UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

Cleber Costa da Fonseca

SISTEMA DE OPERAÇÃO REMOTA E SUPERVISÃO DE
ILUMINAÇÃO PÚBLICA

São Carlos

2013

Cleber Costa da Fonseca

**SISTEMA DE OPERAÇÃO REMOTA E SUPERVISÃO DE
ILUMINAÇÃO PÚBLICA**

Dissertação apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo como parte dos requisitos para a obtenção do Título de Mestre em Ciências, Programa de Engenharia Elétrica¹.

Área de Concentração: Sistemas Dinâmicos

Orientador: Prof. Dr. Dennis Brandão

São Carlos

2013

¹ Trata-se da versão corrigida da dissertação. A versão original se encontra disponível na EESC/USP que aloja o Programa de Pós-Graduação de Engenharia Elétrica.

AUTORIZO A REPRODUÇÃO TOTAL OU PARCIAL DESTE TRABALHO,
POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS
DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

C837s Costa da Fonseca, Cleber
SISTEMA DE OPERAÇÃO REMOTA E SUPERVISÃO DE
ILUMINAÇÃO PÚBLICA / Cleber Costa da Fonseca;
orientador Dennis Brandão. São Carlos, 2013.

Dissertação (Mestrado) - Programa de Pós-Graduação
em Engenharia Elétrica e Área de Concentração em
Sistemas Dinâmicos -- Escola de Engenharia de São
Carlos da Universidade de São Paulo, 2013.

1. Suportabilidade com XML. 2. Operação remota. 3.
Sistema de iluminação pública. 4. Software. 5.
Supervisão. I. Título.

FOLHA DE JULGAMENTO

Candidato: Engenheiro **CLEBER COSTA DA FONSECA**.

Título da dissertação: "Sistema de operação remota e supervisão de iluminação pública".

Data da defesa: 04/03/2013

Comissão Julgadora:

Resultado:

Prof. Dr. **Dennis Brandão (Orientador)**
(Escola de Engenharia de São Carlos/EESC)

Aprovado

Prof. Dr. **Márcio José da Cunha**
(Universidade Federal de Uberlândia/UFU)

APROVADO

Prof. Associado **Evandro Luis Linhari Rodrigues**
(Escola de Engenharia de São Carlos/EESC)

APROVADO

Coordenador do Programa de Pós-Graduação em Engenharia Elétrica e Presidente da
Comissão de Pós-Graduação:
Prof. Titular **Denis Vinicius Coury**

Protestado em
Serviço de Registro
Em 12/03/2013

AGRADECIMENTOS

Agradeço, em primeiro lugar, a Deus pela vida e saúde;

À minha amada família, Vanessa, Igor e Larissa, pelo carinho,
incentivo e paciência;

Aos meus queridos pais, pelo eterno incentivo ao estudo e trabalho;

Ao professor Dr. Dennis Brandão, pela orientação e dedicação;

À Smar, pelo incentivo contínuo à pós-graduação;

À Universidade de São Paulo, por disponibilizar os recursos para o
desenvolvimento deste trabalho;

E aos colegas de pós-graduação pelo auxílio.

RESUMO

FONSECA, C. C. (2013). **Sistema de operação remota e supervisão de iluminação pública**. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2013.

Sistemas de operação remota e supervisão de iluminação pública são compostos por dispositivos acoplados aos pontos de luz interligados via rede, e aplicativos que são executados em computadores que indicam problemas nos pontos de iluminação e apuram o valor do consumo de energia. Estudar as tecnologias dos trabalhos correlatos, propor um sistema dedicado para iluminação pública e implantar o sistema proposto em um teste piloto para avaliação das características de operação e supervisão são os objetivos do trabalho. A arquitetura do sistema proposto é modular e expansível. O modelo baseado em células permite que novos conjuntos de dispositivos possam ser adicionados de acordo com a demanda. No desenvolvimento do trabalho a linguagem C# é adotada para desenvolver a operação e supervisão através do padrão *CyberOPC* (*Cybernetic OPC*) e arquivos do tipo XML são aplicados para descrição dos dispositivos e definição da topologia da rede. Os resultados obtidos em simulação e no teste piloto validam a metodologia e arquitetura proposta.

Palavras-chave: Suportabilidade com XML, Operação remota, Sistema de iluminação pública, Software, Supervisão.

ABSTRACT

FONSECA, C. C. (2013). **System of remote operation and supervision of public street lighting**. M.Sc. Dissertation – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2013.

Systems of remote operation and supervision of public lighting are composed of devices attached to the light points interconnected via a network, and applications that run on computers that indicate problems with lighting points and discharge the amount of energy consumption. Studying the related works technologies, propose a system dedicated to public lighting and deploy the proposed system in a pilot test to evaluate the operating characteristics and supervision are the objectives of the work. The architecture of the proposed system is modular and expandable. The cell-based model allows new sets of devices can be added according to demand. In developing the work the C # language is adopted to develop the operation and monitoring via standard CyberOPC (Cybernetic OPC) and XML file types are applied to the device description and definition of the network topology. The results obtained from simulation and from the test pilot validate the methodology and the proposed architecture.

