

HealthStack: Providing an IoT Middleware for Malleable QoS Service Stacking for Healthcare 4.0

Vinicius F. Rodrigues, Rodrigo da R. Righi, Cristiano A. da Costa

¹Programa de Pós-Graduação em Computação Aplicada
Universidade do Vale do Rio dos Sinos (UNISINOS)
São Leopoldo, Rio Grande do Sul, Brazil

Abstract. *With smart sensors in hospitals, monitor and process data is crucial to improve medical services. In particular, these devices can assist in making critical decisions that can save lives. In this context, quality of service (QoS) is essential to ensure application data reliability. That said, this article proposes HealthStack, a middleware for operating rooms with automatic QoS and support for real-time data transmission. We also present a QoS strategy based on artificial neurons to select middleware components with critical performance. We developed a prototype of the model and tested it in an actual operating room. The evaluation shows that the strategy improves the average jitter experienced by the application by 90.3%.*

Resumo. *Com a utilização de sensores inteligentes em hospitais, o monitoramento e processamento dos dados é de extrema importância para melhorar serviços médicos. Em particular, esses dispositivos podem auxiliar na tomadas de decisões críticas. Nesse contexto, a qualidade do serviço (QoS) é essencial para garantir a confiabilidade dos dados das aplicações. Dito isso, esse artigo propõe HealthStack, um middleware para salas cirúrgicas com QoS automático e suporte a transmissão de dados em tempo real. O artigo propõe uma estratégia de QoS baseada em neurônios artificiais para seleção dos componentes do middleware com baixa performance. Foi desenvolvido e testado um protótipo do modelo em uma sala de cirurgia real. A avaliação demonstra que a estratégia pode melhorar o jitter médio de aplicações em até 90,3%.*

1. Introdução

Após o surgimento da Indústria 4.0, atualmente já é comum o termo “Hospital 4.0” para se referir à revolução dos hospitais devido à chegada das tecnologias IoT em ambientes de saúde [Aceto et al. 2020]. Em hospitais, existem pacientes em todas as instalações com requisitos de urgência distintos para análises médicas. Neste cenário, o tempo de resposta para situações críticas é decisivo para salvar vidas em cirurgias, por exemplo. A variedade de sensores e aplicações em salas cirúrgicas exige que, os sistemas de tempo real estejam atentos à heterogeneidade dos dados e sua relevância para a escolha das melhores estratégias em seu processamento e transmissão. Isso significa, que a solução deve adaptar suas estratégias de acordo com os dados e requisitos da aplicação (atraso, jitter, etc.) fornecendo qualidade de serviço, comumente chamado de QoS (*Quality of Service*). Nesse contexto, este documento propõe o HealthStack, um middleware multicamada para salas cirúrgicas com suporte dinâmico de QoS para transmissão de dados em tempo real. HealthStack apresenta uma arquitetura com componentes distribuídos e protocolos de

comunicação para fornecer dados em tempo real de sensores para aplicativos em uma forma de publicar-assinar (PubSub). A estratégia de QoS consiste em empilhar serviços de QoS para cada componente da arquitetura de middleware dinamicamente, de acordo com as violações de QoS no lado dos aplicativos. Para avaliar o modelo, implantamos o HealthStack em uma sala de cirurgia equipada com sensores de imagem de profundidade e tags de banda ultralarga (UWB) para rastreamento de posição interna. As principais contribuições apresentadas por este artigo são: (i) um middleware para salas cirúrgicas com suporte QoS automático para transmissão de dados em tempo real; e (ii) estratégia de QoS baseada em neurônios artificiais para selecionar componentes de middleware com desempenho crítico.

2. Trabalhos Relacionados

A Tabela 1 apresenta as principais características dos trabalhos analisados. Com base na análise da literatura, três lacunas de pesquisa principais são aparentes: (i) as estratégias não se concentram em ambientes hospitalares críticos; (ii) as soluções não combinam estratégias múltiplas em níveis diferentes; e (iii) iniciativas não colocam esforço especificamente em transmissões de dados em tempo real. Primeiro, a maioria das estratégias concentram-se em saúde remota, como lares de idosos, e apenas WBANs para monitoramento da saúde do paciente. Em segundo lugar, um sistema de saúde tem dois atores principais: os sensores no nível do *hardware*, gerando dados, e os usuários no nível de *software*, que consomem e processam os dados do sistema. As estratégias se concentram principalmente no primeiro nível para fornecer QoS na transmissão de dados do sensor para a rede. Embora algumas iniciativas apresentem preocupações em relação ao tempo real, elas não se concentram intensamente nesta questão. Em geral, essas soluções se consideram apenas melhorar o atraso de tempo de pacotes prioritários ou fornecer uma arquitetura que suporte transmissões de dados em tempo real.

