Multi-Protocol Transport Layer QoS: A Performance Analysis for The Smart Grid

James Wilcox, Dritan Kaleshi
Center for Communications Research
University of Bristol, UK
James.wilcox@bristol.ac.uk
Dritan.kaleshi@bristol.ac.uk

Mahesh Sooriyabandara
Telecommunication Research Laboratory
Toshiba Research Europe Limited, UK
Mahesh@toshiba-trel.com

***Abstract*— Application specific interaction models will be required to support efficient communications between distributed applications with disparate network requirements in the consumer side Smart Grid (SG) data network. While much work has been done to quantify SG communications requirements in general, there is little public information on how to support the individual applications. This paper will show that specific transport protocols are able to provide increased efficiency of network resource usage under specific network conditions and that by providing a real-time adaptive selection of these transport protocols it would be possible to achieve a distributed embedded system with heterogeneous actors that can react to both application-specified Quality of Service (QoS) requirements and varying network conditions.**

Keywords – Quality of Service, Adaptive Transport Layer, Network Emulation, Middleware, Smart Grid

I. INTRODUCTION

Traditionally networked applications are designed with a pre-selected transport layer protocol. Optimisations for a specific application are done at the application layer and all messages are transported using the same protocol, either TCP (Transmission Control Protocol) [1], UDP (User Datagram Protocol), or with an overlay transport protocol such as RTP (Real Time Protocol) [2]; Fig. 1 represents this paradigm. This is typically fixed at application development; however there is no fundamental requirement for this to be the general rule. Whilst networked applications need to exchange information, there is no reason why application layer code should be concerned with how that information is transported. There are a multitude of existing, mature transport layer protocols available each designed to tackle specific network problems [3]. Utilising these many protocols, a single application could leverage the advantages of each protocol individually at the appropriate time given an environment with dynamic network conditions and application requirements. Acknowledging these points raises the challenge of defining a generic framework that allows for run-time selection of transport protocols to dynamically match specific application requirements and, specifically, message patterns used by the application. If the low level network interactions enforced by a specific transport protocol and higher level architectural messaging pattern are completely decoupled from the application then dynamically modifying the combination can be used as standard. If certain transport protocols and messaging pattern combinations are able to provide higher performance in terms of bandwidth, latency and reliability in certain network environments than others can do, then by supporting adaptive selection of these combinations it becomes possible to have a distributed real-time embedded (DRE) system with heterogeneous actors that can react to both dynamic application QoS requirements and network conditions. This model is shown in Fig. 2. The middleware system required for managing the selection of the large numbers of transport protocols referenced in Fig. 2 is referenced for completeness but is outside the scope of this paper.

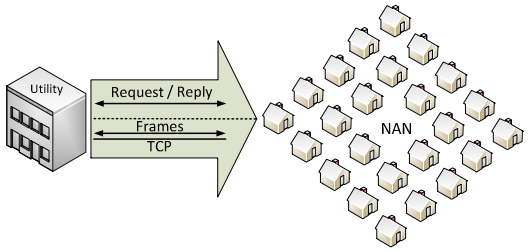

Figure 1. Traditional: Applications are supported by a single transport (in this case, TCP Request / Reply). The utility represents systems providing the SG infrastructure.

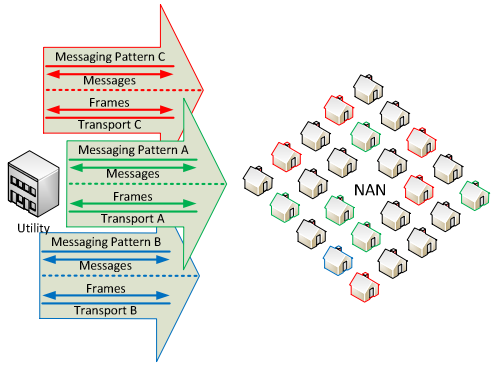

Figure 2. Proposed: Applications using multiple run time optimised transports (managed using middleware) instead of only TCP Request / Reply.