Keywords: Supportability with XML, Remote operation, Lighting System, Software, Supervision

LISTA DE FIGURAS

Figura 2.1 – Modelo arquitetural por segmento típico	15
Figura 2.2 – Modelo arquitetural por célula típico	16
Figura 2.3 – Distribuição dos controladores de comunicação típico via rede Ethernet	17
Figura 3.1 – Arquitetura do sistema	33
Figura 4.1 - Arquivo de descrição do nó sensor	38
Figura 4.2 – Coleção de parâmetros	40
Figura 4.3 – Atributos do nó <i>Parameter</i>	42
Figura 4.4 – Nós <i>Description</i> do nó <i>Parameter</i>	43
Figura 4.5 – Nó <i>Enumerations</i>	43
Figura 4.6 – Nó <i>Range</i> e <i>Unit</i>	44
Figura 4.7 – Coleção de alarmes	46
Figura 4.8 – Coleção de parâmetros de processo	47
Figura 4.9 – Coleção de parâmetros de operação do sensor.....	48
Figura 4.10 – Nós <i>Description</i> do nó <i>OperationStatus</i>	49
Figura 4.11 – Estados de operação de um parâmetro	49
Figura 4.12 – Coleção de IHM	50
Figura 4.13 – Nós <i>Description</i> do nó <i>Group</i>	51
Figura 4.14 – IHM <i>Battery</i>	51
Figura 4.15 – Diagrama de classe UML - <i>Library</i>	52
Figura 4.16 – Diagrama de classe UML - <i>AssetDescriptor</i>	53
Figura 4.17 – Diagrama de classe UML - <i>Parameter</i>	55
Figura 4.18 – Diagrama de classe UML - <i>UIGroup</i>	56
Figura 4.19 – Criando o componente <i>XMLLibrary</i>	57
Figura 4.20 – Iterando sobre a lista de modelos	58
Figura 4.21 – Arquitetura do servidor <i>CyberOPC</i>	59
Figura 4.22 – Retângulo e os seus vértices enumerados obedecendo à regra para formatação do campo Coordenada da região para leitura em grupo	64
Figura 4.23 – Regra para formatação do campo Coordenada da região para leitura em grupo utilizando o círculo	64
Figura 4.24 – Estrutura do arquivo de topologia	70
Figura 4.25 – Coleção de nós sensores do arquivo de topologia.....	70

Figura 4.26 - Coleção de controladores de comunicação do arquivo de topologia	71
Figura 4.27 – Coleção de células	72
Figura 4.28 – Sensores de uma célula.....	73
Figura 4.29 – Processo de verificação de atualização baseado em requisições HTTPs	75
Figura 5.1 - Conector <i>IPGateway</i> integrado ao servidor <i>CyberOPC</i>	78
Figura 5.2 – Aplicativo de Simulação de nós sensores.....	81
Figura 5.3 – Aplicativo de Simulação, lista dos parâmetros de um nó sensor	82
Figura 5.4 – Aplicativo <i>Null-modem emulator</i> (2012) utilizado para criar a conexão de <i>loopback</i> entre portas seriais.....	83
Figura 5.5 – Simulador recebendo solicitações de serviços e respondendo aos mesmos.....	84
Figura 5.6 – <i>CyberOPC</i> recebendo solicitações de serviços e respondendo aos mesmos.....	84
Figura 5.7 - Aplicativo de configuração e topologia	85
Figura 5.8 – Descrição gráfica de um modelo de sensor	86
Figura 5.9 –Servidor de operação remota	87
Figura 5.10 –Acesso da página de operação do sistema ao Servidor de operação remota	88
Figura 5.11 – Acesso remoto ao Servidor de operação remota ativo no Laboratório de Automação Industrial na USP, São Carlos - SP	89
Figura 5.12 – Acesso via Internet da página de operação do sistema ao Servidor de operação remota instalado na USP.....	90
Figura 5.13 – Navegador de Internet em modo de depuração, indicando as múltiplas requisições do serviço “getAlarms.xml” executadas pela página ao Servidor de operação remota	91
Figura 5.14 – Navegador de Internet em modo de depuração, mostrando detalhes de uma requisição HTTP ao serviço “getAlarms.xml”	92
Figura 5.15 – Navegador de Internet em modo de depuração, mostrando o XML de resposta à solicitação ao serviço “getAlarms.xml”	93
Figura 5.16 – Indicação dos eventos de requisições HTTP recebidos no Servidor de operação remota enviados pela página HTML.....	94
Figura 5.17 – Aplicativo de operação remota.....	95
Figura 5.18 – Vistas de alarmes	96
Figura 5.19 – Filtro de alarmes	97
Figura 5.20 – Histórico de alarmes	98
Figura 5.21 – Menu de operação com o estado de operação das lâmpadas do Aplicativo de operação remota.....	99

Figura 5.22 – Visualização somente das lâmpadas acesas	100
Figura 5.23 – Menu de operação individual do Aplicativo de operação remota.....	101
Figura 5.24 – Menu de operação em grupo do Aplicativo de operação remota.....	102
Figura 5.25 – Operação em grupo com retângulo	103
Figura 5.26 – Operação em grupo com círculo	104
Figura 5.27 – Gráfico de consumo diário do mês de dezembro de 2012	105
Figura 5.28 – Gráfico de consumo mensal do ano de 2012	105
Figura 5.29 – Configuração do experimento para o teste de integração de comunicação entre o sistema e os nós sensores reais	107
Figura 5.30 – Software <i>Wireshark</i> (2012) dispondo das capturas executadas durante a integração.....	108
Figura 5.31 – Integração do sistema de operação remota e supervisão de iluminação pública com dispositivos físicos.....	109
Figura 5.32 – Configurações do simulador para o experimento do comportamento do sistema perante latência e a não entrega de leituras e escritas.....	111
Figura 5.33 – Resumo dos resultados obtidos no teste de partida de uma célula.....	113
Figura 5.34 – Indicação de progresso da supervisão, indicando valores lidos e não lidos.....	115
Figura 5.35 – Configuração do experimento para validação da arquitetura.....	117
Figura 5.36 – Cliente <i>CyberOPC</i> usual disposto no Servidor de operação remota enviando comando de leitura a célula 1	118
Figura 5.37 – Servidor <i>CyberOPC</i> disposto na célula 1 respondendo ao comando de leitura solicitado pelo cliente usual <i>CyberOPC</i> disposto no Servidor de operação remota	119
Figura 5.38 – Cliente <i>CyberOPC</i> usual disposto no Servidor de operação remota enviando comando de leitura a célula 2	119
Figura 5.39 – Servidor <i>CyberOPC</i> disposto na célula 2 respondendo ao comando de leitura solicitado pelo cliente usual <i>CyberOPC</i> disposto no Servidor de operação remota	120
Figura 5.40 – Experimento com duas células apresentado no mapa pelo Aplicativo de operação remota.....	121

