Uma Abordagem Ciente de Contexto na IoT Aplicada à Pesquisa Agropecuária

Patrícia Davet*, Renato Dilli*, Lucas Xavier*‡, Rodrigo Souza†, Nelsi Warken§,
Anderson Afonso Cardozo‡, Nicácia Machado‡, Wemerson Parreira‡, Adenauer Yamin*‡

***Resumo*— As aplicações introduzidas pela Internet das Coisas, em função de sua escalabilidade, possuem em sua maioria caráter autônomo, e para tal necessitam terem ciência das informações contextuais que lhe interessam e, quando for o caso, reagirem as mesmas. Este artigo apresenta o projeto plenUS e a arquitetura de software para Ciência de Contexto que lhe dá suporte, denominada EXEHDA-IoT. Para avaliar as funcionalidades do EXEHDA-IoT, foi desenvolvido um estudo de caso junto a Embrapa Clima Temperado, empregando tecnologias de hardware disponíveis no mercado e soluções de software de natureza *open-source*. Os resultados obtidos até o momento se mostraram promissores.**

***Palavras-Chave*— Ciência de Contexto, Internet das Coisas, Agropecuária.**

***Abstract*— Applications introduced by the Internet of Things, due to its scalability, have mostly autonomous character, and for this, they need to be aware of contextual information that interest you and, when necessary, react the same. This paper presents the PlenUS project and its software architecture for Context-Awareness support, called EXEHDA-IoT. To evaluate the functionalities of the EXEHDA-IoT, a case study was developed with Embrapa Temperate Climate, using available hardware technologies and open-source software solutions. The results obtained until now are promising.**

***Keywords*— Context-Awareness, Internet of Things, Agricultural and Livestock.**

I. INTRODUÇÃO

O projeto plenUS (*plentiful Ubiquitous Systems*) tem como premissa o emprego do middleware EXEHDA com o objetivo de disponibilizar Sistemas Ubíquos, que explorem relações pró-ativas entre usuários, softwares e equipamentos, auxiliando nas pesquisas em desenvolvimento na Embrapa Clima Temperado. O mesmo vem sendo desenvolvido por um consórcio de pesquisa envolvendo o LUPS (*Laboratory of Ubiquitous and Parallel Systems*) da Universidade Federal de Pelotas (UFPEL) e a Embrapa Clima Temperado.

De modo mais específico o projeto plenUS tem como foco atender as demandas referentes ao provimento de serviços computacionais cientes de contexto, permitindo um registro histórico dos estados contextuais em que se encontram os equipamentos dos seguintes laboratórios da Embrapa: (i) Laboratório de Qualidade do Leite (Lableite), (ii) Laboratório de Reprodução Animal e (iii) Laboratório de Análise de Sementes Oficial (LASO), como também outras necessidades de sensoriamento em ambientes administrativos como o do Núcleo de Tecnologia da Informação (NTI), e de uma Sala de Testes do plenUS.

Com a implementação do sistema de qualidade nestes laboratórios, aumentaram as exigências no controle dos equipamentos onde são realizadas as análises, visando precisão e confiabilidade nos resultados. Deste modo, tornam-se necessários registros das leituras de grandezas físicas como temperatura, umidade e luminosidade dos equipamentos envolvidos (germinadores, câmaras frias, salas de trabalho, etc.), bem como uma atuação automatizada em função das informações sensoriadas, sempre que for o caso.

A infraestrutura computacional proporcionada pela Internet das Coisas (Internet of Things - IoT) possibilita que estes equipamentos possam trocar informações entre si e/ou com usuários via Internet, permitindo o desenvolvimento de aplicações com maior valor agregado. Isto é possível devido a incorporação de dispositivos, unicamente identificados, e com capacidade de sensoriamento, comunicação, armazenamento e processamento [1].

Porém, para o desenvolvimento de aplicações neste cenário da IoT, há uma série de funcionalidades que devem ser providas, envolvendo desde a aquisição de informações contextuais adquiridas por sensores, a partir de um conjunto de fontes heterogêneas e distribuídas, até a representação dessas informações, seu processamento, armazenamento, e realização de inferências para seu uso em procedimentos de atuação [2].

