 490 – 498.

[4] W, F, Truszkowski, M, G, Hinchey, et al; "Autonomous and Autonomic Systems: A Paradigm for Future Space Exploration Missions", IEEE Trans. on Systems, Man, and Cybernetics, Vol. 36, No. 3, May 2006, pp 279 – 291.

[5] K. Scott, S. Burleigh, "Bundle Protocol Specification", IETF Internet Draft, December 2006.

[6] V, Cerf, S; Burleigh, et al; "Delay-Tolerant Networking Architecture", Network Working Group, RFC4838, April 2007.

[7] C. Peoples, G. Parr, et al, "Improving the Performance of Asynchronous Communication in Long-Distance Delay-Sensitive Networks through the Development of Context-Aware Architectures," Proc. of IEEE Int. Conf. on Autonomic and Autonomous Systems, July 2006, pp. 49 – 57.

[8] J, Postel, "Transmission Control Protocol", RFC 793, September 1981.

[9] J, Postel, "User Datagram Protocol", RFC 768, August 1980.

[10] M. Ramadas et al., "Licklider Transmission Protocol – Specification," IETF internet draft, September 2006.

[11] Ö, B, Akan, J. Fang, I. F. Akyildiz, "TP-Planet: A Reliable Transport Protocol for InterPlaNetary Internet", IEEE Journal on Selected Areas in Communications, Vol. 22, No. 2, February 2004, pp. 348 – 361.

[12] L. Wood, C. Peoples, G. Parr et al, "TCP's protocol radius: the distance where timers prevent communication", submitted to the Third International Workshop on Satellite and Space Communications, September 2007.