LISTA DE TABELAS

Tabela 2.1 – Resumo comparativo entre as técnicas de descrição de dispositivos	26
Tabela 2.2 – Comparação entre Google maps e Bing maps	27
Tabela 4.1 - Atributos do nó principal <i>Sensor</i>	39
Tabela 4.2 – Enumeração do atributo <i>accessRights</i>	41
Tabela 4.3 – Enumeração do atributo <i>type</i>	41
Tabela 4.4 - Equivalência entre os atributos da classe <i>AssetDescriptor</i> e o arquivo de descrição	54
Tabela 4.5 – Equivalência entre os atributos da classe <i>Parameter</i> e o arquivo de descrição...	56
Tabela 4.6 - Equivalência entre os atributos da classe <i>UIGroup</i> e o arquivo de descrição.....	57
Tabela 4.7 – Tipos de serviços do protocolo de comunicação e o seu número de função correspondente.....	61
Tabela 4.8 – Campos do pacote de requisição para leitura individual	62
Tabela 4.9 – Campos do pacote de resposta para leitura individual.....	63
Tabela 4.10 – Campos do pacote de requisição para escrita individual	66
Tabela 4.11 – Campos do pacote de requisição para escrita em grupo	67
Tabela 4.12 – Campos do pacote de notificação de alarme.....	68
Tabela 5.1 – Nível de severidade do alarme com a sua cor correspondente.	96
Tabela 5.2 – Cenários e resultados dos testes executados para obtenção do tempo de partida de uma célula	113
Tabela 5.3 – Desempenho do requisito operação individual perante a latência e taxa de entrega	114

LISTA DE SIGLAS

AJAX	<i>Asynchronous JavaScript and XML</i>
AC	<i>Alternating Current</i>
CO2	<i>Dióxido de carbono</i>
CyberOPC	<i>Cybernetic OPC (padrão OPC desenvolvido para uso sob HTTP)</i>
DD	<i>Device Description</i>
EDD	<i>Electronic Device Description</i>
EEDL	<i>Electronic Device Description Language</i>
FDT / DTM	<i>Field Device Tool / Device Type Manager</i>
GGPSR	<i>Geocast Greedy Perimeter Stateless Routing</i>
GPRS	<i>General Packet Radio Service</i>
GPS	<i>Global Positioning System</i>
GSM	<i>Global System for Mobile Communications</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
HTTPS	<i>Hyper Text Transfer Protocol Secure</i>
IHM	<i>Interface Homem-Máquina</i>
IP	<i>Internet Protocol</i>
JSON	<i>Java Script Object Notation</i>
JSON-RPC	<i>Java Script Object Notation - Remote Procedure Call</i>
OO	<i>Object-Oriented</i>
OPC	<i>Ole for Process Control</i>
OPC DA	<i>OPC Data Access</i>
OPC XML-DA	<i>OPC XML Data Access</i>
Open-DD	<i>Open Device Description</i>
PROCEL RELUZ	<i>Programa Nacional de Iluminação Pública e Sinalização Semafórica Eficiente</i>
RO	<i>Read-Only</i>
RS232	<i>Recommended Standard 232</i>
RW	<i>Read-Write</i>
SGBD-OR	<i>Sistemas de Gerenciadores de Banco de dados - Objetos-Relacionais</i>
SOAP	<i>Simple Object Access Protocol</i>

INPE	Instituto Nacional de Pesquisas Espaciais
UML	<i>Unified Modeling Language</i>
USB	<i>Universal Serial Bus</i>
XSLT	<i>Extensible Style Sheet Language Transformations</i>
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

1	INTRODUÇÃO.....	7
2	TRABALHOS CORRELATOS	11
2.1	MODELOS ARQUITETURAIS	14
2.2	REQUISITOS DOS SISTEMAS DE ILUMINAÇÃO PÚBLICA	18
2.3	TÉCNICAS DE SUPERVISÃO.....	21
2.4	TÉCNICAS DE DESCRIÇÃO DE DISPOSITIVOS.....	22
2.5	GEOREFERENCIAMENTO EM MAPAS NA INTERNET	27
2.6	CONSIDERAÇÕES FINAIS.....	28
3	TRABALHO PROPOSTO	31
3.1	ARQUITETURA.....	31
3.2	REQUISITOS DO APLICATIVO DE OPERAÇÃO REMOTA	33
3.3	ESCOPO	35
4	METODOLOGIA.....	37
4.1	SUPORTABILIDADE.....	37
4.1.1	ESPECIFICAÇÃO DO ARQUIVO DE DESCRIÇÃO	38
4.1.2	COLEÇÃO DE PARÂMETROS.....	39
4.1.3	AGRUPAMENTO DE ALARMES	45
4.1.4	AGRUPAMENTO DOS PARÂMETROS DE PROCESSO	46
4.1.5	AGRUPAMENTO DOS PARÂMETROS DE OPERAÇÃO	47
4.1.6	AGRUPAMENTO DE IHM	49

4.1.7	COMPONENTE DE SOFTWARE <i>XMLLibrary</i>	52
4.2	COMUNICAÇÃO ENTRE O SISTEMA E OS NÓS SENSORES	58
4.2.1	<i>CYBEROPC</i>	59
4.2.2	PROTOCOLO DE COMUNICAÇÃO ENTRE O <i>CYBEROPC</i> E O CONTROLADOR DE COMUNICAÇÃO	60
4.2.3	FORMATAÇÃO DOS PACOTES DE REQUISIÇÃO E RESPOSTA DOS SERVIÇOS DE LEITURA, ESCRITA E NOTIFICAÇÃO DE ALARME.	60
4.3	APLICATIVOS DE GERENCIAMENTO	69
4.3.1	APLICATIVO DE CONFIGURAÇÃO / TOPOLOGIA.....	69
4.3.2	ARQUIVO DE TOPOLOGIA	69
4.3.3	SERVIDOR DE OPERAÇÃO REMOTA	73
4.3.3.1	OPERAÇÃO REMOTA	73
4.3.3.2	MONITORAMENTO	75
5	ENSAIOS E RESULTADOS	77
5.1	RECURSOS DO SISTEMA	77
5.1.1	<i>CYBEROPC</i>	77
5.1.2	SIMULADOR	79
5.1.3	APLICATIVO DE CONFIGURAÇÃO E TOPOLOGIA.....	85
5.1.4	SERVIDOR DE OPERAÇÃO REMOTA	86
5.1.5	APLICATIVO DE OPERAÇÃO REMOTA.....	94
5.1.5.1	ALARME.....	95
5.1.5.2	ESTADO DE OPERAÇÃO	98
5.1.5.3	OPERAÇÃO INDIVIDUAL.....	100