Neste sentido, este trabalho tem como objetivo a concepção de uma arquitetura de software comprometida com a premissa de fornecer mecanismos para o desenvolvimento de aplicações cientes de contexto na IoT, denominada EXEHDA-IoT. Esta arquitetura foi modelada segundo os pressupostos do CoIoT (Context+IoT) e sua avaliação aconteceu junto ao projeto plenUS. O CoIoT especializa o middleware EXEHDA (*Execution Environment for Highly Distributed Applications*) no tratamento das questões de Ciência de Contexto na IoT [3].

O artigo está organizado da seguinte forma: a seção 2 descreve a modelagem e principais funcionalidades da arquitetura de software proposta. A seção 3 apresenta a avaliação da arquitetura pelo projeto plenUS. A seção 4 discute os trabalhos relacionados, e, por fim, a seção 5 apresenta as considerações finais.

*Universidade Federal de Pelotas (UFPEL), Pelotas, RS, Brasil.
Emails: {ptdavet, renato.dilli, lmdsxavier}@inf.ufpel.edu.br

†Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brasil. Email: rssouza@inf.ufrgs.br

‡Universidade Católica de Pelotas (UCPEL), Pelotas, RS, Brasil.
Emails: anderson.cardoso@ucpel.edu.br, nicacia@gmail.com,
{wemerson.parreira, adenauer.yamin}@ucpel.edu.br

§Embrapa Clima Temperado, Pelotas, RS, Brasil.
Email: nelsi.warken@embrapa.br

II. EXEHDA-IoT: Arquitetura e Funcionalidades

O EXEHDA é um middleware baseado em serviços que visa criar e gerenciar um ambiente ubíquo, bem como promover a execução de aplicações sobre esse ambiente. O mecanismo de adaptação proposto para o EXEHDA emprega uma estratégia colaborativa entre aplicação e ambiente de execução, através da qual é facultado ao programador individualizar políticas de adaptação para reger o comportamento de cada um dos componentes que constituem o software da aplicação [4].

O fato do EXEHDA ter uma origem enquanto middleware para UbiComp (Computação Ubíqua), faz com que o tratamento da Ciência de Contexto seja inerente à sua arquitetura. Para que o EXEHDA continue provendo Ciência de Contexto tendo em vista os desafios do cenário da IoT, onde a aquisição das informações de contexto ocorre a partir de um grande número de dispositivos heterogêneos, distribuídos dinamicamente e com capacidade restrita, está em desenvolvimento o CoIoT [3].

Considerando isso, O EXEHDA-IoT propõe um ambiente computacional constituído por células de execução, nas quais os dispositivos computacionais são distribuídos (vide Figura 1). Cada célula é constituída dos seguintes componentes: (i) EXEHDAbase, o elemento central da célula, sendo responsável por todos serviços básicos e constituindo referência para os demais elementos; (ii) o EXEHDAnodo que corresponde aos dispositivos computacionais responsáveis pela execução das aplicações; (iii) o EXEHDAnodo móvel, um subcaso do anterior, que corresponde aos dispositivos tipicamente móveis que podem se deslocar entre as células do ambiente ubíquo, como *notebooks*, *tablets* ou *smartphones*, (iv) o EXEHDAborda, responsável por fazer a interoperação entre os serviços do middleware e os diversos tipos de *gateways*; e (v) o EXEHDAgateway, que consiste no elemento responsável por setorizar pontos de coleta e/ou atuação distribuídos, disponíveis no meio físico, realizando a interação destes com os outros componentes do middleware.

