World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:10, 2007

Pervasive Differentiated Services: A QoS Model for Pervasive Systems

Sherif G. Aly

***Abstract*—**In this article, we introduce a mechanism by which the same concept of differentiated services used in network transmission can be applied to provide quality of service levels to pervasive systems applications. The classical DiffServ model, including marking and classification, assured forwarding, and expedited forwarding, are all utilized to create quality of service guarantees for various pervasive applications requiring different levels of quality of service. Through a collection of various sensors, personal devices, and data sources, the transmission of context-sensitive data can automatically occur within a pervasive system with a given quality of service level. Triggers, initiators, sources, and receivers are four entities labeled in our mechanism. An explanation of the role of each is provided, and how quality of service is guaranteed.

***Keywords*—**Pervasive Systems, Quality of Service, Differentiated Services, Mobile Devices.

I. Introduction

PERVASIVE systems were first envisioned by Mark Weiser at Xerox PARC as environments saturated with computing and communication capability, yet gracefully integrated with human users [1]. As mobile devices become an ever-increasing reality of every day life, there exists an ever-increasing demand for both infrastructure and applications that support the interweaving of mobile devices into the fabric of every day life. Through a combination of mobile devices, sensors, networking infrastructure, and application intelligence, devices are capable of becoming context-sensitive, aware of their surrounding environment, and hence interacting and taking appropriate action to support and facilitate their owner's presence in the surrounding context. Not only should pervasive systems be context-aware, but they should be capable of blending into the surrounding environment in a way that defies variations in configurations and standards across various environments.

The common most characteristic however amongst such devices is their limited resource capabilities, including storage, computation, and communication. Most such communication-capable devices use some form of wireless communication such as Bluetooth, 802.11, cellular digital packet data (CDPD), or infrared, and have relatively limited storage and processing abilities. The supporting infrastructure and applications of pervasive systems must take into consideration the limited resource capabilities of mobile devices. Numerous pervasive systems projects were actually implemented as mentioned in [11] [12] [13] and [14].

As practical examples of pervasive systems, a physician standing in front of a patient's bed can have the patient's medical history record automatically downloaded to their accompanying Personal Digital Assistant (PDA). An airplane mechanic can have the service records of an airplane automatically downloaded to their PDA upon approaching the plane. A waiter at a restaurant can receive warning if they attempt to deliver an order to the wrong table through a network of intelligence and proximity sensors. A manager with their PDA going into a meeting room to deliver a scheduled presentation can have their PDA trigger the presentation to be automatically migrated to the presenting machine, the light settings automatically configured, and the climate control system customized.

However, the situation is immensely different between a physician performing a routine check across patient's beds, and a physician responding to an emergency call at the intensive care unit. Both situations are similar in the sense that patient records should be downloaded onto the physician's PDA, however, significant preferential treatment should be given to the latter scenario to support the nature of the situation. Similarly, a manager presenting a high-profile presentation should have a guarantee of migration of their presentation in due time.

The need for quality of service guarantees supporting pervasive infrastructure and applications arises. Quality of service can be defined as offering service differentiation based on the requirements of users and applications. In this article, we present the utilization of an Internet Engineering Task Force (IETF) model idea, namely DiffServ, to be applicable for the pervasive systems domain.

II. Differentiated Services (DiffServ)

The differentiated services architecture [2] proposed by the Internet Engineering Task Force (IETF) is inarguably the most famous and sustainable solution for providing Quality of Service (QoS) over IP networks. The architecture is capable of providing different types or levels of services for network traffic. Network elements are customized to service multiple classes of traffic. IP flows are thus classified and aggregated

Sherif G. Aly is with the Department of Computer Science, The American University in Cairo, P.O. Box 2511, 113 Sharia Kasr El Aini, Cairo, Egypt (phone:+20-2-797-5313; fax:+20-2-795-7565; e-mail: sgamal@aucegypt.edu).

into different forwarding classes [4]. One such standardized forwarding mechanism is the assured forwarding per-hop behavior (AF PHBs) [3]. Under such mechanism, packets are monitored and marked according to a service level agreement. At the time of congestion, packets marked for a higher quality of service, namely in-profile packets, are provided preferential treatment using queuing mechanisms at the expense of other packets not fitting the higher quality of service profile, namely, out-profile packets [5]. Within the assured forwarding per-hop behavior, four classes of traffic are defined [9], namely AF1, AF2, AF3, and AF4 as shown in Table I.