5.1.5.4	OPERAÇÃO EM GRUPO.....	101
5.1.5.5	CONSUMO DE ENERGIA	104
5.1.5.6	SUPORTABILIDADE.....	106
5.2	INTEGRAÇÕES COM O CONTROLADOR DE COMUNICAÇÃO E OS NÓS SENSORES REAIS.....	106
5.3	COMPORTAMENTO DO SISTEMA PERANTE LATÊNCIA E A NÃO ENTREGA DE LEITURAS E ESCRITAS	109
5.3.1	DESEMPENHO DA PARTIDA PARA OBTENÇÃO DO ESTADO INICIAL DE OPERAÇÃO E ALARMES DE UMA CÉLULA	111
5.3.2	DESEMPENHO DE LEITURA E ESCRITA NA OPERAÇÃO INDIVIDUAL.....	114
5.4	VALIDAÇÃO DA ARQUITETURA	116
6	CONSIDERAÇÕES FINAIS E CONCLUSÕES.....	123
6.1	PUBLICAÇÕES.....	124
6.2	TRABALHOS FUTUROS.....	124
	REFERÊNCIAS BIBLIOGRÁFICAS	127

1 Introdução

A resolução 414 de setembro de 2010 da Agência Nacional de Energia Elétrica (Agência Nacional de Energia Elétrica - ANEEL, 2010) estabelece que a pessoa jurídica de direito público é o responsável pela operação e manutenção da iluminação pública, sendo a pessoa jurídica de direito público um dos seguintes poderes públicos federal, estadual ou distrital e municipal. A resolução ainda cita que a pessoa jurídica de direito público pode contratar empresas ou concessionárias de energia elétrica para manter os pontos de iluminação.

As principais atribuições da empresa ou concessionária contratada são estipular o valor de consumo de energia elétrica da iluminação pública mensalmente do município e manter o parque de iluminação pública operando.

A metodologia adotada para cálculo do consumo de energia é baseado em número de pontos de iluminação instalados e quantidade fixa de horas de operação da lâmpada, podendo provocar dúvidas com relação ao valor estipulado do consumo de energia elétrica. A resolução 414 de setembro de 2010 (Agência Nacional de Energia Elétrica - ANEEL, 2010) determina que uma lâmpada permaneça ligada diariamente onze horas e cinquenta e dois minutos para efeitos do cálculo de consumo de energia elétrica na iluminação pública, ressalvado o caso de logradouros que necessitem de iluminação permanente, em que o tempo de operação é de vinte quatro horas por dia. Na resolução, o consumo de energia é referenciado na seção X, artigo 24 e parágrafo 1, da seguinte forma:

O tempo a ser considerado para consumo diário pode ser diferente do estabelecido no *caput*, após estudo realizado pelo consumidor e a distribuidora junto ao Observatório Nacional, devidamente aprovado pela ANEEL.

Ainda, na seção no artigo 26 da resolução 414, cita-se também o consumo de energia:

Caso sejam instalados equipamentos automáticos de controle de carga que reduzam o consumo de energia elétrica do sistema de iluminação pública, devidamente comprovado e reconhecido por órgão oficial e competente, a distribuidora deve proceder à revisão da estimativa de consumo e considerar a redução proporcionada por tais equipamentos.

Atualmente para manter os ativos de iluminação em perfeito funcionamento, a notificação de problemas relacionados aos pontos de iluminação pública são de responsabilidade do usuário do sistema. As concessionárias são obrigadas a disponibilizar plantões por telefone vinte e quatro horas por dia e sete dias por semana para atender reclamações dos usuários. No contrato entre o poder público e a concessionária, multas podem ser estabelecidas, caso a reclamação não seja atendida dentro de um prazo estipulado. O pronto atendimento para resolução de problemas de iluminação pública, como exemplo substituição de lâmpadas defeituosas, esta relacionada diretamente com a segurança, pontos não iluminados facilitam a execução de atos ilícitos por parte de criminosos, assim a substituição deve ser providenciada o mais rápido possível pela concessionária. Outro problema na iluminação pública são as lâmpadas que permanecem acesas durante todo o dia desperdiçando energia elétrica.

O bom funcionamento da iluminação pública ainda traz os seguintes benefícios: favorece o desenvolvimento econômico e social (turismo e lazer), proporciona maior segurança no trânsito e promove a identidade visual do município.

A proposta deste trabalho é propor e implementar um sistema de operação remota e supervisão de iluminação pública. Sistemas de operação remota e supervisão de iluminação pública são compostos por dispositivos acoplados aos pontos de luz interligados via rede, e aplicativos que são executados em computadores que indicam problemas nos pontos de

iluminação (sem a dependência do relato do usuário) e apuram o valor do consumo de energia (utilizando medidores).

Os municípios brasileiros procuram alternativas eficientes para operar e manter a iluminação pública, pois a partir de 31 de janeiro de 2014 todos os ativos da iluminação serão transferidos das concessionárias para os municípios, esta nova política é definida na resolução normativa 414 da ANEEL. Um estudo preliminar da Confederação Nacional dos Municípios (Confederação Nacional dos Municípios – CNM, 2012) estima uma média de acréscimo de 28% a despesa anual dos municípios.