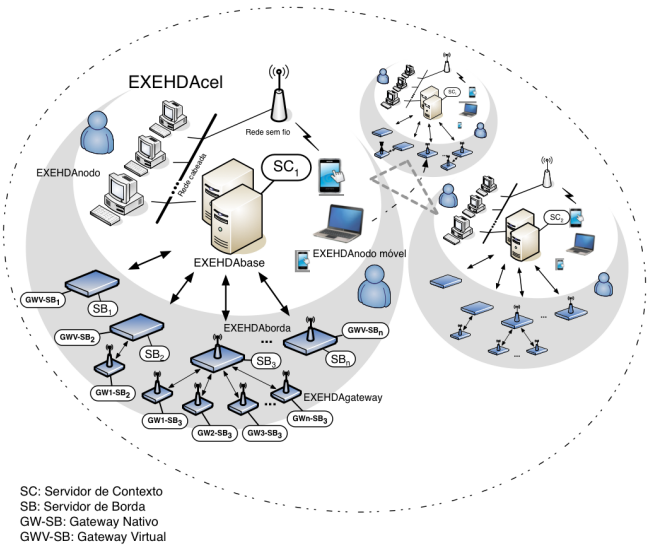

Fig. 1. Organização Celular do Ambiente Ubíquo Gerenciado pelo EXEHDA-IoT

Para provimento de Ciência de Contexto na IoT, o EXEHDA-IoT se vale de dois tipos principais de servidores: Servidor de Borda e Servidor de Contexto (vide Figura 1). O Servidor de Borda é instanciado em um equipamento do tipo EXEHDAborda e se destina a gerenciar a interação com o meio físico através de *gateways*, bem como a execução de regras de contingência e armazenamento temporário das informações contextuais coletadas, em caso de falha de comunicação. O Servidor de Contexto, por sua vez, é alocado no EXEHDAbase e atua no armazenamento e no processamento das informações contextuais, integrando informações históricas e aquelas provenientes de diferentes Servidores de Borda distribuídos no ambiente ubíquo.

Na proposta do EXEHDA-IoT a premissa é que os sensores e/ou atuadores sejam incorporados ao Servidor de Borda através de *gateways*. Os *gateways* são utilizados então, para tratar a heterogeneidade, tanto de hardware como de software, inerentes a dispositivos de sensoriamento e/ou atuação, realizando a conversão de protocolos e o gerenciamento dos dispositivos, além de fornecer a estes capacidade de comunicação via Internet.

Outrossim, o EXEHDA-IoT possibilita a identificação e descoberta automática dos dispositivos de sensoriamento e/ou atuação a partir dos *gateways* integrados à arquitetura, por meio do protocolo de comunicação UPnP (*Universal Plug and Play*) [5]. As características de descoberta e autoconfiguração de dispositivos inerentes ao UPnP, que constituem um facilitador quando da inclusão e/ou remoção de dispositivos em ambientes escaláveis e dinâmicos, associado ao fato deste utilizar protocolos e padrões consolidados da Internet, tais como HTTP, SOAP e XML, tornam o UPnP bastante utilizado no cenário da IoT.

Com o intuito de garantir a interoperabilidade com as tecnologias de mercado, e também potencializar a distribuição das iniciativas de coleta e/ou atuação, foram previstos três tipos distintos de *gateways*:

- Gateways Proprietários (GW-PR), que tem funcionalidades e protocolos heterogêneos variando de acordo com seus fabricantes;
- Gateways Nativos (GW-SB), cujas funcionalidades operam de maneira integrada à arquitetura do EXEHDA, sendo instanciados em equipamentos do tipo EXEHDAgateway, e;
- Gateway Virtual (GWV-SB), localizado no mesmo hardware em que executa o Servidor de Borda, constituindo-se em uma virtualização do Gateway Nativo, dispensando quando possível a necessidade de um hardware específico para o *gateway*.

A arquitetura de software do EXEHDA-IoT (vide Figura 2) provê comunicação: (i) entre os *gateways* e o Servidor de Borda; (ii) entre os Servidores de Borda e o Servidor de Contexto; (iii) entre os Servidores de Contexto localizados em diferentes células do ambiente ubíquo; e (iv) com outros serviços do *middleware* ou aplicações. As linhas tracejadas na representação dos *gateways* indicam que algum dos tipos de *gateways* podem não estar presentes na arquitetura, a qual possibilita utilizar somente *gateways* físicos (Nativos ou Proprietários), somente Gateway Virtual, ou ambos.

