Context-Aware Policy-Based Framework for Self-Management in Delay-Tolerant Networks: A Case Study for Deep Space Exploration

Cathryn Peoples, Gerard Parr, Bryan Scotney, and Adrian Moore, University of Ulster

ABSTRACT

Policy-based management allows the deployment of networks offering quality services in environments beyond the reach of real-time human control. A policy-based protocol stack middleware, the context-aware broker, has been developed by the authors to autonomically manage the remote deep space network. In this article example policy rules demonstrate the concept, and prototype results from ns-2.30 show the overall positive cost-benefit impact in an example scenario.

INTRODUCTION

Self-managing techniques, driven by context-aware capability, benefit next-generation network (NGN) management by enabling real- and pseudo-real-time responses to changes in the operational environment, therefore reducing operator costs and network downtime, and optimizing performance, application quality of service (QoS), and user quality of experience (QoE). Transmission between Earth and Mars, for example, can take up to 22 minutes when the path is traversed at the speed of light without interruption (Fig. 1). If network elements can autonomically manage themselves, local and intelligent decisions can be made in the presence of exception conditions, instead of having to communicate from the remote node back to Earth for any task or protocol update. Subsequently, problems can be resolved in less time and science return maximized. Contact with the National Aeronautics and Space Administration (NASA) Mars Global Surveyor (MGS), for example, was lost in November 2006. At the time, the reason for the failure was unknown, but was later discovered to be due to battery failure caused by an intensive sequence of energy-draining events. Real-time policy-based self-management of the spacecraft using contextual information could have prevented this by monitoring the rate at which battery energy was declining, enabling proactive action to be taken in situ before contact with it was lost.

In addition to managing individual spacecraft, there are opportunities for policy-based management of communications on the end-to-end link. In the past, each deep space communication was planned in advance to maximize opportunities that application QoS was achieved by ensuring line-of-sight (LOS) connectivity and available network resources. However, advances to this approach are occurring with the rollout of the delay-tolerant network (DTN) architecture described in [1]. The NASA Mars Phoenix spacecraft landed on Mars in May 2008 and used the Mars Reconnaissance Orbiter (MRO) and the European Space Agency (ESA) Mars Express Orbiter to return signals to Earth using the Consultative Committee for Space Data Systems (CCSDS) data link layer protocol, Proximity-1 (Fig. 1). Such dependency between missions has previously not been seen, due to incompatibility between their designs. As the deep space network (DSN) continues to expand, policy-based techniques can increasingly be used to autonomically manage the network. The deployment of more network resources will improve the chance that resources will be available when required and that connectivity issues can be accommodated. It is with this objective that the context-aware broker (CAB) has been designed. The CAB describes a network management middleware, deployed alongside protocol stacks used in DTNs, which influences decisions in the transport layer with regard to how and when transmission occurs. Its algorithm comprises an event-condition-action (ECA) framework, which uses contextual information to achieve its objective of improving transmission reliability.

The remainder of this article is structured as follows. Policy-based network management (PBNM) is discussed in greater detail in the next section, in terms of current deployment examples and the advantages achieved, investigating specifically the ability to optimize QoS in DTNs. In the following section, context awareness (CA) and policy-based capabilities as key enablers of autonomy in deep space are discussed, with sev-

0163-6804/10/$25.00 © 2010 IEEE

eral systems evaluated as examples. We then describe the CAB in terms of key algorithm phases, and policies that influence its design are then explored. Experiments and results from a deployment of the CAB in ns-2.30 [2] are then presented, and the article concludes in the final section.

Policy-Based Network Management

Policy-based network management describes the integration of event-condition-action rules into the network to enable self-management and achieve application QoS. In the *business* context, this includes monitoring network resource usage to ensure capacity availability at critical times and subsequent revenue maximization. For individual users, policy rules can benefit next-generation *personal* distributed environments by managing competing network devices with real-time and/or interactive QoS requirements. PBNM can also be applied to *scientific research*: a PBNM system can predict changes in the monitored object or environment to maximize science discovery proactively. In each instance described, the requirement for human input to respond to the network can result in the business or mission objective failing to be achieved through increased latency, and reduced responsiveness and reliability. It may result in insufficient bandwidth to support a customer phone call and subsequent loss of that customer. It may result in unacceptable jitter in an Internet Protocol television (IPTV) transmission, making it unwatchable. Or it may result in missing the opportunity to prepare data collectors prior to a volcanic explosion.

One of the main advantages of PBNM is the reduced reliance on real-time human control (although we are not suggesting that human control will be removed completely — policies can be conflicting and must be corrected by those who created them). This has the benefit of minimizing operational overhead costs and improving responsiveness to network events, features that are particularly beneficial in areas beyond the reach of real-time human control. These include environments with long propagation distances, for example, deep space, the North and South Poles, and deep sea. In each, real-time human control of the network is difficult due to remoteness and the subsequent round-trip latency involved when identifying and resolving issues. Furthermore, the probability of requiring a network management function is heightened due to harsh environmental conditions and limited network resources. In deep space, network management is required due to interruption to line-of-sight communication periods: during the ESA Venus Express mission, for example, there are blackout periods of up to two weeks. A management function therefore controls the time at which communication occurs, requiring understanding of planetary movements, calculation of the time at which connection between end nodes will be restored, and transmitting during this interval. A streamlined effort uses PBNM to calculate and initiate communications spontaneously.