Tabela 1. Artigos focados em estratégias de QoS para soluções na área da saúde.

Artigo	Tempo real	Foco	Método QoS
[Samanta and Misra 2018]	✓	WBAN	Protocolo de escalonamento (priorização de dados)
[Samanta et al. 2018]		WBAN	Protocolo de escalonamento com priorização de dados, controle de admissão
[Guezguez et al. 2018]	✓	WSN, WBAN	Priorização de dados, controle de admissão, protocolo de roteamento
[Wang et al. 2018]		WBAN	Controle de energia
[Sodhro et al. 2019]	✓	Telemedicina	Suavização de vídeo
[Wang, Q. et al. 2019]		Telemedicina	Priorização de dados
[Bai et al. 2019]		WBAN	Sincronização de relógio
[Goyal et al. 2020]	✓	WBAN	Priorização de dados e adaptação de taxa de dados

3. O Modelo HealthStack

O modelo Healthstack se concentra em integrar dados de múltiplos sensores em ambientes hospitalares e fornecê-los em um formato padrão aos aplicativos. HealthStack é composto por Medical Data Collectors, que são componentes com acesso direto aos dados de dispositivos oferecendo serviços de QoS individualmente. A Figura 1 ilustra a arquitetura do modelo em detalhes. O Core é responsável por colectar dados dos múltiplos Collectors que fornecem dados dos sensores. Além disso, esse módulo processa os dados os persistindo em um banco de dados e também os encaminhando para aplicações. No nível das aplicações, a estratégia fornece uma biblioteca baseada em PubSub as quais as aplicações

de usuário podem se inscrever em streaming de dados de sensores. Essa biblioteca abstrai métricas de QoS calculadas quando dados são entregues para as aplicações. O módulo QoS Manager é encarregado de monitorar todo o sistema e realizar os ajustes necessários nas pilhas de serviços QoS.

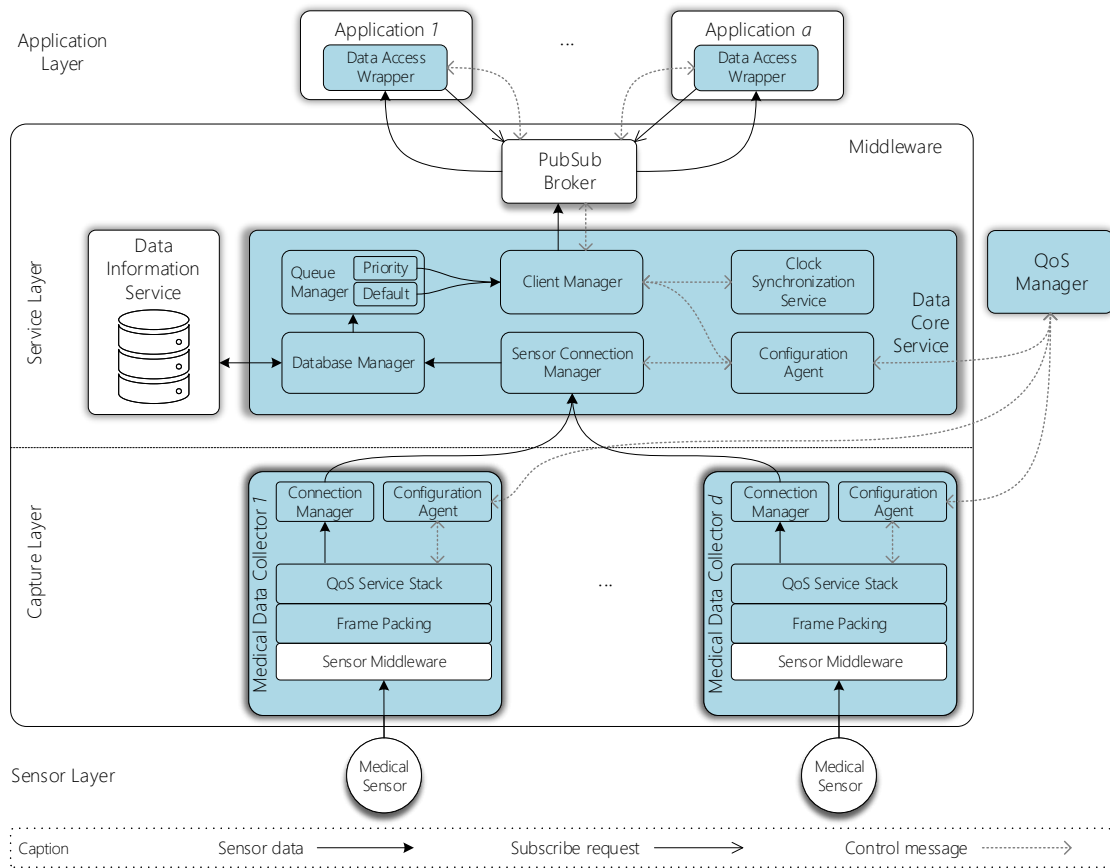


Figura 1. Arquitetura do modelo HealthStack e seus componentes.