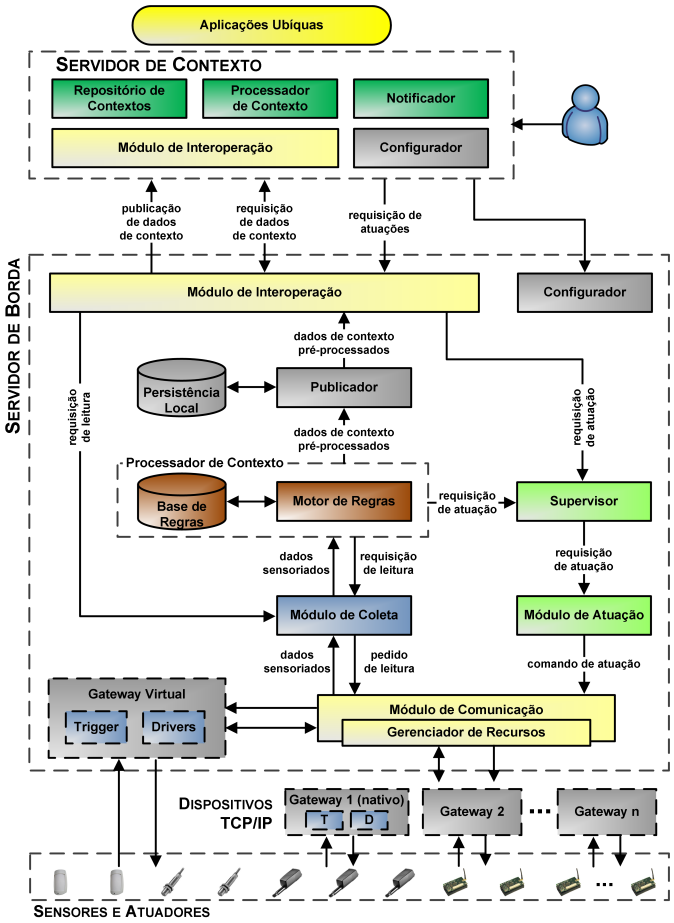

Fig. 2. Arquitetura de Software do EXEHDA-IoT

A. Servidor de Contexto

O Servidor de Contexto no EXEHDA-IoT tem suas funcionalidades organizadas através dos seguintes módulos arquiteturais.

Repositório de Contextos: armazena dados de configuração da arquitetura, e os valores sensoriados publicados pelos Servidores de Borda existentes no ambiente ubíquo.

Processador de Contexto: realiza tarefas de manipulação e dedução das informações contextuais. Baseia-se em regras do tipo ECA (Evento-Condição-Ação), de forma a tratar eventos gerados a partir de mudanças nos estados dos contextos de interesse.

Notificador: utiliza uma estratégia de notificação baseada no modelo *publish/subscribe*, em que recebe subscrições de todos os serviços e/ou aplicações que requerem notificações a respeito das mudanças nos estados contextuais. Também é função do Notificador enviar comandos de atuação aos Servidores de Borda quando necessário, bem como a realização de notificação pela aplicação Web de novo *gateway* descoberto, o qual só será utilizado mediante autenticação pelo usuário interessado.

Módulo de Interoperação: considerando o perfil inerentemente distribuído das aplicações alvo do EXEHDA-IoT, o Módulo de Interoperação foi concebido para gerenciar a troca de informações com os Servidores de Borda, bem como os demais serviços do middleware em execução em outros equipamentos, como também as aplicações, oferecendo para isso uma interface padronizada e bem definida. A concepção deste módulo teve como referência o estilo arquitetural REST, sendo utilizado o protocolo HTTP e suas operações (GET, PUT, POST, DELETE) no suporte às trocas de informações.