This paper shows experimental results which demonstrate that specific combinations of transport protocol and messaging patterns will provide performance gains over a pre-defined communication transport architecture and that certain combinations provide useful gains over other potentially viable options. The potential of generating a large scale mapping of transport combinations and application requirements will be explored.

The main contributions of this paper are:

- An analysis of performance of a distributed system for different transport protocol and messaging pattern combinations by running application scenarios using a fixed, realistic, network topology.
- Results that show that the performance of the Distributed Real-time Embedded (DRE) applications varies significantly for different messaging pattern and

 ISBN: 978-1-61208-332-2

transport protocol combinations, suggesting that this then can be used to optimise the performance of the individual transactions that make up the application network traffic.

The paper is organized into the following sections: section 2 provides the related material and further motivation for this work. Section 3 presents the experimental emulation test bed setup and the viable communication interaction models. Section 4 presents the experimental parameters and the results. Section 5 presents the conclusions from the experiments and the direction of future work.

II. RELATED WORK AND MOTIVATION

Environments targeted by this work have the following characteristics:

1. They are distributed and built from a large number of heterogeneous embedded devices, running a number of different applications.
2. They are typically loosely-coupled.
3. The majority of the actors are communication network-constrained rather than computing resource-constrained.
4. Each device is expected to run many different applications with varying network requirements.

One example of such a system is the SG, and in particular, the subset of applications that intend to use consumer / demand side equipment and systems to achieve grid specific goals such as load shedding or load shifting or in more general terms, Demand Side Management (DSM).

Section A introduces the SG and its edge applications. Several related works exist which support with the premise of providing DRE software applications, such as those that will operate in a SG, with flexible communication choices in order to either achieve better network resource allocations or meet specific communication requirements. These overviews are presented in sections B and C.

A. The Smart Grid and Demand Side Management (DSM)

The SG can be seen as a large scale distributed system, with a large component being embedded sensor-actuator networks to support distribution power network monitoring and control and DSM interactions. DSM focuses around the control of demand side loads in the electricity distribution network in order to manipulate network conditions [4]. DSM breaks down into a number of related but still significantly different enough sub-applications to warrant different communications approaches. DSM can be broken down into two major sub categories, Demand Control (DC) [5] and Demand Response (DR) [6]. DC is defined as DSM programs that have centralized direct control over consumer loads [5]; DR is defined as DSM programs that use indirect methods (typically pricing) to affect changes [7]. Each approach requires a different communications paradigm in order to utilise network resources efficiently and operate optimally.

The above presents an ideal system for this work. It presents the rare opportunity to take a completely different approach to facilitating machine-to-machine communications in a DRE environment. The SG will eventually call for millions of networked geographically-distributed embedded devices to be deployed into the demand side of the power distribution grid. These devices are either designed to utilise existing networks such as domestic broadband or cellular networks and coexist with the existing traffic, or to utilise purpose built resource constrained networks such as various forms of wireless mesh or power line communications [8]. Both approaches result in strict network resource constraints for the applications. These constraints further increase the impact run-time transport level adaptive QoS will have in such environments. The main argument against dynamically matching transport protocols and messaging patterns to application requirements and real time network conditions has been one of complexity. With sensor networks, the SG and the Internet of Things (IoT) in general becoming more prevalent the environment is changing and these arguments are no longer valid. Our proposition is that the performance gain introduced by dynamically matching transport protocols and messaging patterns outweighs the required increase in complexity of the architecture and embedded hardware.

DSM can be shown to be a good example of how tailored communication paradigms could be beneficial in the SG and similar environments.