A elasticidade de serviços QoS consiste em pilhas de serviço de QoS elásticas individuais para cada instância de Collector. O Manager gerencia essas pilhas de acordo com os requisitos das aplicações e o desempenho atual do middleware. Ele monitora periodicamente se os limites estabelecidos de QoS (delay ou jitter) são respeitados ou não. Assim que ocorre uma violação de limite, o Manager empilha diferentes serviços de QoS para os dispositivos sensores para resolver o problema. Portanto, as pilhas de serviço são elásticas e o Core muda quando necessário. O middleware compreende quatro serviços de QoS diferentes: (i) Priorização de dados; (ii) Compressão de dados; (iii) Adaptação de taxa de dados; e (iv) Elasticidade de recursos.

Para analisar, planejar e executar adaptações, o Manager monitora periodicamente várias variáveis de desempenho e status dos componentes e aplicativos de middleware. No lado da aplicação de usuário, duas variáveis medem os valores atuais de delay ($delay_a$) e jitter ($jitter_a$) que o middleware entrega para uma dada aplicação a . Os Collectors têm sete variáveis diferentes, uma vez que, essas instâncias são as principais responsáveis por adquirir e empacotar as amostras dos sensores: CPU (cpu_s), memória (mem_s), rede (net_s), delay ($delay_s$), jitter ($jitter_s$), conexões ($conn_s$), e prioridade (pri_s). As duas

últimas variáveis representam, respectivamente, quantas aplicações estão consumindo dados do Collector s , e o parâmetro de nível de prioridade do sensor para este Collector s . Dadas todas as variáveis de entrada, o modelo emprega um processo de orquestração de serviços para identificar violações de QoS e realizar adaptações no middleware. O objetivo principal deste processo é selecionar uma instância de Collector em cada ciclo de monitoramento e empilhar serviços de QoS nela se houver pelo menos uma violação de QoS no lado das aplicações. Para resolver esse problema, o modelo propõe a métrica Potencial de Adaptação (PA) para calcular a probabilidade de selecionar um determinado Collector. PA é modelado como um neurônio artificial com base nos conceitos de redes neurais artificiais. Mais especificamente, ele emprega o neurônio sigmoide, que usa a função de ativação sigmoide para produzir sua saída [Nielsen 2015]. Sua saída é produzida por $\sigma(W \cdot X + b)$, em que σ é a função sigmoide logística $\sigma(x) = \frac{1}{1+e^{-x}}$.

4. Metodologia de Avaliação

Foi desenvolvido um protótipo de todos os módulos da arquitetura utilizando a linguagem de programação C++. Para a avaliação do modelo, foi instalado um protótipo de HealthStack em um hospital real parceiro do projeto. Foi instalado um sistema de localização em tempo real da Sewio (<https://www.sewio.net/>), o qual emprega tecnologia UWB. Em adição, foi instalado um dispositivo Microsoft Kinect para extração de imagens de profundidade. Para adquirir dados dos dispositivos, foram empregados 2 computadores Dell Optiplex 3050 Mini, processador Intel i7-6560U 2.20 GHz, 8GB de memória, placa de rede de 1 Gbps. No momento de realização desta pesquisa, não havia disponível na literatura benchmarks de carga de trabalho para aplicações médicas em tempo real. Portanto, foi modelado um cenário composto por 48 aplicações que se conectam ao HeathStack sequencialmente, uma a cada seis segundos. Cada nova aplicação (App) solicita um tipo de dados diferente do anterior. Especificamente, App 1 solicita dados de imagem do Kinect, App 2 dados de localização, App 3 dados de imagem, e assim por diante. Além disso, as aplicações definem seu limite de QoS de destino na seguinte sequência: delay=50ms, delay=50ms, jitter=25ms, jitter=25ms, e assim sucessivamente. A avaliação consistiu em observar a execução do middleware e das aplicações em uma janela de dez minutos em dois cenários: modo da elasticidade de serviços QoS (i) habilitado e (ii) desabilitado.