Configurador: aglutina todos os procedimentos de configuração necessários para o funcionamento do EXEHDA-IoT. Através deste módulo é disponibilizada uma aplicação através da qual é possível gerenciar dispositivos *gateways* e regras de processamento de contexto, bem como realizar todo tipo de cadastro necessário ao funcionamento da aplicação IoT ciente de contexto.

B. Servidor de Borda

No EXEHDA-IoT o Servidor de Borda tem suas funcionalidades sistematizadas através dos seguintes módulos arquiteturais.

Módulo de Interoperação: é destinado a gerenciar a comunicação com o Servidor de Contexto, tendo também como referência o estilo arquitetural REST. Com isso, todas as funcionalidades associadas ao Servidor de Borda, *gateways*, sensores e atuadores são mapeadas na forma de recursos e acessadas via URIs (*Uniform Resource Identifiers*).

Configurador: é responsável pelos ajustes necessários para o funcionamento do Servidor de Borda. Entre as funcionalidades oferecidas tem-se: definição do Servidor de Contexto para a publicação das informações sensoriadas, bem como a inserção, atualização, ativação e desativação dos *gateways* e seus serviços por meio do protocolo de comunicação UPnP.

Publicador: promove a publicação de todas as informações contextuais coletadas e garante uma **Persistência Local** das mesmas nos períodos em que a publicação ficar inviabilizada. As transações ocorrem sobre o protocolo HTTPS (*HyperText Transfer Protocol Secure*), o que garante que as informações serão transferidas criptografadas, evitando acessos indevidos aos mesmos.

Processador de Contexto: é responsável por tratar regras de contingência, com a intenção de evitar que os dispositivos envolvidos atinjam estados críticos, o que faz com que este módulo adquira elevada importância quando a comunicação com o Servidor de Contexto for interrompida por problemas na infraestrutura de rede.

Supervisor: possui como objetivo aglutinar os comandos de atuação. Uma vez recebidos os parâmetros para controle da atuação, o módulo Supervisor avalia eventuais conflitos entre as requisições de atuação oriundas de diferentes fontes, para após enviar o comando de atuação ao Módulo de Atuação.

Módulo de Atuação: recebe do Supervisor comandos com o ID do atuador e os correspondentes padrões de operação (tempo de duração, potência de ativação, etc.), sob demanda tanto do Motor de Regras, como do Servidor de Contexto ou das aplicações a qualquer momento e os envia ao Módulo de Comunicação a fim de serem interpretados e direcionados ao *gateway* responsável pelo efetivo tratamento.

Módulo de Coleta: trata os pedidos de coleta recebidos de forma assíncrona, repassando o ID do sensor a ser lido pelo

gateway responsável. Também tem a função de direcionar aos demais elementos da arquitetura as informações sensoriadas enviadas pelos *gateways*, produzidas a partir de eventos do ambiente.

Módulo de Comunicação: gerencia a comunicação com os diferentes tipos de *gateways*, além de prover aos demais módulos da arquitetura acesso padronizado aos recursos disponibilizados pelos *gateways* no que tange ao tratamento de sensores e/ou atuadores.

Gerenciador de Recursos: administra procedimentos, como a entrada e saída de operação de dispositivos, bem como a reposição por avaria ou necessidade de adicionar outro ponto de monitoramento. É neste módulo que é instanciado o Ponto de Controle UPnP, que possui como tarefa descobrir os dispositivos UPnP presentes na rede, armazenando as informações sobre os dispositivos e seus respectivos serviços descobertos (sensores e/ou atuadores) em um diretório, para eventual controle destes.

Gateway Virtual: realiza o interfaceamento entre sensores e/ou atuadores (dispositivos sem conexão Internet) ligados diretamente ao Servidor de Borda. Possui dois módulos básicos: **Drivers** e **Trigger**. Os Drivers são responsáveis pelo encapsulamento individualizado de aspectos tecnológicos, tanto de sensores como atuadores, o que evita que as suas diferenças operacionais se projetem nos demais módulos da arquitetura. O Trigger é um módulo que tem a finalidade de gerenciar a leitura dos sensores através de eventos. Consiste em um tipo simplificado de regra de produção, que dá suporte a um conjunto limitado de funções para o tratamento de condições e ações.