B. Transport Mechanisms

The concept of adaptive transport layer services for resource-constraint environments is well explored; however, the approach taken usually considers the transport protocol to be used already pre-selected at development stage, and consider adaptations above the transport layer. They do not propose to provide application optimisations at the lower transport level. However applications, regardless of the type of resource-constrained environment, can benefit from tailored communication service at the transport layer. Mutlu et al. [9] presents a middleware solution for performing transport level QoS focused on Bluetooth application profiles and uses CORBA (Common Object Request Broker Architecture) [10] to facilitate the middleware. While the scope is clearly limited, and transport protocol choices are not part of the QoS mechanism, the motivation is similar. Furthermore, it can be shown that different communication protocols have inherently different QoS characteristics and that using targeted protocols with specific applications can improve performance with a number of chosen metrics. Weishan et al. [11] recognise this and provide experimental results related to protocol switching overhead and also implement the system using a middleware solution. They conclude that protocol switching overhead is minimal with their chosen transport protocols and that protocol switching is beneficial to DRE environments.

C. QoS Architectures for DRE systems

The works highlighted here are attempting to improve or maintain DRE application performance in sub-optimal or resource-constrained networks by utilising real-time adaptive QoS management mechanisms. [12-14] focus on a single messaging pattern and attempt to provide adaptive QoS within these confines. It demonstrates that additional QoS optimisation opportunities are available if the scope of the system includes

 ISBN: 978-1-61208-332-2

controlling lower level attributes such as transport protocols and messaging pattern combinations in conjunction with the adaptive QoS mechanisms. For example Wenjie et al. [13] propose a QoS adaptive framework for Publish-Subscribe Service called QoS Adaptive Publish-Subscribe (QAPS). They define several QoS policies and focus on fault tolerance and dependability of services. Schantz et al. [15] present a distributed, real-time embedded system capable of adaptive QoS. They describe in detail several methods of implementing end-to-end adaptive QoS mechanisms and explain how the work gives DRE applications more precise control over how their end-to-end resource allocations are managed. These proposed adaptive QoS mechanisms all address the same problem as this paper, but these implementations are limited to the application layer instead of considering a multi-protocol transport layer to access additional optimisation opportunities. Zieba et al. [14] develop the concept of quality-constrained routing in publish / subscribe messaging architectures. They develop a system which integrates application quality requirements into the message routing architecture in order to better support dealing with varying network conditions such as dynamic network topologies and link characteristics. The idea of integrating the dynamic application requirements into the communication paradigm provides a critical distinction from the others and further reinforces the need for verified optimised communication paradigms in order to meet these dynamic requirements.

III. EXPERIMENTAL SCENARIOS AND EMULATION TEST BED

A. Experimental Scenarios

The topology used is shown in Fig. 3. It is a simple fan out type topology where one node is distributing data to a group of 300 nodes representing consumer smart meters.

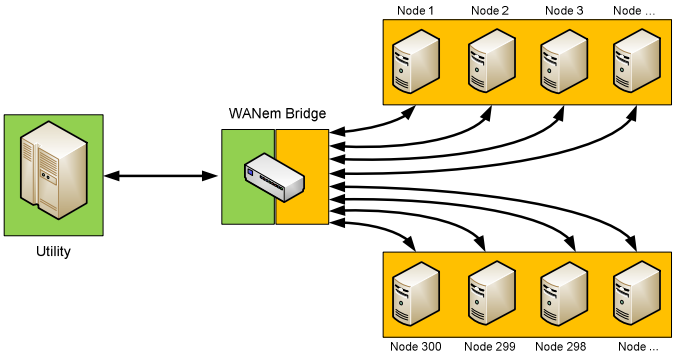

Figure 3 – Test bed network topology. 300 nodes are connected to a utility system through a software Ethernet bridge. The Utility publishes the update.

The topology represents the logical grouping that could be used in a Real Time Pricing DSM operation [16]. Traffic shaping is provided by WANem [17], which is a software wide-area network emulator. It provides the ability to manipulate many common network characteristics including available bandwidth, latency and packet loss.

B. Selected Communication Paradigms

Four viable transport protocol and messaging patterns were chosen to experiment with; these are shown in Table I and II. All are tested with both ideal and resource constrained, lossy network conditions in a set of eight experiments.

TABLE I - DOWNLINK (UTILITY→CONSUMERS) TRANSPORT & MESSAGING PATTERN CHOICES.