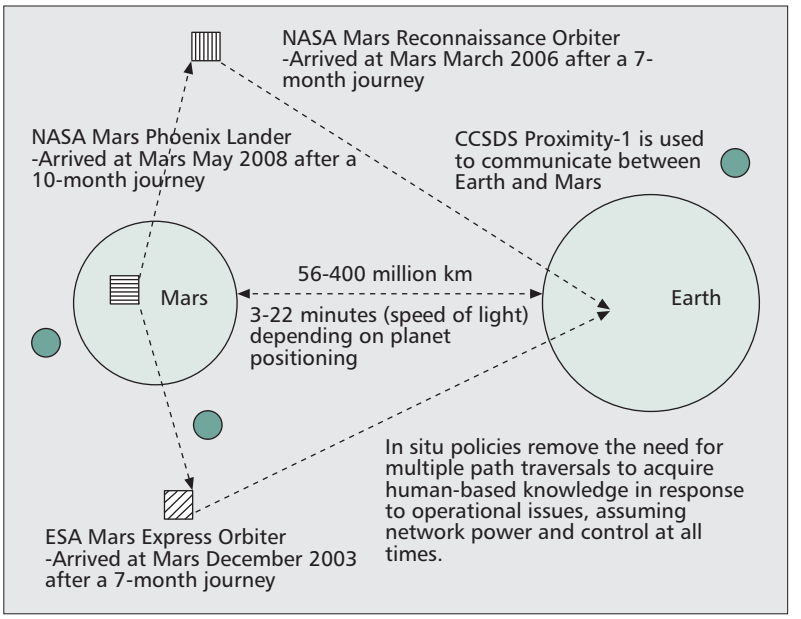

Figure 1. *DTN transmission scenarios between Earth and Mars.*

Context Awareness and Policy-Based Capabilities as Autonomy Enablers

Context awareness and policy-based capabilities drive autonomy, as validated by Hadjiantonis *et al.* (2006) in [3]. Autonomy, in parallel with biological systems, describes involuntary operation occurring both continuously and dynamically in response to events. In contrast to biological systems, engineered PBNM systems must be explicitly designed to collect information upon which decisions are made. This is described as context awareness, and engineered systems collect information autonomically and, ideally, without real-time human initiation of the data gathering process. Once collected, policy rules can be applied and autonomic control through reaction or prediction achieved.

Autonomic systems are under development for use in DSNs to advance science discovery in terms of the way in which a mission is executed and the type of data that may subsequently be collected. Two main types of autonomic systems have been developed for deep space exploration, including swarms and onboard spacecraft processing, as shown in Table 1. Truszkowski *et al.* (2006) describe a revolutionary approach to explore the asteroid belt in [4], involving swarms of multiple network components that communicate wirelessly between themselves and downlink scientific data to Earth. Each element is used to form a mesh network around an asteroid and identify key characteristics, including size and velocity.

The Autonomous Sciencecraft Experiment (ASE) [5], in contrast to swarms, demonstrates the ability to process science data onboard, discard unwanted data, downlink relevant datasets, and base future actions on sensed data. The ASE was deployed on Earth Orbiter-1 (EO-1) to monitor volcanic activity, but has also been used for flood detection and monitoring. This experiment demonstrates the ability to minimize ground station costs by guiding spacecraft autonomically:

While the benefits of PBNM in deep space have been recognized, it has not been applied to manage the DSN as a single entity, and communication between Earth and space continues to be defined and controlled specifically for each case.

Autonomous mission	Developer	Autonomic and CA technology
Autonomous Nano-Technology Swarm (ANTS) [4]	NASA	Swarms of wirelessly networked components
ASE [5]	NASA	Onboard processing influencing operations and science return to Earth

Table 1. *Autonomic deep space missions.*

before the ASE was deployed, EO-1 commands were uplinked several days in advance.

Context-aware techniques also exist in current DTN developments, documented but not yet deployed. The Bundle Agent Discovery mechanism, for example, identifies which nodes can communicate with each other. When an autodiscovery bundle arrives at a node, flags set indicate its properties: Bit 0, for example, is set when the node is willing and able to take custody of bundles. When the autodiscovery bundle returns to the source, the transmission is subsequently manipulated according to discovered information. As another example, the Licklider Transmission Protocol's (LTP) link state cue mechanism also demonstrates use of context awareness to autonomically guide operation. Link state cues distribute information to communicating nodes regarding link status and informs on the operational state of other nodes. These examples of Bundle Agent Discovery and LTP, proposed from research groups involved in deep space communication development, therefore validate the intention to deploy increasing amounts of context awareness in future space missions. The Bundle Agent Discovery Mechanism, however, was documented as an experimental Internet draft which expired in 2007 and has not since been updated, demonstrating the fact that autonomic context awareness with regard to deep space systems exists at an embryonic stage. While the benefits of PBNM in deep space have been recognized, it has not been applied to manage the DSN as a single entity, and communication between Earth and space continues to be defined and controlled specifically for each case. A research gap in terms of the application of PBNM and management of the Interplanetary DSN therefore exists.