III. EXEHDA-IoT: Avaliação no Domínio Agropecuário

As funcionalidades propostas para o EXEHDA-IoT, foram avaliadas através de estudo de caso no domínio agropecuário relacionado ao projeto plenUS. Os ambientes priorizados pelo projeto, para recebimento de infraestrutura para ciência do contexto em que se encontram seus equipamentos, bem como os tipos de grandezas mensuradas e a disposição dos equipamentos utilizados podem ser visualizados na Figura 3.

Considerando a estrutura física do estudo de caso, um Servidor de Contexto, cinco Servidores de Borda e quatro Gateways Nativos foram instalados para atender as demandas da Embrapa como um todo (vide Figura 3), constituindo uma célula de execução do ambiente ubíquo.

O Servidor de Contexto e o Servidor de Borda 4 (SB 4) foram posicionados no NTI, localizado na unidade Sede da Embrapa Clima Temperado, enquanto os outros Servidores de Borda foram posicionados na unidade Estação Experimental Terras Baixas (ETB), distante 21Km.

O hardware utilizado para o Servidor de Contexto foi um desktop com processador Intel Pentium E2160-1.8GHz de dois núcleos, com 3GB de memória RAM e com o Sistema Operacional Ubuntu Server. Já os Servidores de Borda foram prototipados sobre o Sistema Operacional Raspbian, sendo usado como hardware a Raspberry PI Modelo B+[1].

[1]http://www.raspberrypi.org

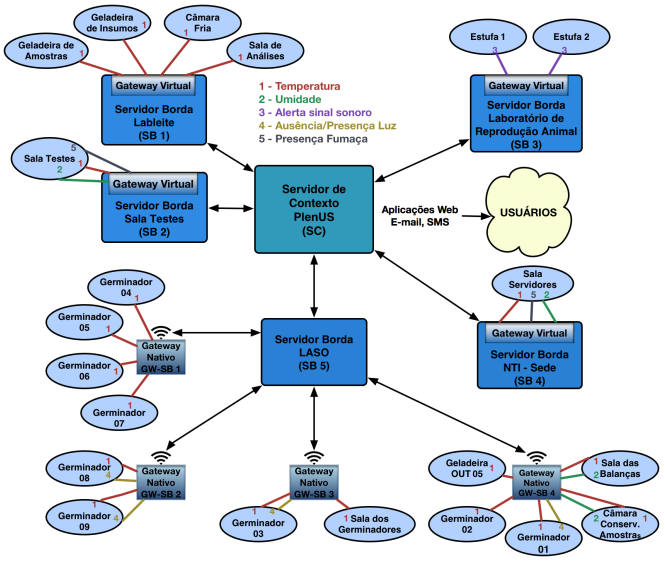

Fig. 3. Estudo de Caso - Projeto plenUS

Quanto aos *gateways* previstos pelo EXEHDA-IoT e utilizados para o estudo de caso, estes são de duas naturezas: **Gateways Virtuais**, internos ao Servidor de Borda (SB), e **Gateways Nativos** com conexão Wi-Fi com o SB. Os Gateways Nativos foram utilizados no caso onde as demandas de sensoriamento introduziram a sua necessidade, seja pela disposição física dos equipamentos e/ou salas ou pela quantidade significativa de sensores.

Para a escolha do hardware utilizado na concepção do Gateway Nativo foram considerados aspectos como: baixo consumo energético, possibilidade de operação em regime sem fio (Wi-Fi) e emprego de plataforma de software disseminada. Após revisão de literatura a escolha recaiu sobre o processador ESP8266-12 [6].

A operacionalização do cenário propiciado pelo estudo de caso é realizada através de regras distribuídas do tipo ECA (Evento-Condição-Ação). Uma contextualização da forma de processamento dessas regras no cenário de avaliação é apresentado na Figura 4 por meio de uma notação BPMN [7].