Scenario	Transport Protocol	Messaging Pattern
1	TCP	Router / Dealer
2	TCP	Publish / Subscribe
3	PGM [18]	Publish / Subscribe
4	UDP	Request / Response

TABLE II - UPLINK (CONSUMERS→UTILITY) TRANSPORT & MESSAGING PATTERN CHOICES

Scenario	Transport Protocol	Messaging Pattern
1	TCP	Request / Response
2	TCP	Request / Response
3	TCP	Request / Response
4	UDP	Request / Response

Router / Dealer is a tightly-coupled request-response style messaging pattern belonging to the ZeroMQ [19] socket API. It allows messages prefixed with a globally unique identifier (GUID) to be routed to a socket, remote or local, which has that same GUID. Each message sent needs to be prefixed with a valid GUID of a node which requires additional initialisation steps in order to acquire this information. Publish / Subscribe is a loosely coupled data distribution style messaging pattern. A publisher publishes a message prefixed with a topic / channel identifier. Only subscribers which have confirmed their interest in messages belonging to this topic / channel get the message routed to them. Scenarios 2 and 3 both use publish / subscribe but they use different transport protocols. Scenario 2 uses standard TCP. TCP is a unicast transport which implies that if a pricing update is to be sent to 300 nodes the Utility node will have to generate and send 300 individually addressed packets (assuming no fragmentation). Conversely, Pragmatic General Multicast [18] (PGM) is an experimental IETF (Internet Engineering Task Force) transport protocol designed to provide reliable multicast communications. In this case, the utility generates only a single packet (again assuming no fragmentation). Whereas TCP and PGM are both reliable transport protocols, UDP is unreliable. It does not have any mechanisms for ensuring reliable delivery but this does mean that it exhibits a lower network overhead.

C. Link configurations of selected network scenarios

Table III shows the network condition scenarios used in conjunction with the scenarios shown in Table I and II.

TABLE III - THE PHYSICAL NETWORK RESTRICTIONS.

Network Condition	Description
Ideal	No restrictions on bandwidth (10Gbps nominal – effectively unlimited) or any additional latency or packet loss
Resource Constrained	Bandwidth limited to 250Kb/s, additional 30ms +/- 5ms latency and 30% packet loss.

The resource constrained experiment emulates specific network conditions and represents a hypothetical resource-constrained lossy network on the link from the consumers to the utility such as an IEEE 802.15.4 based solution. Even though this represents two opposite extreme scenarios the results would still support the conclusions made for other network conditions.

 ISBN: 978-1-61208-332-2

Further experimental details are:

- In all experiments, the application layer maximum transmission unit (MTU) was configured for each transport protocol to ensure the packet size on the wire did not exceed 127 bytes. This was done to emulate the larger transport overhead (due to fragmentation) that would be seen when using these transport protocols with data link layers that can only support small packet sizes.
- The virtualised Ethernet bridge interface cards were configured for half duplex communication in order to emulate a half-duplex radio link.
- The payload used was a 1699 byte Extensible Markup Language (XML) string which is compatible with the OpenADR EventState.xsd XML schema [20].
- A price update was issued every 0.15 seconds in the request / response architectures and every 45 (0.15*300 = 45) seconds for the publish / subscribe architectures. This approach produces comparable test results as the fundamental differences in how the data is distributed between request / response and publish / subscribe would otherwise make this difficult. All scenarios achieve the goal of generating the same total number of responses from the consumers.
- All experiments issued price updates for up to 90 seconds and generated 600 responses from the consumers. Tests were allowed to run until all inflight responses were obtained.

The frequency of the Real Time Pricing (RTP) update is higher than any real world application. However, as the number of packets being generated, and hence the congestion, vary linearly with the RTP update frequency, using this frequency simply allows results to be collected easier. The higher frequency has no effect on the conclusions that are made in these experiments.