No Brasil, observa-se a aplicação destes tipos de sistemas para gestão da iluminação pública. Na cidade de Aparecida do Norte, São Paulo, é previsto a instalação cerca de 15.000 medidores inteligentes por toda a cidade, com o objetivo de evitar erros de medições e identificar falta de energia, almejando-se a redução com custos de manutenção. Outro caso é da cidade de Búzios situada no Rio de Janeiro, que prevê a instalação de medidores inteligentes e sistemas automatizados, testes iniciais indicam uma economia de 30 a 40% no consumo de energia do município após a instalação do sistema.

Os municípios que possuem interesse em incluir projetos de iluminação inteligente são incentivados pelo Programa PROCEL RELUZ (2000), que financia até 75% do valor total do projeto. O sistema proposto é um projeto elegível pelo Programa PROCEL RELUZ (2000), categorizado como “Inovação tecnológica na iluminação pública”. Assim, estabelecem-se os objetivos do sistema:

- Estudar as tecnologias dos trabalhos correlatos a fim de definir a arquitetura do sistema de operação remota e supervisão de iluminação para grandes áreas e os seus requisitos;

- Propor o sistema de operação remota e supervisão de iluminação pública, com possibilidade de integração de dispositivos com características distintas. A integração é uma contribuição do trabalho;
- Implantar o sistema proposto em um teste piloto para avaliação das características de medição de consumo de energia e recepção de alarmes.

2 Trabalhos correlatos

Nesta seção são apresentados os estudos dos trabalhos correlatos, estudando as tecnologias aplicadas a fim de definir a arquitetura e os requisitos do sistema de operação remota e supervisão de iluminação pública.

Os autores Yao et al. (2006) apresentam uma solução, composta por Aplicativo de gerenciamento central, um controlador principal e nós sensores. O Aplicativo de gerenciamento central baseia-se nas coordenadas geodésicas e parâmetros de supervisão/controle dos nós sensores que são armazenados em banco de dados. O controlador central é instalado na unidade de transformação e agrupa os nós sensores do segmento, responsável por receber e retransmitir as solicitações do aplicativo para os nós sensores e notificar eventos de falhas dos nós sensores ao Aplicativo. O aplicativo interage com o controlador trocando mensagens pela rede GSM (*Global System for Mobile Communications*).

Um sistema de operação e supervisão em redes sem fio de pontos de iluminação em estradas é proposto pelos autores Long et al. (2009). A arquitetura é subdividida em um centro de controle e por zonas de redes sem fio. O centro de controle é composto por um computador e um modem GPRS (*General packet radio service*) que se comunicam pelo padrão RS232 (*Recommended Standard 232*). Cada zona é constituída por um módulo de comunicação e pelos sensores sem fios. O módulo de comunicação comunica-se com o centro de controle via Internet - GPRS, responsável por direcionar pedidos de leitura/escrita aos pontos de iluminação e notificar alarmes ocorridos nos sensores sem fios ao centro de controle.

A arquitetura desenvolvida pelos autores Chunguo et al. (2007) é composta por três elementos o centro de controle, a unidade de terminal remota e os sensores sem fios. No centro de controle, o aplicativo instalado no computador é responsável por ilustrar a topologia do sistema, posicionar os sensores em mapas, supervisionar as informações relevantes das lâmpadas, configurar o agendamento da iluminação, enviar comandos para a unidade de terminal remota, prover efeitos audiovisuais para alarmes ocorridos, transmitir mensagens de texto para celulares, consultar dados descritivos dos sensores armazenados em banco de dados e produz relatórios de alarmes ocorridos.

A unidade de terminal remota serve como ponto concentrador dos sensores, e é instalada na estação de transformação. As funções da unidade de terminal remota são comunicar-se com o centro de controle via GPRS, processar solicitações de leitura/escrita do centro de controle e coletar dados de consumo/mudança de estado de operação das lâmpadas e enviar ao centro de controle. Os sensores sem fio são responsáveis pela supervisão/controle do ponto de iluminação, podendo ligar, desligar ou controlar a luminosidade da lâmpada mediante solicitação da unidade de terminal remota.

O trabalho correlato dos autores Chen e Liu (2009) é composto por um computador central, estação central de supervisão e terminal de supervisão da lâmpada. O computador central se conecta a rede do sistema através de um equipamento acoplado pelos padrões RS232/485 ou USB (*Universal serial bus*), onde tal equipamento comunica-se via rede sem fio com a estação central de supervisão.

A estação central de supervisão tem como atribuição aplicar a comunicação entre o computador central e o terminal de supervisão das lâmpadas que estão conectados via rede cabeada, os alarmes provenientes dos terminais de supervisão devem ser transmitidos ao computador central e as requisições de supervisão/controle solicitadas pelo computador central são direcionadas ao terminal de supervisão das lâmpadas.

As principais funções do terminal de supervisão da lâmpada são: controle de ligar e desligar a lâmpada, coletar o estado de operação da lâmpada em tempo real e comunicar-se com estação central de supervisão.

A arquitetura proposta pelos autores Iordache et al. (2008) é composta pelo computador, variador AC (*Alternating current*) conectado a um modem GPRS e os pontos de iluminação. O computador conectado a Internet estabelece a comunicação via GPRS com o modem do variador AC. O variador AC agrupa e regula a energia para certo número de lâmpadas, sensores de presença e luminosidade são acoplados ao variador AC.

O sensor de presença indica ao variador AC a presença de pedestres ou carros permitindo racionalizar a energia elétrica e o sensor de luminosidade permite ao variador AC controlar a energia que será aplicada aos pontos de iluminação em função da luz ambiente.

O sistema proposto pelos autores Mendalka et al. (2010) é composto pelos seguintes componentes: aplicativo de gerenciamento central, controlador de comunicação e nós sensores. O aplicativo proposto caracteriza-se por controle individual ou por grupo de lâmpadas, indica o estado de operação e localização das lâmpadas em mapas, armazenamento de variáveis (temperatura, ruído, umidade, luminosidade e intensidade do tráfego), armazenamento de alarmes provenientes dos nós sensores e coordenar a comunicação com vários controladores de comunicação.