5. Análise dos Resultados

A Figura 2 ilustra a variação no delay médio para cada aplicação ao habilitar a elasticidade de serviços QoS. Os valores negativos representam um delay médio inferior, o que é uma melhoria. Como a aplicação solicita diferentes tipos de informações, os resultados estão segmentados de acordo com o tipo de dado para analisar a relevância. A Figura 2 (a) demonstra que a maioria das aplicações que consomem dados de imagem experimentaram melhorias, mas nem todos eles. De um total de 24, 18 aplicações (75%) apresentaram melhorias no delay médio. Por outro lado, olhando a Figura 2 (b), as aplicações que consomem dados de localização alcançaram resultados mais significativos. Apenas uma entre 24 aplicações alcançou um delay médio mais alto ao habilitar a elasticidade de serviços QoS. Comparando os dois diagramas (a) e (b), o tamanho dos dados transmitidos às aplicações do primeiro conjunto é consideravelmente maior do que os dados do segundo conjunto. Os dados de imagem requerem mais processamento e tempo de transmissão do que os dados de localização.

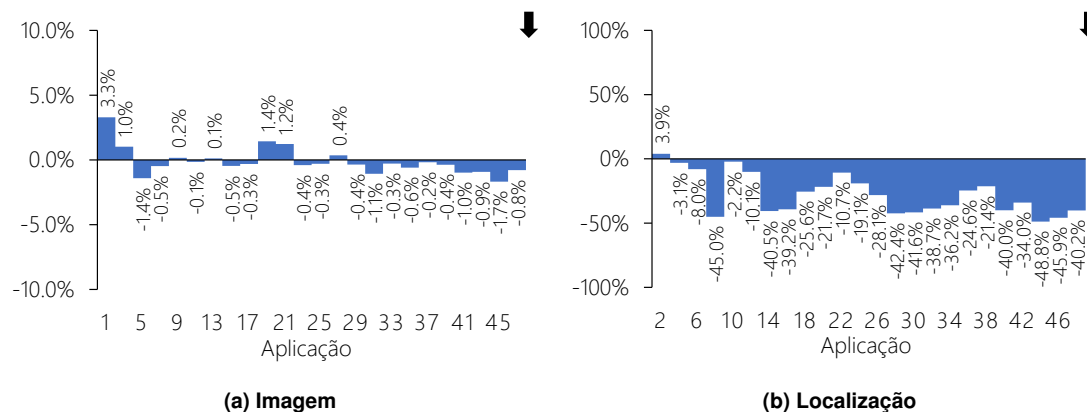


Figura 2. Variação no delay das aplicações com elasticidade de serviços QoS.

Agora, focando a métrica jitter, a Figura 3 descreve a variação no jitter médio para cada aplicação ao habilitar a elasticidade dos serviços QoS. Aqui, as variações são mais expressivas do que as de delay. No entanto, diferentes tipos de dados resultaram em valores contrastantes. Por um lado, a Figura 3 (a) demonstra os resultados para aplicações que consomem dados de imagem. Apenas cinco das 24 aplicações tiveram melhorias em seu jitter. Embora não demonstrando muitas melhorias, o jitter resultante ainda é baixo. Isso demonstra que habilitar a elasticidade de serviços QoS resulta em um melhor atraso não impondo maiores penalidades de jitter, já que um jitter menor que 25ms é totalmente aceitável [Nanda and Fernandes 2007]. Por outro lado, a Figura 3 (b) mostra que todas as aplicações que consomem dados de localização tiveram melhorias consideráveis. Considerando as 24 aplicações deste diagrama, a média de seu jitter médio foi de 2,1ms ao habilitar a elasticidade de serviço QoS contra 21,7ms ao desabilitá-la. Representa uma diminuição substancial em 90,3%. A principal razão para isso é que o sistema de localização não fornece amostras de posições de tags a uma taxa estável, impactando diretamente no jitter. Nesse cenário, o serviço de adaptação da taxa de dados desempenha um papel essencial porque estabiliza a taxa de dados para um valor mais baixo.