A Context-Aware PBNM System

An autonomic network management function, the CAB [6], has been developed by the authors to support communication in DTNs (Fig. 2) and is not part of the state of the art. The purpose of the CAB is to intelligently configure a transmission before initiation, monitor transmission performance in long-distance networks, and take action when QoS benchmarks are not met. The Broker operates autonomically, collecting contextual information from the application and environment, and manages the transmission using a series of event-condition-action policies. Based on scientific findings from [7], for example, the Broker could eliminate the Transmission Control Protocol (TCP) from its list of acceptable protocols during the transmission configuration phase once the propagation distance between nodes exceeds an acceptable 1.5 s when measured based on the time to traverse the link at the speed of light. In this case it could substitute the LTP or Deep-Space Transport Protocol as being more suitable. A policy phase describes a set of rules and actions related to a module within the algorithm and may enforce selection of the most appropriate transport protocol for the transmission scenario, for example, or validation of application data accuracy. Each policy phase is composed of policy decision and policy enforcement points (as defined in RFC 3198): a policy decision point (PDP) represents the phase at which the decision-making process takes place. For example, *Phase 2 Evaluation* (Fig. 2) represents a PDP. A policy enforcement point (PEP) describes a phase at which policy enforcement is attempted. *Phase 2 Evaluation*, for example, may also be a PEP in addition to being a PDP.

QoS with regard to data networks is a measure of user satisfaction resulting from a network transfer. Satisfaction may result from the degree of timeliness, accuracy, and/or reliability of the service. These, therefore, are properties that the CAB takes into account in its decision-making process. This is in spite of the fact that timeliness, for example, has a different meaning when applied to long-distance deep space transfers in comparison to terrestrial services: when moving at the speed of light, a communication between Earth and Mars can take 22 minutes when all other network operating conditions are ideal. While real-time interactive transmissions will therefore not occur over such distances, in certain cases it may be required that the transfer can be achieved within a predefined latency, therefore requiring a timeliness QoS to be achieved in this example scenario to fulfill user satisfaction. It is for this reason that we consider it appropriate to refer to the QoS of a long-distance transfer in the same manner as for terrestrial services.

In the following discussion it also becomes apparent that the CAB includes considerations for real-time and interactive applications. The evaluation of applications with real-time QoS requirements may be questioned in a system developed for use in the long-distance deep space environment. It is included, however, due to the design assumption that the CAB is deployed on all nodes in a network communication. It will be deployed, for example, on a spacecraft. As the spacecraft travels further from Earth, it will require capability to transmit on the long-distance link. Once at the destination, however, it may land on a planet and release rover components to travel across the surface on which it has landed. If these rovers are controlled from the spacecraft, it will also therefore

require capability to communicate on the short-distance link and for the different range of applications that may be transmitted there.

A description of the CAB process follows. The CAB algorithm begins with the data collection (DC) of contextual information regarding the application and environment. It is important to identify key characteristics of the application to understand QoS requirements, including mission criticality and real-time and/or interactive features. A selection of contextual attributes is shown in Table 2. This information is validated (V) as part of DC to ensure accuracy and maximize the suitability of later decisions made. If an application request has been received, the CAB enters the *evaluation* process. Otherwise, it will progress to the *slee*p state to await further instruction. *Phase 1 evaluation* ($Phase1_{ex}$) processes collected information. If $Phase1_{ex}$ checks are passed, additional contextual facts will be inferred to maximize the amount of information upon which decisions can be made. Examples of inferred attributes relate to both application and environment aspects, and include the attributes acceptable application QoS (high/medium/low), one-way propagation delay (seconds), estimated delay to propagate a link (seconds), and estimated time of loss of line-of-sight connectivity. The estimated delay to propagate a link, for example, can be inferred from application and environment attributes, including transmission volume (bytes), packet size (bytes), available bandwidth (bytes per second), number of intermediary nodes, and the average queuing delay at each (seconds). Within *phase 2 evaluation* ($Phase2_{ex}$), the contextual information is evaluated in greater detail using the collected and inferred range. If $Phase2_{ex}$ checks are passed, CAB execution progresses to *Phase 3 Evaluation* ($Phase3_{ex}$). Here, key requirements of a transport protocol are determined, given application QoS requirements and characteristics of the operational environment. This will help in the process of selecting the most appropriate protocol in the next stage, *Protocol Choice* (P_{choice}). Once the protocol is selected and configured ($P_{configure}$), application traffic will be sent.

Context Broker functionality does not end once application traffic has been sent. *Monitoring* capabilities are also integrated into the middleware to allow the achieved level of QoS to be logged. If the Broker identifies a gap between the achieved and required QoS, it will take action. The overall objective of the Broker is to maximize transmission sustainability, but it will, in the worst case scenario, suspend transmission and progress to a *sleep* state when QoS is no longer reached and resource consumption is wasteful in the capacity-constrained environment. An example scenario demonstrates Broker functionality and the overall positive cost-benefit impact on performance achieved in a later section.