O controlador de comunicação controla um grupo de nós sensores via rede sem fio e comunica-se via Internet com o aplicativo gerenciador, responsável por receber anomalias/alarmes detectadas pelos nós sensores e transmiti-las ao aplicativo, além de coordenar solicitações de supervisão/controle do aplicativo gerenciador direcionada aos nós sensores.

Hong et al. (2011) apresentam o desenvolvimento de uma solução para iluminação em estradas por meio de sensores instalados nos pontos de iluminação e transmitem informações a um gateway que coordenada a comunicação entre a sala de controle e os sensores. Os sensores

são multifunção capazes de realizar as seguintes medições de CO₂ (Dióxido de carbono), luminosidade do ambiente e umidade.

O estudo de trabalhos correlatos permite estabelecer, como um sistema de operação e supervisão de iluminação pública é constituído em termos arquiteturais, requisitos de supervisão/operação e a integração de dispositivos.

2.1 Modelos arquiteturais

Definem-se modelos arquiteturais como a interconexão dos computadores, redes de comunicação, sensores e computadores para prover a supervisão e operação de um sistema de iluminação pública. Nesta seção são apresentados os modelos arquiteturais estudados na literatura.

Nos trabalhos de Atici et al. (2011), Yao et al. (2006), Long et al. (2009), Chunguo et al. (2007), Chen e Liu (2009), Iordache et al. (2008) e Hong et al. (2011), nota-se um padrão no que se refere a concepção da arquitetura. Nós sensores são agrupados por controladores de comunicação, que formam uma célula, várias células são formadas para que geograficamente uma área possa ser monitorada/controlada. O controlador de comunicação coordena a comunicação entre os nós sensores da célula com um aplicativo de gerenciamento.

A célula é composta por nós sensores interligados em rede e são agrupados seguindo dois modelos, um por segmento e outro por região. O modelo por segmento é empregado quando o meio físico de comunicação entre os nós sensores e o controlador de comunicação é cabeado, tipicamente o controlador de comunicação é instalado na unidade de transformação. Exemplos deste modelo são ilustrados pelos autores Yao et al. (2006), Atici et al. (2011), Chunguo et al. (2007), Chen e Liu (2009), Iordache et al. (2008) e Hong et al. (2011). A Figura 2.1 ilustra o modelo arquitetural por segmento típico. Um segmento é formado pela composição da rede cabeada, nós sensores e o controlador de comunicação. Várias redes podem ser conectadas ao computador, que prove a supervisão e operação do sistema.

Figura 2.1 – Modelo arquitetural por segmento típico

O modelo por região é adotado quando a comunicação entre o controlador de comunicação e os nós sensores é via rede sem fio. Um controlador de comunicação é posicionado para que o maior número de nós sensores seja coberto. Exemplos deste modelo são ilustrados pelos autores Long et al. (2009) e Mendalka et al. (2010).

O controlador de comunicação é o coordenador de comunicação entre os nós sensores e o aplicativo de gerenciamento. Anomalias ocorridas nos nós sensores são informadas ao aplicativo de gerenciamento e solicitações de leitura/escrita por parte do aplicativo devem ser direcionadas aos nós sensores. Tais características são comuns entre os trabalhos Yao et al. (2006), Atici et al. (2011), Long et al. (2009), Chunguo et al. (2007), Chen e Liu (2009), Iordache et al. (2008), Mendalka et al. (2010) e Hong et al. (2011).

A Figura 2.2 mostra o modelo arquitetural por célula típico, onde uma célula é composta pela rede de sensores sem fio, os nós sensores e o controlador de comunicação. Várias células podem ser formadas e cada célula é conectada ao computador para a cobertura plena de um município.

Figura 2.2 – Modelo arquitetural por célula típico

A principal função do aplicativo é disponibilizar a supervisão, operação e recebimento de alarmes dos nós sensores, como mencionam os autores Yao et al. (2006), Atici et al. (2011),

Long et al. (2009), Chunguo et al. (2007), Chen e Liu (2009), Iordache et al. (2008), Mendalka et al. (2010) e Hong et al. (2011) com exceção de Atici et al. (2011), Iordache et al. (2008) e Hong et al. (2011) não citam a aplicação de alarmes nas soluções.

Os trabalhos dos autores Atici et al. (2011), Yao et al. (2006), Long et al. (2009), Chunguo et al. (2007), Chen e Liu (2009), Iordache et al. (2008), Mendalka et al. (2010) e Hong et al. (2011) apresentam soluções distribuídas, onde o controlador de comunicação e o aplicativo de gerenciamento se comunicam remotamente através da rede Ethernet. A Figura 2.3 apresenta a solução típica encontrada na literatura para a distribuição dos controladores de comunicação via rede Ethernet. A distribuição dos controladores via rede Ethernet não limita as soluções, pois os controladores de comunicação não precisam estar conectados fisicamente ao computador.

Figura 2.3 – Distribuição dos controladores de comunicação típico via rede Ethernet

O modelo por região é o aplicado neste trabalho, pois trata-se de uma rede de sensores sem fio para iluminação pública. Este modelo possibilita a distribuição dos nós sensores por controladores de comunicação, aplicando o critério de proximidade geográfica. Cada controlador de comunicação comanda um determinado número de nós sensores, formando uma célula. Várias células são formadas para a cobertura geográfica de uma determinada área.

A manutenção da topologia se faz necessária, pois, as áreas podem sofrer alterações como expansão, remoção e alteração. Um exemplo típico é a criação de um novo bairro a um centro urbano, assim a nova célula deve ser adicionada a topologia.