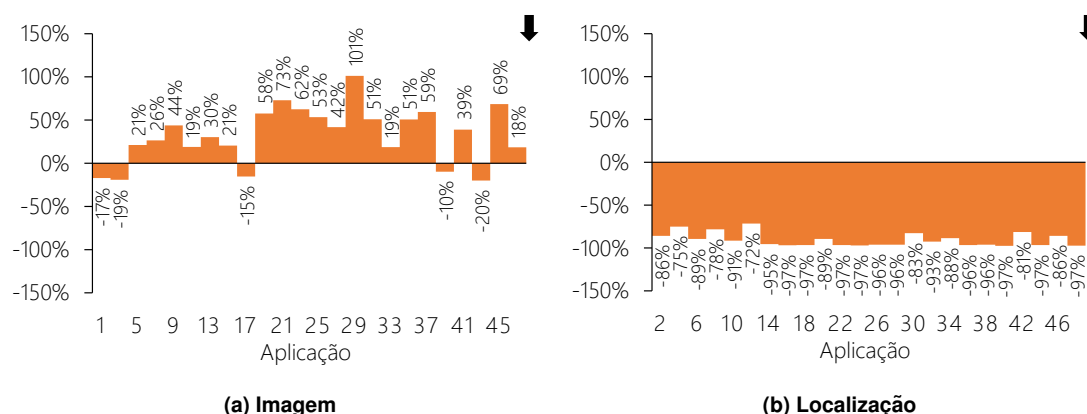


Figura 3. Variação no jitter das aplicações com elasticidade de serviços QoS.

6. Conclusão

Este artigo apresentou um modelo de middleware para ambientes hospitalares 4.0 chamado HealthStack. Diferente de trabalhos relacionados que empregam estratégias isoladas de QoS, HealthStack provê elasticidade de serviços QoS, visando fornecer QoS para aplicações de acordo com a demanda. A contribuição é viabilizada através do monitoramento de desempenho de delay e jitter das aplicações e execução ações caso limites

sejam violados. As principais contribuições apresentadas por este artigo são: (i) um middleware para salas cirúrgicas com suporte QoS automático para transmissão de dados em tempo real; e (ii) estratégia de QoS baseada em neurônios artificiais para selecionar componentes de middleware com desempenho crítico. Além da contribuição técnico-científica, HealthStack também traz contribuições ao hospital e aos pacientes em nome da sociedade. Visando os hospitais do futuro, o modelo oferece estratégias para garantir o desempenho de aplicações em tempo real que processam dados de processos médicos. Trabalhos futuros podem derivar desta arquitetura e incluir outros dados e serviços QoS.

Agradecimentos

Os autores gostariam de agradecer a CAPES (Código Financeiro 001), o CNPq (Números de concessão 309537/2020-7), e a FAPERGS (Código 21/2551-0000118-6).

Referências

- Aceto, G., Persico, V., and Pescapé, A. (2020). Industry 4.0 and health: Internet of things, big data, and cloud computing for healthcare 4.0. *Journal of Industrial Information Integration*, 18:100129.
- Bai, T., Lin, J., Li, G., Wang, H., Ran, P., Li, Z., Pang, Y., Wu, W., and Jeon, G. (2019). An optimized protocol for qos and energy efficiency on wireless body area networks. *Peer-to-Peer Networking and Applications*, 12(2):326–336.
- Goyal, R., Patel, R., Bhaduria, H., and Prasad, D. (2020). An energy efficient qos supported optimized transmission rate technique in wbans. *Wireless Pers. C.*, pages 1–26.
- Guezguez, M. J., Rekhis, S., and Boudriga, N. (2018). A sensor cloud for the provision of secure and qos-aware healthcare services. *Arabian Journal for Science and Eng.*, 43(12):7059–7082.
- Nanda, P. and Fernandes, R. C. (2007). Quality of service in telemedicine. In *First Int. Conf. on the Digital Society (ICDS'07)*, pages 2–2.
- Nielsen, M. A. (2015). *Neural Networks and Deep Learning*, volume 2018. Determination Press San Francisco, CA, USA:.
- Samanta, A., Li, Y., and Chen, S. (2018). Qos-aware heuristic scheduling with delay-constraint for wbsns. In *2018 IEEE Int. Conf. on Communications (ICC)*, pages 1–7.
- Samanta, A. and Misra, S. (2018). Dynamic connectivity establishment and cooperative scheduling for qos-aware wireless body area networks. *IEEE Transactions on Mobile Comp.*, 17(12):2775–2788.
- Sodhro, A. H., Luo, Z., Sangaiah, A. K., and Baik, S. W. (2019). Mobile edge computing based qos optimization in medical healthcare applications. *Int. Journal of Information Management*, 45:308 – 318.
- Wang, J., Sun, Y., and Ji, Y. (2018). Qos-based adaptive power control scheme for co-located wbans: a cooperative bargaining game theoretic perspective. *Wireless Networks*, 24(8):3129–3139.
- Wang, Q. et al. (2019). Enable advanced qos-aware network slicing in 5g networks for slice-based media use cases. *IEEE Transactions on Broadcasting*, 65(2):444–453.