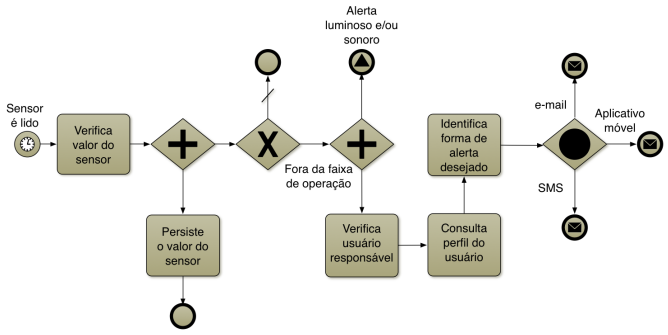

Fig. 4. Processamento de Regra Tipo ECA no Cenário de Avaliação (Notação BPMN)

O cenário consiste em monitorar uma determinada grandeza e produzir registro histórico. Enquanto a grandeza é coletada e armazenada, a mesma tem seu valor avaliado tendo em vista

as faixas de operação para cada dado de contexto coletado. Alertas são enviados aos usuários envolvidos, quando os valores estiverem fora da faixa de operação especificada para o experimento.

A interação do usuário com o cenário é disponibilizada por meio de uma aplicação desenvolvida para navegadores Web, que contempla tanto tarefas de gerenciamento do EXEHDA-IoT como de visualização do registro histórico das informações contextuais, de forma gráfica ou textual.

Para atender o fato da rotina de trabalho dos laboratoristas implicar em uma mobilidade nos diversos recintos do laboratório, a aplicação Web desenvolvida possui características responsivas, o que veio agregar maior funcionalidade ao seu uso, pois possibilitou que as configurações e testes dos dispositivos de sensoriamento e/ou atuação pudessem ser realizadas no local de instalação por meio de *smartphones*, devido ao seu *layout* se adaptar ao dispositivo computacional utilizado.

IV. Trabalhos Relacionados

Uma análise dos trabalhos relacionados EcoDif [8], S^{3}OIA [9], LinkSmart [10] e Carriots [11] em relação à satisfação dos requisitos IoT considerados neste trabalho é apresentada na Tabela I, onde o símbolo "X" denota que o requisito é completamente atendido, o símbolo "O" indica que o requisito é parcialmente atendido, e o símbolo "-" indica que o requisito não é atendido.

TABELA I
Análise dos Trabalhos Relacionados

Plataformas de middleware	Interoperabilidade	Descoberta e Gerenciamento de Dispositivos	Interfaces de Alto Nível	Ciência de Contexto	Escalabilidade
EcoDiF	X	O	X	O	X
S^{3}OIA	X	X	-	X	-
LinkSmart	X	O	-	X	-
Carriots	X	O	X	O	X
EXEHDA-IoT	X	X	X	X	X

Registre-se que o único requisito endereçado de uma maneira ou outra por todos os trabalhos relacionados, foi o de interoperabilidade. Isto é devido ao fato deste ser um requisito primordial e que deve ser imperativamente endereçado por qualquer plataforma de middleware para IoT.

Dentre os trabalhos analisados, o S^{3}OIA destaca-se por ser o único a apresentar uma abordagem de descoberta automática de dispositivos, por meio do protocolo UPnP. De tal forma, o EXEHDA-IoT se utiliza do UPnP como estratégia de autoconfiguração e descoberta automática de seus dispositivos, tendo o gateway como elemento básico do mecanismo de descoberta e a partir dele, os sensores e atuadores são facilmente descobertos e gerenciados. Além disso, a abordagem utilizada considera todo o potencial de protocolos de uso consolidado.