CAB Policies

There are several policy rules fueling the CAB decision-making process, a selection of which is presented in the following sections. Attributes used in CAB rules are summarized in Table 2 and demonstrate examples of contextual information driving the algorithm. Rules are expressed as a series of event-condition-action probabilities to gain an appreciation of the likelihood of each event occurring, with performance trends being used by the CAB during M*monitoring* procedures.

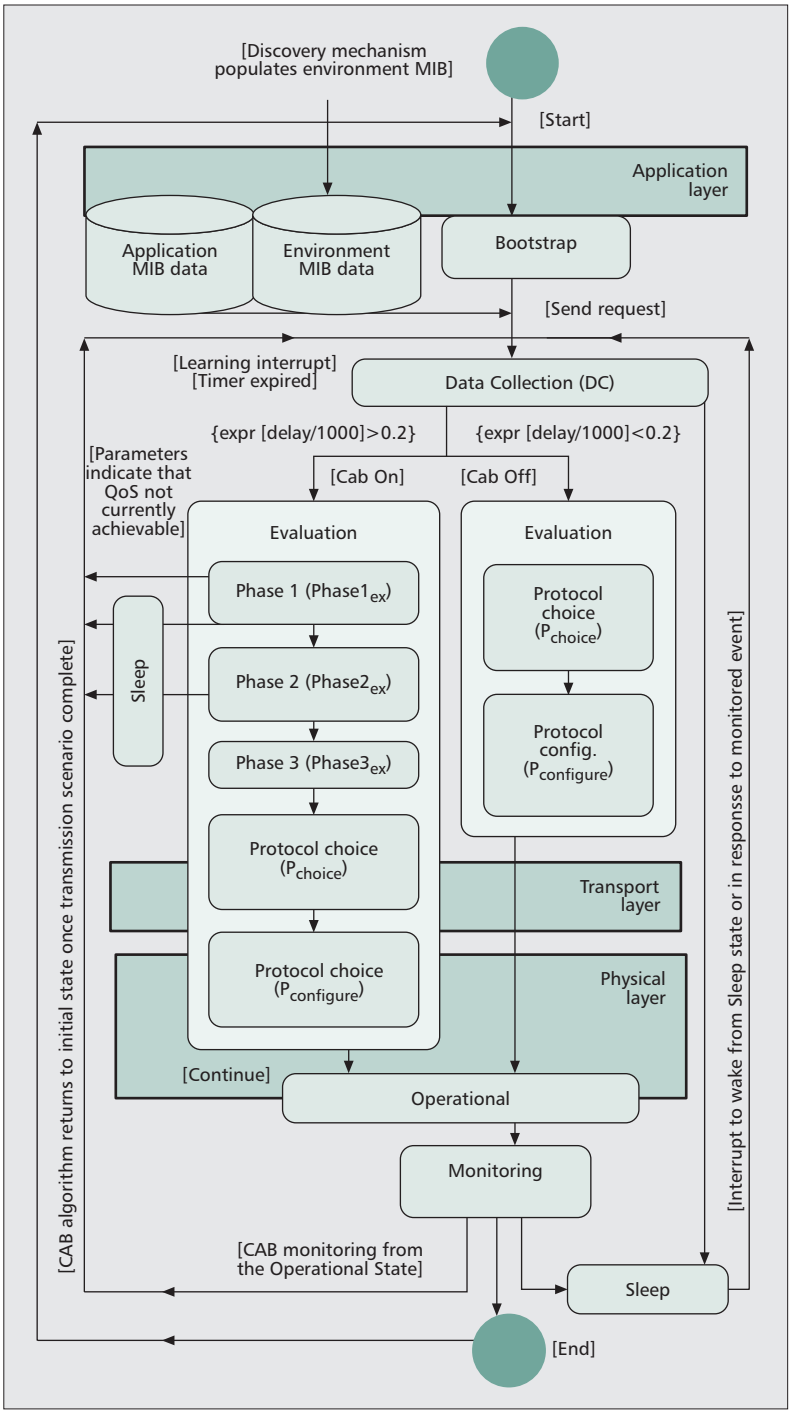

Figure 2. *Context broker state transition diagram.*

Pretransmission Policies

An example of a policy used in the initiating stages of the Broker algorithm follows.

- The probability of execution abort from *Phase 1 Evaluation* is dependent on the probability that LOS connectivity in a wireless network does not exist at the time when transmission is initiated, and the time when line-of-sight will be recovered causes an

The CAB policies presented are relatively straightforward, but demonstrate the use of policy rules and the range of attributes. The complexity of the decision-making process, however, should not be underestimated: in any policy rule, there are further relevant attributes in addition to those shown.

unacceptable waiting latency for the application (Eq. 1):

$$\Pr(E_{abort} \,||\, E_{continue\_be}) \sim \Pr((LOS_{t0} = false) \&\& ((t1 - t0) > L_{acceptable})) \quad (1)$$

where

$$t1 = \min_{t > t0} \left| LOS_t = true. \right.$$

In this scenario the CAB attempts to maintain QoS by avoiding transmission when it identifies in advance that application QoS will not be achieved. In the instance that the application is flexible and mission-critical, however, the Broker will instruct that a best effort (BE) execution continues. This will involve suspension until connectivity is restored and controlling transmission until complete. In this circumstance the Broker attempts the complete range of options to complete transmission and adopts the BE approach when necessary.