TABLE I
DIFFSERV ASSURED FORWARDING CODEPOINT TABLE

DROP PRECEDENCE	CLASS #1	CLASS #2	CLASS #3	CLASS #4
LOW DROP	(AF11) 001010	(AF21) 010010	(AF31) 011010	(AF41) 100010
MEDIUM DROP	(AF12) 001100	(AF22) 010100	(AF32) 011100	(AF42) 100100
HIGH DROP	(AF13) 001110	(AF23) 010110	(AF33) 011110	(AF43) 100110

Each class is given a specific amount of buffer space and interface bandwidth at the routers such that a given service level agreement can be met. The allocation of buffer space and bandwidth can determine the preferential treatment given to traffic assigned to each one of the classes indicated above. Within each AF class, there are three drop precedence values that control how various packets will be dropped during times of congestion as given below in (1):

X is the AF class
Y is the drop precedence value within class x
DP is the drop priority

$$DP(AFX1) \leq DP(AFX2) \leq DP(AFX3) \quad (1)$$

The drop priority of AF11 as an example should be less than or equal to AF12, which in turn should be less than or equal to AF13. In the event of congestion, packets marked with AF13 will have a higher probability of being dropped than packets marked with AF11. Thus, through controlling the resources allocated to each class, and through marking packets with various drop precedence levels, applications may be given quality of service guarantees.

As opposed to Differentiated Services, Integrated services (IntServ) was also proposed by the IETF, and follows the signaled QoS model. End hosts signal their quality of service needs to the network, and hence the network creates microflows with reserved resources such as bandwidth, maximum packet size, and maximum burst size.

The significant problem with Integrated Services is the need to continuously track and update the microflows, that, which adds extra traffic overhead over the network [6]. Other quality of service models were also proposed for specific kinds of applications such as the Controlled Load Service Model for applications with "real-time" characteristics, and the Guaranteed Service Model to provide bounded delay and assured delivery of all packets falling within a given specification of applications [7].

In [8], the usage of policies in pervasive systems was investigated. In [10], a quality of service middleware support for pervasive computing applications is discussed. The authors describe middleware services that facilitate implementation of pervasive computing applications in dynamic and complex environments, including quality of service and resource management.

III. TRIGGERS, INITIATORS, SOURCES, AND RECIPIENTS

In the scenario examples indicated earlier, significant amounts of data transfer may occur amongst various devices. A classical differentiated services model will rely on network elements such as routers residing on the data transmission path to enforce differentiated services. A differentiated services domain will typically consist of ingress nodes, interior nodes (in the core), and egress nodes as shown in Fig. 1.

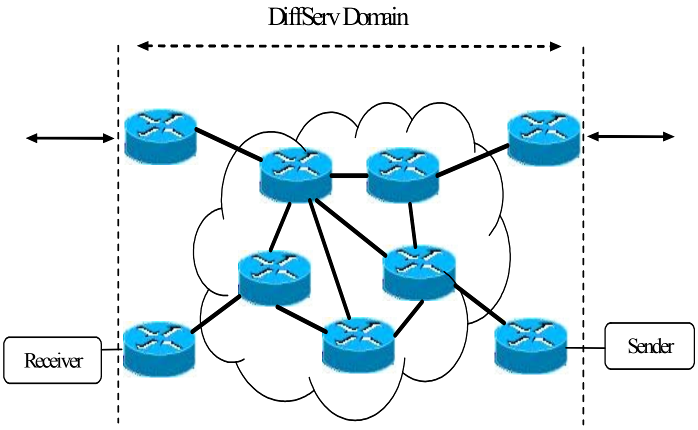

Fig. 1 A Classical DiffServ Domain

However, inside a single one-sited organization, significant amounts of data may be transferred through a limited number of hops over network elements, such as routers, or even as direct wireless communication between data stores and the personal device itself as shown in Fig. 2. A typical data transmission in a small organization may not at all involve the transmission over any routers properly configured for differentiated services provision. Rather, transmission in such close environments with close proximity of users will usually involve point to point communication, or rather, communication that involves an extremely limited number of transmission hops.