2.2 Requisitos dos sistemas de iluminação pública

Na literatura e nas soluções de mercado percebe-se que existe um padrão das funcionalidades necessárias para gestão de um sistema de iluminação pública. De forma geral, Denardin et al. (2009) afirmam que um sistema de iluminação pública deve ser capaz de executar tarefas básicas como ligar, desligar, controlar luminosidade (quando possível), medir consumo e controlar o tempo de vida útil da lâmpada. A partir da análise dos trabalhos correlatos pode-se dividir os requisitos em:

Supervisão: Lee et al. (2006) ilustram um exemplo bem típico de como um aplicativo de gerenciamento pode disponibilizar as informações do estado das lâmpadas em tempo real, através de uma perspectiva gráfica onde o usuário pode facilmente identificar os estados de operação de um trecho da rede.

O aplicativo da empresa *Streetlight.Vision*, por exemplo, monitora e antecipa o fim da vida útil de uma lâmpada supervisionando seu estado. Denardin et al. (2009), Leccese e Leonowicz (2012) e Long et al. (2009) afirmam que além de supervisionar os estados de operação dos pontos de iluminação em tempo real, o sistema dever ser capaz de monitorar variáveis de consumo das lâmpadas.

Operação: Operações como ligar, desligar e controlar luminosidade das lâmpadas são recursos presentes nos aplicativos das empresas *Central Software - Streetlight Intelligence Inc*, *EpiSensor*, *Strategic Telemetry* e *Streetlight. Vision*. Tais operações são efetuadas individualmente, ou seja, um nó sensor é selecionado e o comando é enviado somente àquele dispositivo. A empresa *Streetlight.Vision*, Leccese e Leonowicz (2012) e Long et al. (2009) apresentam que o sistema de iluminação pública pode de acordo com a necessidade de o usuário

estabelecer controle unitário ou em grupo dos nós sensores, um determinado comando é aplicado a vários nós sensores simultaneamente, possibilitando uma maior produtividade para o operador do sistema.

Como exemplo, o usuário deseja alterar o nível de luminosidade das lâmpadas de um município, a aplicação individual não é uma solução prática, pois o usuário deverá selecionar o nó sensor alterar o nível de luminosidade e submeter o novo valor, isso até completar todos os pontos de iluminação do município, enquanto que o controle em grupo permite ao operador a seleção de múltiplos nós sensores para a aplicação do novo valor de luminosidade unicamente.

Alarmes: Long et al. (2009) afirmam que o aplicativo de gerenciamento do sistema instalado no centro de controle deve ser capaz de suportar alarmes para condições anormais de operação.

A empresa *Streetlight.Vision* possui o recurso de manipulação de alarmes. Os alarmes dos dispositivos são armazenados em banco de dados. Esta proposta apresenta uma limitação, pois não existe mecanismo de notificação ao usuário, então relatórios ou telas de alarmes devem ser consultados para averiguar a ocorrência de um alarme.

Operação à Distância: Chunguo et al. possuem como limitação em sua solução o uso de controles ActiveX em navegadores de Internet. ActiveX é uma tecnologia proprietária da empresa Microsoft, que permite ao seu navegador, a Internet Explorer, mostrar e disponibilizar conteúdos em páginas do tipo HTML. Assim, a solução de Chunguo et al. limita-se a apenas ao navegador Internet Explorer.

Os aplicativos das empresas *Central Software - Streetlight Intelligence Inc* e *Streetlight.Vision* disponibilizam o controle completo do sistema via Internet, permitindo somente usuários autorizados operar o sistema remotamente. Já na literatura, Zotos et al. (2012) apresentam a concepção de gerenciamento e monitoramento do sistema via Internet.

Redes de iluminação pública de grandes centros consumidores são complexas e possuem um grande número de dispositivos acoplados, para um melhor gerenciamento e fácil localização dos dispositivos, Chunguo et al. (2007) e as empresas *Central Software - Streetlight Intelligence Inc* (2008), *EpiSensor* (2009) e *Streetlight.Vision* (2009) reconhecem que postes de iluminação devem ser georeferenciados em mapas.

Suportabilidade: As funções disponíveis em um nó sensor são distintas entre fabricantes, para que um aplicativo de gerenciamento possa operar plenamente e disponibilizar as funcionalidades dos dispositivos ao usuário, é necessário que o fabricante descreva as funcionalidades dentro de um padrão comum.

A integração de novos modelos de nós sensores possibilita o aplicativo de gerenciamento a supervisionar/controlar nós sensores com diferentes funcionalidades. Hong et al. (2011) emprega na sua solução nós sensores multifunção capazes de realizar as medições como: nível de dióxido de carbono presente no ambiente, a luminosidade do ambiente e umidade. Fabricantes de nós sensores podem desenvolver seus equipamentos e integrá-los na solução sem que haja a necessidade de modificações adicionais na codificação do aplicativo de gerenciamento.

Uma função não citada em trabalhos correlatos é a capacidade de se agrupar as variáveis dos nós sensores por função. O consumo de uma lâmpada pode ser classificado como uma variável de processo, enquanto variáveis que indicam as condições de operação da lâmpada são agrupadas como alarme. Este tipo de abordagem permite que o aplicativo de gerenciamento seja capaz de separar alarmes e valores de processos em vistas distintas.

Outra capacidade não relatada é o suporte a múltiplos idiomas, dispor as funcionalidades do sensor em várias linguagens possibilita aos operadores selecionar o idioma nativo ou o mais conveniente.

Nos Estados Unidos da América, a organização NIST (*The National Institute of Standards and Technology*, 2012) estabelece padrões para o setor elétrico. Os dispositivos podem ser

descritos e integrados aos sistemas de operação do setor. Ainda a organização estabelece padrões de comunicação entre dispositivos e sistemas. A organização cita a necessidade da padronização, pois existem milhares de dispositivos no mercado americano e os mesmos devem ser capazes de serem integrados aos vários sistemas existentes.

2.3 Técnicas de supervisão

Sistemas de operação e supervisão de iluminação pública devem adotar padrões para disponibilizar a supervisão. O padrão permite a expansibilidade do sistema, assim novos aplicativos podem ser integrados, sem que haja a necessidade do entendimento do protocolo de comunicação por parte do aplicativo.