D. Emulation Test Bed

In order to develop and evaluate the premise that controlling the transport protocol and messaging pattern combinations of an application is an effective approach to manipulating the QoS, a way of allowing the experimental network code to interact with large numbers nodes was required. Emulation was chosen over simulation. Emulation provides a middle ground between a real world trial and simulation. The cost and difficulty of deploying a real world scaled trial is avoided but the ability to produce accurate and detailed data is maintained. There are several advantages to using a fully emulated approach over a simulated or semi emulated one. Firstly the test system is not attempting to approximate another real word system using a simplified model as is the case with a simulated approach. The emulation test bed can be seen as a condensed version of a real life system with all the varying levels of complexity a real world system would have from the standard open source software running on each node down to the physical layer of the network. Network analysis through the use of emulation is not a new area [21-25]. There is a large amount of work that uses network emulation due to the benefits it provides over purely simulation based analysis. More specifically, even the idea of a SG data network has undergone emulation based analysis [24] in the SCORE project which is a SG version of the Common Open Research Emulator (CORE) [26]. However none of the emulators allow manipulation of the transport layer and none allow the customisation of the virtualised node hardware in order to emulate resource constrained devices at the same time as being able to run real application code. Therefore, given the nature of the work it was decided that an emulation based approach would give the necessary flexibility needed and therefore a custom test bed was developed. To implement this test-bed a custom ESXi (VMWare Inc.) bare-metal hypervisor deployment was used.

IV. EXPERIMENTAL RESULTS

The results in this section use the UDP scenarios as a baseline. The raw results are shown in Table IV. This is done due to the UDP scenarios representing the simplest combination being experimented with. By using this scenario as benchmark it is easier to see how the other combinations perform in the given network topology against a well understood, ubiquitous transport protocol.

TABLE IV – THE RAW UDP RESULTS THAT CAN BE USED FOR COMPARISON WHEN RESULTS ARE SHOWN AS PERCENTAGE INCREASES.

	Ideal	**Constrained**
Utility Data	1.424 MBytes	1.424 MBytes
Consumer Data	1.424 MBytes	1.202 MBytes
Overhead	586.080 KBytes	586.080 KBytes
Message Round Trip Delay	2.183 ms	46.814 ms
Message Loss	0 %	39.3 %

The message latency result in Table V show that the publish / subscribe messaging patterns (Exp. 5 and 6) introduce greater latency than the request / response messaging patterns (Exp. 3 and 7).

TABLE V – MESSAGE LATENCY ROUND TRIP DELAY (RTD) AND MESSAGE LOSS (PS: PUBLISH / SUBSCRIBE, RD: ROUTER / DEALER, RR: REQUEST / RESPONSE)

	Message Round Trip Delay (RTD) (ms)	**Message Loss (%)**
1. TCP PS Ideal	345.6	0.00
2. TCP PS Constrained	21500.0	0.33
3. TCP RD Ideal	5.4	0.00
4. TCP RD Constrained	604.3	0.00
5. PGM PS Ideal	363.7	0.00
6. PGM PS Constrained	17040.0	0.33
7. UDP RR Ideal	2.2	0.00
8. UDP RR Constrained	46.8	39.30

Under the lossy, congested network conditions it took on average 17.04 seconds to complete a round trip for the PGM experiment (Exp. 6) and 21.5 seconds for the TCP (Exp. 2). This is extremely high and is due to the way the consumers are responding; the PGM and TCP publish / subscribe consumers use the same TCP request response architecture to respond with. There is no rate limiting which is causing a large amount of congestion. PGM provides an interface to limit the multicast data rate which would be very useful in this case. It can also be seen that even under perfect network conditions, rate unlimited publish / subscribe architectures are not suitable for applications requiring low latency as individual message delays are over 150 times that of the UDP case (Exp. 1 and 5 vs. 7).

 ISBN: 978-1-61208-332-2

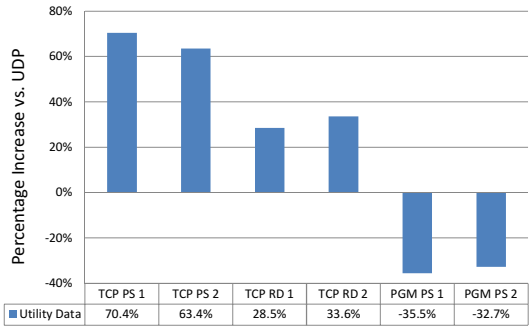

Figure 4 - Data sent by the Utility system compared to the UDP scenario.