Alarm Policies

Policies have been developed to allow the CAB to react to changes in the environment once transmission has begun. These are implemented as alarms (Fig. 2) within the *Monitoring* state, and their invocation indicates that performance is compromising the ability to achieve QoS. Action is therefore invoked to maintain performance at the required level. An example of an alarm policy follows.

- The alarm in Eq. 2 identifies that queuing delay at a node plus the estimated time to propagate the link causes application real-time synchronous transmission requirements to be compromised (Eq. 2):

$$\Pr(T_{continue\_be} \,||\, T_{abort} \,||\, T_{suspend}) \sim \Pr(((Q + L) > \text{app}^{\max}_{latency}) \&\& (I = true)). \quad (2)$$

Depending on application QoS requirements, the transmission can be aborted, suspended, or continued on a BE basis. It will be suspended in the instance that an interruption will allow node queues to clear within an acceptable latency, enabling the application to continue and achieve its required QoS. When the application is mission-critical, the Broker will either transmit on a BE basis or suspend, and will avoid aborting.

Predictive Policies

Policies are incorporated into the Broker that enable predictive functionality in response to trends identified during transmission. An example of a predictive policy follows.

The Broker can predict that a node's battery power will fail during the transmission based on information involving node battery capacities collected using a discovery process and the estimated remaining amount of load to be offered (Eq. 3):

$$\Pr(Nd_{failure}) \sim \Pr(BP < Ld). \quad (3)$$

In this example one unit represents the battery power required to throughput one packet per unit of time. If the estimated node throughput exceeds the remaining amount of power available at the node, it will be unable to support the transmission. In this instance the Broker can take proactive action to avoid the scenario of the transmission failing for this reason, by suspending it until the node recharges, and using the discovery mechanism to inform the network of this decision.

The CAB policies presented are relatively straightforward, but demonstrate the use of policy rules and the range of attributes. The complexity of the decision-making process, however, should not be underestimated: in any policy rule, there are further relevant attributes in addition to those shown. In the case of Eq. 2, for example, the Broker also needs to know the application's mission criticality, node battery capacity, and the relationship between the network bit error rate and acceptable application bit error rate when making the suspend/abort/continue_be decision. This information can be further supplemented, for example, using details and future predictions on trends in the observed available bandwidth. Taking these attributes and their relationships into account, the CAB has been developed to alleviate the negative effects of the network on the achievable QoS. The objective of the policies is to optimize the use of opportunities for achieving application QoS within the dynamic environment.

Experiments and Results

As discussed in the previous section, the CAB uses predictive techniques regarding node battery power and the ability of a node to support transmission based on the amount of power remaining (Eq. 3). When the battery monitoring interval is reached, the Broker processes contextual information collected on the power level of network nodes. It first analyzes if the node has battery power and determines if this power is sufficient for the remainder of the transmission. If the node either does not have power or has insufficient power, the Broker assesses the application's ability to wait until the node recharges. Otherwise, the CAB will assess the ability of the remaining nodes to continue communicating if the failed node is not replaced. If the propagation distance between nodes is acceptable given the operational radius of the signal, the CAB will allow transmission to continue. Otherwise, it will assess if additional nodes have moved into range since analysis began. If they have not, the Broker will return to the *sleep* state. The Context Broker will try all possible options in its attempt to sustain transmission in the dynamic environment. In the worst case scenario, however, returning to a *sleep* state may be its only option.

A deployment of the CAB algorithm has been implemented in ns-2.30. The impact of Broker functionality when node capacity declines, as described above, has been tested using a mobile network with 20 wireless nodes, routing between which uses the Ad Hoc On-Demand Distance Vector (AODV) protocol. The test scenario involves a short-distance mobile node network, which allows demonstration of CAB functionality with regard to mobility issues, including diminishing node battery capacity. The impact of extending this network to a long-distance scenario would be to increase communication latency by a function of the network