A ciência de contexto é contemplada por todos os trabalhos relacionados, porém o suporte oferecido, como no caso do EcoDif e Carriots mostra-se limitado. Já o EXEHDA-IoT fornece uma organização distribuída para coleta e processamento de contexto através de regras, como também para atuação em ambientes com características heterogêneas e escaláveis. Além disso, por ter sido desenvolvido a partir de um middleware para UbiComp, o EXEHDA, também provê suporte a outros serviços ubíquos disponibilizados pelo middleware, como adaptação, comunicação desacoplada, migração de código, etc.

Outrossim, a abordagem distribuída do processamento do contexto entre os Servidores de Borda e Contexto proposta pelo EXEHDA-IoT também contribui com a minimização do fluxo de informações contextuais em relação as abordagens centralizadas, empregadas pelos trabalhos relacionados.

Por fim, outra abordagem adotada pelo EXEHDA-IoT, que trouxe benefícios principalmente no que tange à otimização do tráfego de rede e ao consumo de energia, é a proposição do uso de *triggers* para gerenciar o fluxo de informações transmitidas.

V. Considerações Finais

O principal diferencial do EXEHDA-IoT em relação aos trabalhos relacionados diz respeito a possibilidade de gerenciar a aquisição, armazenamento e processamento das informações de contexto, de forma distribuída, independente das aplicações, em uma perspectiva autonômica baseada em regras.

Os resultados obtidos pelo estudo de caso, no domínio agropecuário atenderam as demandas dos usuários, além de fornecer um *feedback* para avanços futuros, estimulando a continuidade das pesquisas na área.

Como trabalhos futuros, os seguintes aspectos foram priorizados: (i) explorar estudos de caso em que as regras de processamento contextual utilizem outros mecanismos de inferência de mais alto nível; e (ii) implantação de um *framework* para gerenciamento do motor de regras, tanto no Servidor de Borda como no Servidor de Contexto.

Referências

[1] P. F. Pires, F. C. Delicato, T. Batista, T. Barros, E. Cavalcante, and M. Pitanga, "Plataformas para a internet das coisas," *Livro Texto de Minicursos - SBRC 2015*, 2015.

[2] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 1, pp. 414–454, 2013.

[3] R. S. Souza, J. L. B. Lopes, P. T. Davet, C. G. Garcia, G. I. Gadotti, A. C. Yamin, and C. F. R. Geyer, "Context awareness in ubicomp: an iot oriented distributed architecture," *Proceedings of 2015 IEEE International Conference on Electronics, Circuits, and Systems*, 2015.

[4] J. Lopes, R. Souza, C. Geyer, C. Costa, J. Barbosa, A. Pernas, and A. Yamin, "A middleware architecture for dynamic adaptation in ubiquitous computing," *Journal of Universal Computer Science*, vol. 20, no. 9, pp. 1327–1351, sep 2014.

[5] F. UPnP, "Fórum upnp," 2015, disponível em :<http://upnp.org//>. Acesso em julho de 2015.

[6] A. Kurniawan, *Sparkfun ESP8266 Thing Development Workshop*. Canada: PE PRESS, 2015.

[7] O. M. Group, "Business Process Model and Notation," 2015, disponível em :<http://www.bpmn.org>. Acesso em maio de 2015. [Online]. Available: http://www.bpmn.org/

[8] P. F. Pires, E. Cavalcante, T. Barros, F. C. Delicato, T. Batista, and B. Costa, "A platform for integrating physical devices in the internet of things," *Proceedings of the 12th IEEE International Conference on Embedded and Ubiquitous Computing*, pp. 234–241, 2014.

[9] M. Vega-Barbas, D. Casado-Mansilla, M. Valero, D. López-de Ipina, J. Bravo, F. Flórez *et al.*, "Smart spaces and smart objects interoperability architecture (s3oia)," *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pp. 725–730, 2012.

[10] LinkSmart, "Middleware for networked embedded systems," 2015, disponível em :<https://linksmart.eu/redmine/projects/linksmart-opensource/wiki>. Acesso em julho de 2015.

[11] Carriots, "Carriots iot application platform," 2015, disponível em :<https://www.carriots.com/>. Acesso em julho de 2015.