The results indicate that the publish / subscribe architectures need a mechanism for rate limiting publishes and responses. The congestion generated when a published pricing update is sent and then responded to by all of the consumers simultaneously quickly overwhelms the resource constrained network generating message losses, which in turn cause retransmissions which contribute to the large amount of data generated. This can be seen in Figure 4 and Figure 5. The TCP publish / subscribe scenario shows this better than the PGM scenario. In this scenario the resource constrained, lossy network test actually performs better than the ideal case as the artificially imposed packet delay is having the effect of limiting the packet rate which even with the 30% packet loss and the retransmissions this would introduce, causes the scenario to generate less traffic than the 'ideal' case. This also indicates that a component in the virtual network is being stressed to the point of packet loss under the high packet rates being generated by the low MTU. Even with acknowledging this it shows that high messages rates with relatively large payload compared to the MTU will cause worst congestion problems than 30% packet loss does.

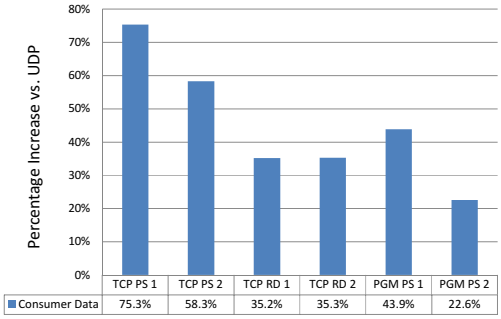

Figure 5 - Data sent by the consumer nodes compared to the UDP scenario.

The latency (Table IV, Exp. 7 and 8) and overhead (Figure 6) results show the UDP experiments outperform both the TCP or PGM equivalents with the TCP. This is not unexpected given the TCP and PGM are both reliable transports, with retransmissions that introduce increased delays against UDP. The notable observation is the performance gap between them. TCP is a generic transport capable of serving many different application requirements quite adequately, but the overhead involved in being so generic is clearly shown in these experiments. There is a clear opportunity to bridge this large gap with a number of UDP based messaging patterns, both unicast and multicast, and apply various application layer reliability mechanisms to them. This would allow applications access to a range of communications service combinations at a higher granularity, so that applications can get a communication service with only the features they need and avoid the general overhead of a one-transport-fits-all approach.

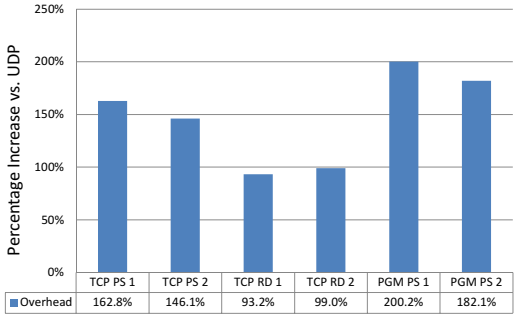

Figure 6 – Protocol overhead compared to the UDP scenario measured as any data that was not the XML payload. (Percentage increase is shown).

Figure 6 shows a large PGM percentage overhead. Given the much lower overall bandwidth consumed using multicast, this is to be expected. Overhead is calculated as any bytes put onto the wire that are not part of the XML payload. In order to generate the 600 responses (the experimental scenario criteria) from the consumers the utility only has to generate 2 PGM packets (ignoring fragmentation due to the low MTU). The overhead is almost entirely due to the TCP request response uplink from the consumer to the utility. Normalising these results against the total data exchanged shows the PGM architectures are in fact the most efficient next to the UDP architectures.