Symbol	Data type	Analysis/action attribute	Attribute description
A_{ach}	int	Analysis	Accuracy achievable (%)
A_{req}	int	Analysis	Accuracy required (%)
app_{time}	real	Analysis	Time spent performing transmission (seconds)
$app^{max}_{latency}$	real	Analysis	Acceptable maximum application latency (seconds)
BER	real	Analysis	Bit error rate (errors/second)
$BER_{acceptable}$	real	Analysis	Acceptable bit error rate (errors/second)
BP	int	Analysis	Node battery power (units) (where 1 unit = power required to throughput one packet per unit of time)
BW	real	Analysis	Bandwidth (bits per second)
DR_{req}	real	Analysis	Required data rate (bits per second)
I	boolean	Analysis	Interactive application (true/false)
L	real	Analysis	Propagation latency between nodes (seconds)
Ld	real	Analysis	Application load (bytes)
$L_{acceptable}$	real	Analysis	Acceptable application transmission latency (seconds)
LOS_t	boolean	Analysis	Line-of-sight connectivity at time t (true/false)
mc	boolean	Analysis	Mission-critical application (true/false)
N_{ofr}	boolean	Analysis	Node out-of-range (true/false)
$Nd_{failure}$	boolean	Analysis	Node failure (true/false)
Q	real	Analysis	Queuing delay at a node (seconds)
SS_1	real	Analysis	Signal strength (previous) (dB)
SS_2	real	Analysis	Signal strength (current) (dB)
E_{abort}	boolean	Action	Abort from CAB execution event
$E_{continue\_be}$	boolean	Action	CAB execution continues on a best-effort basis event
$P_{handover}$		Action	Protocol handover event
$PC_{reliable}$	boolean	Action	Reliable transport protocol selection (true/false)
T_{abort}	boolean	Action	Transmission abort event
$T_{continue\_be}$	boolean	Action	Transmission continues on a best-effort basis event
$T_{suspend}$	boolean	Action	Transmission suspend event
||	—	Action selection	Represents *or*, in terms of the selection of one action or a different action.
~	—	—	Represents *is dependent on*, in terms of the relationship between left and right hand sides of the equation.

Table 2. *Attributes used by CAB policies.*

bandwidth and data rate (assuming all other network conditions are optimal), the duration of which can easily be calculated. In this scenario observing the CAB's reaction in a mobile, and not long-distance, scenario is the objective.

The test scenario is built on an assumption that all nodes require the same recharge time, the network bit error rate remains constant during the transmission, and other nodes do not move into range before transmission completes. Results in Figs. 3 and 4 demonstrate the impact of the experimental framework on performance as the network scales in size, up to a maximum of 64 nodes (the number of nodes chosen due to assumptions on future DSN size). Reliability QoS is calculated based on the relationship between the volume of traffic offered by the sending node and the volume received at the des-

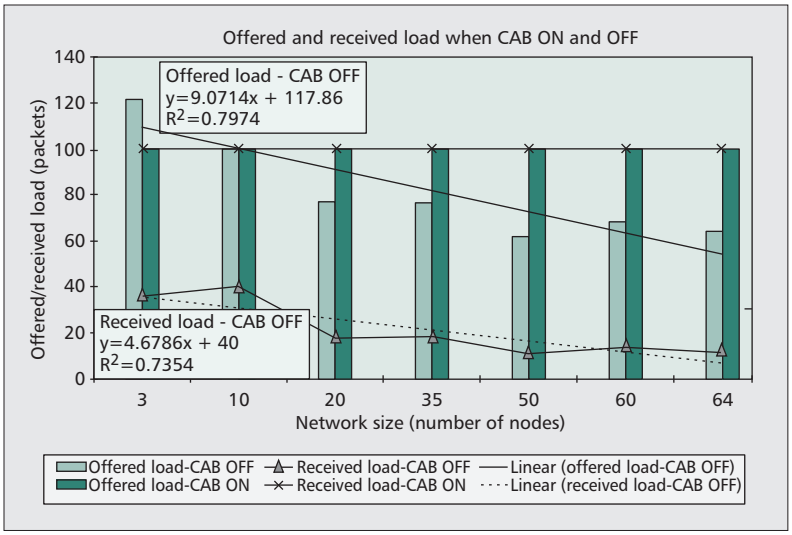

Figure 3. *Differences between offered and received load when CAB is on and off.*

tination When the CAB is turned on, reliability QoS is achieved in networks of all sizes because it predicts that node capacity is insufficient to support the transmission using information collected by the network discovery mechanism and suspends it until the node recharges in accordance with the QoS latency requirements of the application. The offered load therefore equals the received load as all packets sent are received (Fig. 3), and reliability QoS is achieved (Fig. 4). There are step increases in the latency incurred when enforcing reliability QoS, which occur in parallel with increases in the network size. This is because it takes a longer time for the reliable protocol to propagate packets through the network when there are more nodes involved.

When the Context Broker is turned off, node battery power fails 10 s into the transmission and is not recovered. Offered load is higher when the CAB is off (Fig. 3) because packets sent are not received, and the reliable protocol enforces retransmissions. When the network is smaller, there are more packets sent during the transmission period as it takes less time to propagate the network with fewer intermediary nodes between source and destination. Reliability is below the required 100 percent benchmark in all networks and declines as the size increases (Fig. 4). The decline is because offered load decreases as network size increases, and the subsequent received load also decreases. Offered load declines because it takes more time for packets to propagate an increasing number of nodes, and there are therefore fewer packets sent from the source during the period spent attempting to transmit.