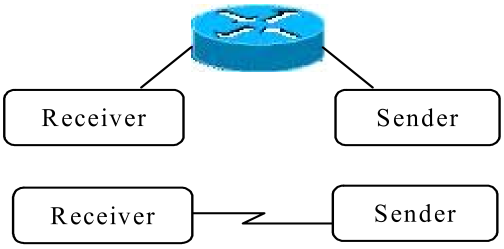

Fig. 2 Transmission Over Limited Hops

Due to the limited number of hops, or even their lack there of, in an environment setting similar to the ones mentioned earlier, it is not feasible to rely solely on a the classical differentiated services mechanism enforced at the various network elements. It is imperative to extend the model to support point to point communication between various elements in a pervasive environment.

We will define four types of entities in our mechanism, namely triggers, initiators, sources, and receivers. A trigger is an entity that triggers the initiation of a data transfer. A location sensor beside a patient's bed is a typical example of a trigger. Upon sensing the presence of a physician beside a patient's bed, the trigger will signal to an initiator that a data transfer needs to take place, along with the type of data to be transferred, and level of differentiated service. The level of required differentiated services is pre-determined at the trigger. This is the primary mechanism that allows the proper quality of service level to be requested at the given physical location where this trigger is located. Different triggers at various locations could be preset with different levels of requested quality of service.

On the other hand, initiators are those entities that actually request the data transfer to occur from a source to a receiver. A typical example of an initiator in this case is the physician's PDA that requests data transfer to occur from a data store (source) to some receiver. The receiver need not be the initiator in all cases, but in most cases it is.

Although an initiator usually requests initiation of data transfer from the data source to itself, the initiator may request data transfer from the data source to another receiver or receivers. An example of such scenario includes the transfer of critical patient records to multiple physicians who may be on their way to the patient in case of extreme emergency. This way the commute time of the physicians to the location of the emergency overlaps the time needed to transfer such vital patient information. Fig. 3 shows the interaction amongst the four entities in our mechanism.

Most importantly, it is worthy to observe how quality of service level is configured at the trigger, and how such quality of service level is mandated by the initiator itself prior to any data transfer.

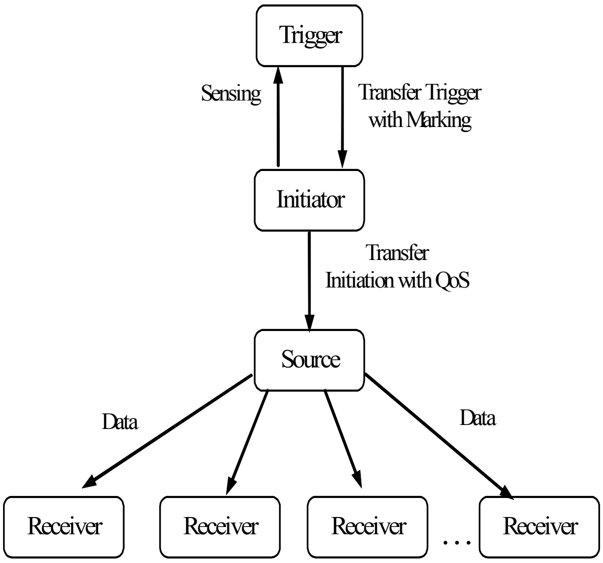

Fig. 3 Entity Interactions

IV. Marking and Classification

Within the classical DiffServ model, packets abiding by the DiffServ model are marked so that they may be given a specific form of treatment upon transmission. The Type of Service (ToS) byte is completely redefined, and the field is called the differentiated services field as shown in Fig. 4.