The results show that even though TCP publish / subscribe would appear to be a potential choice for this type of scenario (data distribution with a large fan-out) given that on face value it appears to provide the necessary interface for providing efficient data distribution, it actually performed the worst. TCP-based publish / subscribe involves a high amount of overhead to effectively allow a unicast architecture to emulate services that require a multicast architecture in order to operate efficiently. It provides no network orientated benefit over TCP Router / Dealer. The only benefit it provides is the ability to distribute messages at a more abstract level due to the use of topics / channels. In fact the lack of control on the distribution rate of the messages means that TCP router / dealer is more flexible and consistently generates less overhead and congestion as can be seen in Figures 4-6.

V. CONCLUSIONS AND FUTURE WORK

This paper has presented and validated an argument for exploiting the performance gains achievable by specifically selecting application appropriate transport protocols dynamically at runtime based on specific application requirements. Given the varied network requirements demanded by SG applications and DRE applications in general this approach provides previously inaccessible optimisation opportunities. Furthermore, these gains are achievable without the need to perform costly modifications to any intermediate network infrastructure and would only require modifications to existing networked applications' network interfacing code. The

 ISBN: 978-1-61208-332-2

cost of this modification could be mitigated by using a middleware system for managing the transport selection.

To summarise, the results have shown:

- For an ideal RTP update distribution use case PGM publish / subscribe and UDP request / response should be used on the down links and up links respectively for best performance and lowest resource utilisation.
- For the non-ideal case, the unreliable UDP is only viable if the application can suffer lost responses from the consumer – this is a possible scenario. If more reliability is required then another low overhead reliable transport should be used with TCP based options used as a last resort.
- There is a significant gap between the performance of the Reliable TCP / PGM scenarios and the unreliable UDP scenario in terms of overhead and latency. Additional transports are needed to fill the gap.
- TCP based Publish / Subscribe provides no network level benefits.
- Rate unlimited Publish / Subscribe is not viable for applications with a low latency requirement. The packet rate needs to be limited at the point of transmission in order to ensure congestion is not generated.

A large number of additional supported transport protocols, would make it possible for a system to generate custom network interfaces for a much wider range of scenarios in order to improve application performance through manipulation at the transport level. Future work will consider how to automatically manage the large number of potential transport protocol choices which are being suggested using middleware solutions.

VI. REFERENCES

[1] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll*, et al.*, "Packet-level traffic measurements from the Sprint IP backbone," *Network, IEEE,* vol. 17, pp. 6-16, 2003.

[2] IETF. (18/02/14). *RTP: A Transport Protocol for Real-Time Applications*. Available: http://www.ietf.org/rfc/rfc3550.txt

[3] IANA. (18/02/14). *Assigned Internet Protocol Numbers*. Available: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

[4] A. Mohsenian-Rad, V. W. S. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia, "Autonomous Demand-Side Management Based on Game-Theoretic Energy Consumption Scheduling for the Future Smart Grid," *IEEE Transactions on Smart Grid,* vol. 1, pp. 320-331, 2010.

[5] A. Gabaldon, A. Molina, C. Roldan, J. A. Fuentes, E. Gomez, I. J. Ramirez-Rosado*, et al.*, "Assessment and simulation of demand-side management potential in urban power distribution networks," in *Power Tech Conference Proceedings IEEE Bologna*, 2003, p. 5 pp. Vol.4.

[6] M. LeMay, R. Nelli, G. Gross, and C. A. Gunter, "An Integrated Architecture for Demand Response Communications and Control," in *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, 2008, pp. 174-174.

[7] S. Chua-Liang and D. Kirschen, "Quantifying the Effect of Demand Response on Electricity Markets," *Power Systems, IEEE Transactions on,* vol. 24, pp. 1199-1207, 2009.

[8] Z. M. Fadlullah, M. M. Fouda, N. Kato, A. Takeuchi, N. Iwasaki, and Y. Nozaki, "Toward intelligent machine-to-machine communications in smart grid," *Communications Magazine, IEEE,* vol. 49, pp. 60-65, 2011.