In response to transmission reliability, additional latency incurred when the CAB is turned on is acceptable, given the fact that it enables reliability QoS to be achieved, and this reliability is achieved within a period that is acceptable to the application. Reliability QoS is not achieved when the CAB is turned off; therefore, the latency incurred when attempting the transmission, and the offered and received load passed between nodes represent wasted transmission time and unnecessary resource consumption. This fact is particularly important when the network resource is finite, as is the case for node battery capacity.

Conclusion and Further Work

The CAB middleware uses policy-based decision making at its core to merge application transmission requirements with capabilities of the operational environment. This technique, which is not yet state of the art in DTNs, presents a self-managing approach in networks beyond real-time human control, and, in particular, managing the Interplanetary DTN as a single network.

Results in the previous section from a deployment of the CAB in ns-2.30 validate the impact of the policy-driven model on a transmission, its ability to achieve autonomy and self-management, and improvement of the relationship between offered and received load in the test scenario, with an average increase in reliability QoS by 75 percent for the seven networks tested. When the CAB was not applied to the transmission, reliability QoS was not achieved in any scenarios due to the inability to preempt node battery failure. This ability of the CAB to improve transmission performance is due to the flexibility of the application with regard to its QoS requirements or, specifically, the latency within which it must be transmitted. For applications with real-time requirements, it is unacceptable to suspend transmission mid-flow; therefore, such a reaction would not be taken for an application with latency requirements that are different than in this scenario. A range of results demonstrating positive operation of the CAB in different scenarios could have been presented, including the cost-benefit impact of transmission suspension during $Phase2_{ex}$ due to propagation delay on the end-to-end link (L) and acceptable application transmission latency ($L_{acceptable}$), or its autonomic protocol selection and configuration ability. The power-aware scenario is, however, considered important in representing positive CAB performance on its own, given the fact that deployment of wireless nodes in deep space is costly in terms of time, human, and financial resources, and being able to optimize capacity use during their lifetime is a priority.

The ability to roll out completely autonomic communication in deep space will be challenging, given the dynamics of the environment and the requirement for all eventualities to be fully identified and incorporated as policies for a system to be fully effective. While a complete system may initially be unrealistic, each functionality deployed will advance techniques further, allowing inefficiencies to be identified and resolutions supplied. It is therefore most likely that autonomic policy-based self-management in deep space will be rolled out using the current stepping stone approach used for other aspects of the network to date.

Further work within this research involves expanding the test scenarios to identify for which transmissions the CAB benefits performance most and for which scenarios its inclusion leads to unnecessary overhead without improvements. Findings to these results will then be integrated into the CAB decision-making process to implement further control over the intelligent decisions it makes.

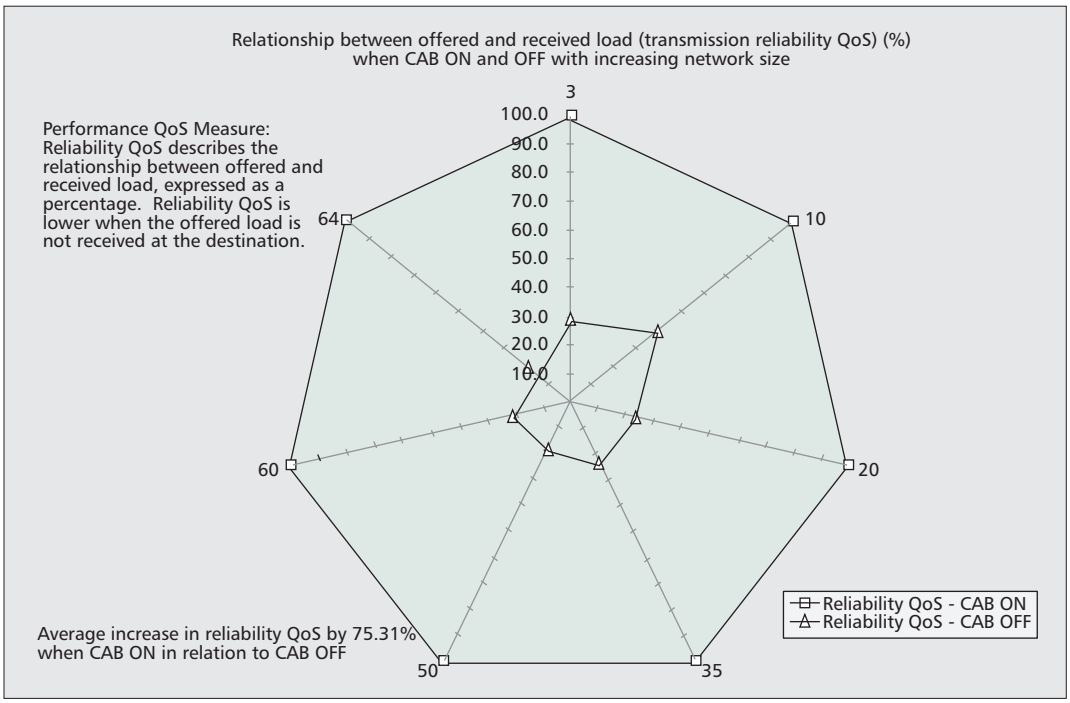

Figure 4. *Relationship between reliability QoS when CAB ON and OFF.*

Further work within this research involves expanding the test scenarios to identify which transmissions the CAB benefits performance most and for which scenarios its inclusion leads to unnecessary overhead without improvements.