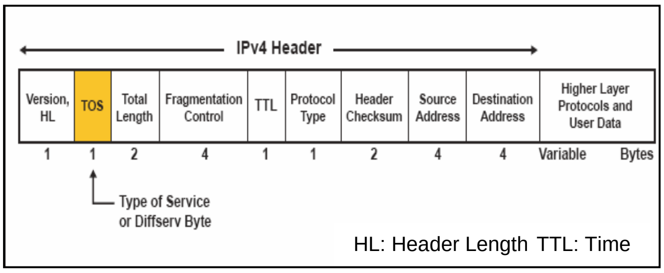

Fig. 4 IPv4 TOS Field

Within our model, every time a trigger entity senses the presence of an anticipated data request, it sends a transfer trigger message to the initiator, which then performs the actual data request from a data source.

However, a trigger present in a normal hospital room for example is very different than a trigger present at an intensive care unit. It is the trigger's responsibility to be properly configured to send the required quality of service level to the initiator. The initiator will then request the data transfer with the given quality of service.

As indicated earlier, there are two primary transmission scenarios. The first being data transfer across multiple hops of network elements, all configured for DiffServ support, and the second primarily being point to point wireless transmission.

In all cases, the initiator will forward the data transfer request to the source, with a given level of desired quality of service called marking. If transmission occurs through multiple hops over DiffServ enabled equipment, the traditional DiffServ model applies, and packets are given preferential treatment using the marking defined in the normal DiffServ model.

However, if communication is done on a point to point wireless communication, the marking sent by the initiator will determine the priority given to the requested data transfer request. Requests with a higher level of quality of service will be given a proportionally higher chance of transmission from the source to the receivers than lower priority tasks.

As such, the same DiffServ marking sent by the initiator is used in two scenarios, the first for the classical application of DiffServ upon transmission over DiffServ enabled equipment, and if communication is done as wireless point to point, marking is used to give preferential transmission priorities.

Generally, we can classify transmission according to two families, namely expedited transmission and assured transmission. In some scenarios, it becomes of high importance to ensure the speediest transmission mechanism of data on a best effort basis. Such applications usually require very low packet loss, guaranteed bandwidth, low delay, and low jitter. In other scenarios, speedy transmission is not the primary concern, but rather assuring a given level of transmission quality based on a service level agreement.

V. Experimental Results

Three common performance parameters are of interest, namely throughput, drop probability, and latency. Throughput is a measure of the amount of data transferred between two points in a specific amount of time. Drop probability is the probability that a given packet of data will not be transmitted upon occurrence of congestion. Latency on the other hand is the time taken to complete a given transmission. All three indicators represent a different viewpoint in the determination of quality of service. We are primarily interested in monitoring the latency associated with various download requests.

Experiments were conducted to compare the latency associated with data downloads under two scenarios: one where absolutely no quality of service guarantees are made, and the second with our quality of service mechanism applied. We define latency as follows:

Latency = Time taken since the initiator's download request until the download is complete.

One thousand initiators arriving at an exponential distribution were used in the experiments, with a lambda arrival rate of two arrivals per second. All initiators perform a similar data download request from a data source. Initiators do not require quality of service guarantees in their entirety, rather, only a proportion of initiators require certain quality of service guarantees.

We conducted our experiments with approximately ten percent of initiators requiring certain quality of service guarantees. In turn, we monitored the download latency of those initiators requiring higher quality of service guarantees.

In the first scenario, we calculate the latency of data downloaded without any quality of service guarantees, and in the second scenario, we apply our quality of service mechanism, and calculate the download latency. The download latency here involves the delay since the download request was made until the data is finally received at the receiver side.

The chart in Fig. 5 illustrates the indicated ten percent of initiators performing the data download requests. No quality of service guarantee is provided in this scenario. Under light load circumstances, the graph illustrates the latency associated with a download as approximately one hundred milliseconds. However, queuing delays cause the latency to deviate from the norm as indicated by the spikes in the curve. The delays can cause a significant increase in latency, depending on the distribution of data download requests made.