[9] U. Mutlu, R. Edwards, and P. Coulton, "QoS Aware Bluetooth Middleware," in *Information and Communication Technologies, 2006. ICTTA '06. 2nd*, 2006, pp. 3239-3244.

[10] M. Henning, "The Rise and Fall of CORBA," *Queue,* vol. 4, pp. 28-34, 2006.

[11] Z. Weishan, K. M. Hansen, J. Fernandes, Schu, x, J. tte*, et al.*, "QoS-Aware Self-adaptation of Communication Protocols in a Pervasive Service Middleware," in *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on & Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, 2010, pp. 17-26.

[12] K. Young-Jin, L. Jaehwan, G. Atkinson, K. Hongseok, and M. Thottan, "SeDAX: A Scalable, Resilient, and Secure Platform for Smart Grid Communications," *Selected Areas in Communications, IEEE Journal on,* vol. 30, pp. 1119-1136, 2012.

[13] Z. Wenjie, Y. Nanhua, C. Hui, H. Jiajian, L. Chuanjian, and L. Xin Jie, "Providing adaptive QoS for Real-Time Publish-Subscribe Service in SGIOC-HQ," in *Innovative Smart Grid Technologies - Asia (ISGT Asia), 2012 IEEE*, 2012, pp. 1-5.

[14] B. Zieba, M. v. Sinderen, and M. Wegdam, "Quality-constrained routing in publish/subscribe systems," presented at the 3rd International Workshop on Middleware for Pervasive and Ad-hoc Computing, MPAC 2005, Grenoble, France, 2005.

[15] R. E. Schantz, J. P. Loyall, C. Rodrigues, D. C. Schmidt, Y. Krishnamurthy, and I. Pyarali, "Flexible and adaptive QoS control for distributed real-time and embedded middleware," presented at the Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware, Rio de Janeiro, Brazil, 2003.

[16] Z. Wei and A. Feliachi, "Residential load control through real-time pricing signals," in *Proceedings of the 35th Southeastern Symposium on System Theory.* , 2003, pp. 269-272.

[17] H. K. Kalitay and M. K. Nambiar, "Designing WANem : A Wide Area Network emulator tool," in *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, 2011, pp. 1-4.

[18] IETF. (21/02/14). *RFC3208 - Pragmatic General Multicast Reliable Transport Protocol Specification*.

[19] ZeroMQ.org. (28/01/14). *Using DEALER and ROUTER Sockets*. Available: http://zeromq.org/tutorials:dealer-and-router

[20] OpenADR. (18/02/14). *OpenADR EventState.xsd XML schema*. Available: http://openadr.lbl.gov/src/EventState.xsd

[21] S. Doshi, U. Lee, R. Bagrodia, and D. McKeon, "Network Design and Implementation using Emulation-Based Analysis," in *Military Communications Conference, 2007. MILCOM 2007. IEEE*, 2007, pp. 1-8.

[22] K. Fall, "Network emulation in the VINT/NS simulator," in *Computers and Communications, 1999. Proceedings. IEEE International Symposium on*, 1999, pp. 244-250.

[23] S. Maier, A. Grau, H. Weinschrott, and K. Rothermel, "Scalable Network Emulation: A Comparison of Virtual Routing and Virtual Machines," in *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*, 2007, pp. 395-402.

[24] T. Song, S. Wen-Zhan, D. Qifen, and T. Lang, "SCORE: SmartGrid common open research emulator," in *Smart Grid Communications (SmartGridComm), 2012 IEEE Third International Conference on*, 2012, pp. 282-287.

[25] E. Weingärtner, F. Schmidt, H. V. Lehn, T. Heer, and K. Wehrle, "SliceTime: a platform for scalable and accurate network emulation," presented at the Proceedings of the 8th USENIX conference on Networked systems design and implementation, 2011, pp. 19-19.

[26] J. Ahrenholz, C. Danilov, T. R. Henderson, and J. H. Kim, "CORE: A real-time network emulator," in *Military Communications Conference, 2008. MILCOM 2008. IEEE*, 2008, pp. 1-7.

 ISBN: 978-1-61208-332-2