References

[1] V. Cerf *et al.*, "Delay-Tolerant Networking Architecture," IETF RFC 4838, Apr. 2007.

[2] Network Simulator ns-2.30, accessed Mar. 30, 2010; http://nsnam.isi.edu/nsnam/index.php/Main_Page

[3] A. M. Hadjiantonis, A. Malatras, and G. Pavlou, "A Context-Aware Policy-Based Framework for the Management of MANETs," *Proc. 7th IEEE Int'l. Wksp. Policies Distrib. Sys. Net.*, June 2006, pp. 1–10.

[4] W. F. Truszkowski *et al.*, "Autonomous and Autonomic Systems: A Paradigm for Future Space Exploration Missions," *IEEE Trans. Systems, Man, Cybernetics — Part C: Apps. Reviews*, vol. 36, no. 3, May 2006, pp. 279–90.

[5] G. Rabideau *et al.*, "Mission Objectives of Earth Observing-1 with Onboard Autonomy," *Proc. 2nd IEEE Int'l. Conf. Space Mission Challenges Info. Tech.*, July 2006, pp. 1–10.

[6] C. Peoples *et al.*, "A Reconfigurable Context-Aware Protocol Stack for Interplanetary Communication," *Proc. 3rd IEEE Int'l. Wksp. Satellite Space Commun.*, Sept. 2007, pp. 281–85.

[7] L. Wood *et al.*, "TCP's Protocol Radius: The Distance where Timers Prevent Communication," *Proc. 3rd IEEE Int'l. Wksp. Satellite Space Commun.*, Sept. 2007, pp. 163–67.

Biographies

CATHRYN PEOPLES (c.peoples@ulster.ac.uk) received B.A. (Hons) (2004), M.Sc. (2005), and Ph.D. (2009) degrees from the University of Ulster, Northern Ireland, where she is currently working as a postdoctoral research assistant. During her Ph.D., improving communication sustainability and reliability in deep space was a research objective. Current research interests being pursued in the Faculty of Computing and Engineering include energy-efficient networking using cross-layer protocol stack optimization to reduce ICT carbon emissions and assist operation in rural and challenged environments.

GERARD PARR (gp.parr@ulster.ac.uk) holds the Full Chair in Telecommunications Engineering in the Faculty of Computing and Engineering at the University of Ulster, Coleraine. He holds a Ph.D. in self-stabilizing protocols, aspects of which he developed while a visiting scientist at the DARPA-funded Information Sciences institute in Los Angeles. Areas of research within the group include self-stabilizing protocols, interplanetary network protocols, real-time network management systems, CNNs and intelligent mobile agents in xDSL and SNMP, real-time data analytics for NMS, energy-aware infrastructure, resource management protocols, applications performance management in virtualized environments, bandwidth provision over SONET/SDH in the presence of chaotic impulses, and fuzzy inference systems for multicriteria handoff in tactical communications. He is the U.K. principal investigator of the EPSRC-DST India-U.K. Centre of Excellence in Next Generation Networks of which BT is the lead industrial partner, and also a principal investigator in the EPSRC funded Project Sensing, Unmanned, Autonomous Aerial Vehicles (SUAAVE). For further information see http://www.suaave.org.

BRYAN SCOTNEY (bw.scotney@ulster.ac.uk) received a B.Sc. in mathematics from the University of Durham, United Kingdom, in 1980, and a Ph.D. in mathematics from the University of Reading, United Kingdom, in 1985. He is a professor of informatics and currently director of the Computer Science Research Institute at the University of Ulster. He has over 150 publications in total, spanning statistical databases, image processing, and mathematical computation. He has published widely in the area of knowledge discovery and data mining, focusing on solving the problems of integration of heterogeneous data from distributed sources, particularly where the information available is imprecise and/or uncertain. Much of this work has been supported by funding from four EU FP5 projects (with McClean) to address issues of harmonization of official statistics. Most recently, he is co-investigator on EPSRC award EP/F030118/1. His early research interest in computational mathematics is now focused on the area of digital image processing; this work is concerned mainly with methodological development, but also has application to machine vision and medical imaging. He is currently President of the Irish Pattern Recognition and Classification Society.

ADRIAN MOORE (aa.moore@ulster.ac.uk) has been a member of academic staff at the University of Ulster since 1990, taking up the post immediately after completing a D.Phil. in computer graphics at the university. At Ulster he is the school coordinator for Internet and e-learning activities, and represents the university on the Coleraine Borough Council's broadband working group. He has taught in a wide number of subject areas in computer science, especially Web applications development, multimedia, and computer graphics, and has completed a textbook, *Multimedia Web Development* (Palgrave Macmillan). He has written and delivered a number of professional development courses for local industry, including Web development and C programming. He has also undertaken a number of external consultancy projects and is the technical director of Consultus International Limited, a Northern Ireland e-consultancy company.