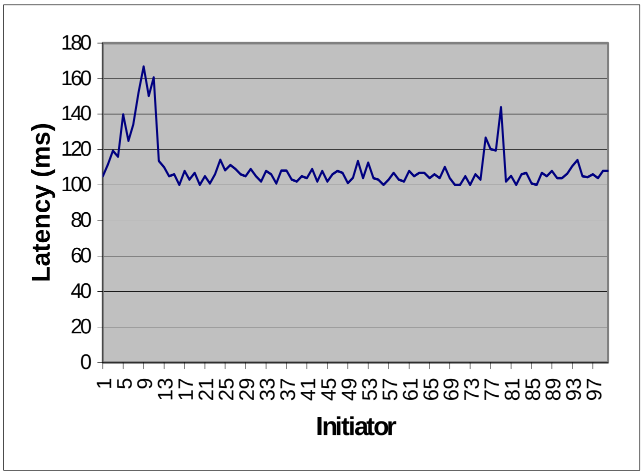

Fig. 5 Latency without QoS

Moreover, Fig. 6 illustrates the download latency, but with our quality of service guarantee applied. Albeit the presence of queuing delays like before, our quality of service mechanism guarantees for tasks requiring high priority, a latency close to the norm indicated earlier, namely one hundred milliseconds. The spikes shown earlier in Fig. 5 are almost entirely eliminated. The tasks requiring quality of service are safely decoupled from any queuing delays. Nevertheless, it is expected that such behavior is only applicable if a reasonable number of users request quality of service guarantees. If a larger number of users require quality of service guarantees, the differentiation amongst various users is not applicable any more.

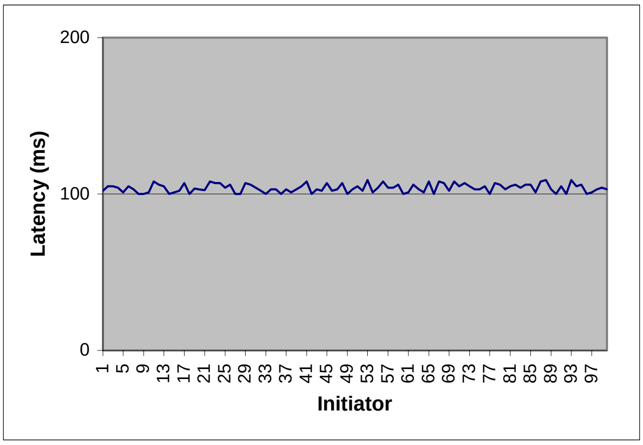

Fig. 6 Latency with QoS

The chart in Fig. 7 illustrates the overall reduction in latency. As shown in the figure, latency reduction is positive for the most part, and contributes towards eliminating the spikes incurred due to queuing delays. However, the overhead of applying the quality of service mechanism can cause a negative reduction in latency at times, yet with almost insignificant levels. The achievement in eliminating the latency reduction lies in the fact that queuing delays for tasks requiring higher quality of service are almost entirely eliminated utilizing this mechanism.

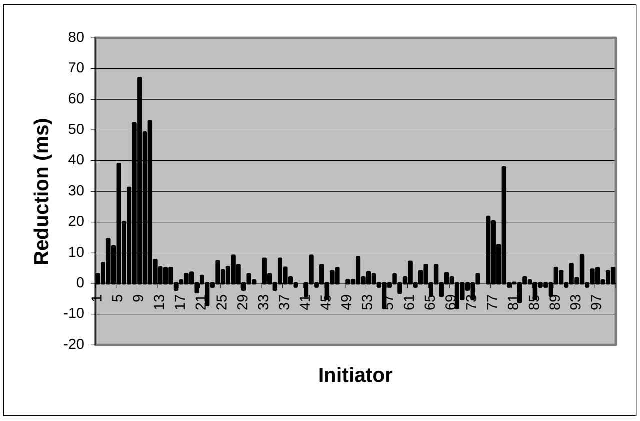

Fig. 7 Latency Reduction

VI. Conclusion

In this paper, we utilized the classical Differentiated Services quality of service mechanism to provide quality of service guarantees to pervasive systems applications, primarily those applications requiring data downloads upon existence within a given context.

Examples of such applications include the automatic download of patient records on a resource-limited PDA upon sensing the presence of the PDA beside a patient's bed. Different quality of service guarantees should be provided depending on the type of patient.

Triggers, initiators, sources and receivers are four entities interacting to provide the quality of service guarantee. Triggers are coupled with various sensors, and initiators are coupled with user devices.

Upon sensing the presence of a user, a trigger will send a message marked with a given requested quality of service level to an initiator coupled with the user. The initiator will then forward the requested download request, along with the quality of service marking to a source. The source will then provide the data to a receiver or receivers with a required quality of service level. We demonstrated how differentiated services, along with source priority queuing are utilized to provide the quality of service guarantee.

Experiments applied on one thousand initiators, of which approximately ten percent require quality of service guarantees were conducted. The graphs show how a quality of service guarantee is achieved, and how latency is decoupled from queuing delays.

References

[1] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges", IEEE Personal Communications, vol. 8, no. 4, pp 10-17, August, 2001.

[2] S. Blake, et al., Internet Engineering Task Force (IETF), RFC 2475, " An Architecture for Differentiated Services". Available: http://www.ietf.org/rfc/rfc2475.txt

[3] J. Heinanen et. al., "Assured Forwarding PHB Group", RFC 2597, June, 1999.

[4] S. Yi et. al, "Providing Fairness in Diffserv Architecture ", IEEE Globlecom 2002, Taiwan, 2002.

[5] D. Clark and W. Fang, "Explicit Allocation of Best-Effort Packet Delivery Service", IEEE/ACM Transactions on Networking, vol. 6, no. 4, pp. 362-373, 1998.

[6] J. Kurose and K. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, 3rd Edition, Addison Wesley, 2004.

[7] P. Chimento, "Tutorial on QoS Support for IP", Technical Report 23, Center for Telematics and Information Technology (CTIT), Netherlands, 1998.

[8] A. Patwardhan et. al, "Enforcing Policies in Pervasive Environments", The International Conference on Mobile and Ubiquitous Systems: Networking and Services, pp. 299-308, Massachusetts, USA, 2004.

[9] Cisco Diffserv, "The Scalable End-to-End QoS Model", Available Online: http://www.cisco.com, 2001.

[10] B. Shirazi, et. al., "QoS Middleware Support for Pervasive Computing Applications", The 37th Hawaii International Conference on System Sciences, Track 9, p. 90294a, USA, 2004.

[11] [11] D. Garlan, et. al., "Project Aura: Towards Distraction-Free Pervasive Computing", IEEE Pervasive Computing, pp. 22-31, April-June, 2002.

[12] M. Esler, "Next Century Challenges: Data-Centric Networking for Invisible Computing", The Portolano Project at the University of Washington, ACM/IEEE MOBICOM 97, pp. 256-262, Seattle, WA, USA, 1997.

[13] A. Aiken, et. Al, Endeavour Project. Available: http://endeavour.cs.berkeley.edu, 1999.

[14] D. De Roure, Equator Project. Available: http://www.iam.ecs.soton.ac.uk/projects/equator, 2003.

Sherif G. Aly received his B.S. degree in Computer Science from the American University in Cairo, Egypt, in 1996. He then received his M.S. and Doctor of Science degrees in Computer Science from the George Washington University in 1998 and 2000 respectively. He worked for IBM during 1996, and later taught at the George Washington University from 1997 to 2000 where he was nominated for the Trachtenberg prize-teaching award for his current scholarship and scholarly debate. He spent two years as a guest researcher for the National Institute of Standards and Technology at Gaithersburg, Maryland from 1998 to 2000. Dr. Aly also worked as a research scientist at Telcordia Technologies in Morristown, New Jersey, in the field of Internet Service Management Research, and as a Senior Member of Technical Staff at General Dynamics Network Systems. He also consulted for Mentor

Graphics and taught at the German University in Cairo. He is currently a faculty member at the Department of Computer Science at the American University in Cairo. Dr. Aly published numerous papers in the area of distributed systems, multimedia, digital design, and programming languages. His current research interests include pervasive systems, programming languages, multimedia, directory enabled networks, and image processing. Dr. Aly is a member of IEEE.