Trabalhos correlatos em sua maioria não aplicam padrões para supervisão. Porém, os autores Atici et al. (2011) apresentam uma arquitetura centralizada no servidor OPC-DA (*OPC Data Access*), múltiplos segmentos de nós sensores comunicam-se com o servidor OPC-DA, dispondo as suas variáveis.

O servidor OPC-DA facilita a manutenção de supervisão dos nós sensores, funções como adição, remoção e edição são centralizadas, como exemplo, ao se adicionar novos nós sensores a uma célula já existente, as variáveis são mantidas no servidor e automaticamente os novos nós sensores estarão disponíveis ao aplicativo de gerenciamento para supervisão, sem que nenhuma mudança seja necessária no aplicativo.

Outra vantagem na adoção de um servidor OPC-DA é a capacidade de expansão do sistema, múltiplos servidores podem ser gerenciados por um único aplicativo de gerenciamento, centralizando assim a gestão do sistema. A padronização na comunicação entre o aplicativo de gerenciamento e o controlador de comunicação permite uma arquitetura: aberta, distribuída, de fácil manutenção e expansível.

As informações trocadas entre o aplicativo e o OPC-DA não são textos, formato binário, o que limita a aplicação na Internet, *firewalls* restringem o tráfego deste tipo de informação,

limitando a operação remota do sistema de operação e supervisão. Os padrões OPC XML-DA (OPC XML *Data Access*) e o *CyberOPC* foram concebidos para troca de dados remotos pela Internet.

O autor TORRISI (2011) apresenta um estudo comparativo entre OPC XML-DA e o *CyberOPC*, e como conclusão apresenta que o *CyberOPC* provê serviços similares ao OPC XML-DA de leitura e escrita, e desempenho superior na troca de dados via Internet. O *CyberOPC* é uma alternativa viável e eficaz para leitura e escrita de variáveis em sistemas que requerem a operação remota.

2.4 Técnicas de descrição de dispositivos

Padrões de descrição de dispositivos permitem ao fabricante do dispositivo descrever todas as funcionalidades do seu dispositivo. Uma vez descrito, o dispositivo poderá ser integrado de forma plena, sem perda de funcionalidades, a qualquer software que adote aquele padrão de descrição.

Shu et al. (2009) apresentam a utilização de arquivos do tipo XML (*eXtensible Markup Language*) para descrição de dispositivos de diferentes funções e fabricantes em redes de controle para a indústria na China. O arquivo do tipo XML representa um dispositivo, e possui as seguintes informações: data de criação/revisão do arquivo e dados de identificação/parâmetros/blocos do dispositivo. O objetivo é integrar em uma mesma rede digital dispositivos de diferentes fabricantes e possibilitar a troca de dispositivos de mesma função.

Os autores Mori et al. (2008) mostram uma solução residencial integrada para múltiplos dispositivos em rede que podem ser operados por computadores ou dispositivos portáteis. Os autores propõem a utilização de arquivos do tipo XML para a descrição dos múltiplos modelos de dispositivos da residência, como exemplo ilustra um controlador de lâmpada que possui as seguintes funções liga, desliga, controla luminosidade e indica o estado corrente de operação. Tais funções são listadas no arquivo XML correspondentes do controlador de lâmpada.

Jiang et al. (2010) afirmam que o principal problema de sistemas de iluminação pública é a incompatibilidade e interoperabilidade entre dispositivos de diferentes fabricantes e propõe a descrição do equipamento como solução para tal problema. O padrão XML de arquivo é adotado para a descrição dos equipamentos, pois se caracteriza por: independência de plataforma, escalabilidade, fácil interpretação e legibilidade.

O arquivo de descrição é estruturado em dois nós principais: informações do arquivo e Descrição do equipamento. Criação do arquivo e número da revisão são as informações contidas no nó Informações do arquivo. O nó Descrição do equipamento armazena informações básicas do equipamento (Fabricante, Tipo, Modelo e Identificação), funções específicas e descrição dos possíveis estados de operação.

A EDDL (2011) é o outro padrão utilizado para descrever as funcionalidades de dispositivos, é uma linguagem baseada em texto para descrição de características de dispositivo, o texto é compilado e arquivos do tipo DD (*Device Description*) são gerados. A EDDL apresenta as seguintes características: portabilidade, consistência, agrupamentos de parâmetros por classe, controle de revisão e suporte a um padrão internacional (IEC 61804-2), desenvolvedor de dispositivos necessita de um compilador proprietário para geração da DD e o desenvolvedor de aplicativos necessita de um interpretador proprietário para a leitura da DD.

Enhanced EDDL é nova versão do padrão EDDL que possibilita o fabricante descrever elementos gráficos para o dispositivo, como gráficos e janelas.

Outro padrão muito comum para acesso aos recursos dos equipamentos é o FDT / DTM (*Field Device Tool / Device Type Manager*). A tecnologia padroniza a interface de comunicação e configuração entre os dispositivos e os sistemas. O FDT é o sistema, que prove um ambiente comum para acessar os recursos dos dispositivos, possibilitando a configuração, operação e manutenção através da interface padronizada de software, independente do fabricante ou tipo de protocolo de comunicação adotado. O DTM é a representação do dispositivo, e o mesmo é

desenvolvido pelo fabricante do equipamento, que deve prover a interface software padronizado para o mesmo ser integrado em qualquer sistema do tipo FDT.

Na tecnologia FDT/DTM a interface gráfica é provida pelo próprio fabricante do equipamento, assim o dispositivo irá possuir uma visualização uniforme independente do FDT que estiver o acessando. Enquanto que na tecnologia *Enhanced EDDL*, elementos gráficos são descritos pelo fabricante e interpretados pelo sistema, a mesma descrição pode apresentar distintas visualizações dependendo do sistema.

Porém, a tecnologia *Enhanced EDDL* permite uma maior flexibilidade aos desenvolvedores de sistemas, por ser uma tecnologia descritiva, é possível programar sistemas especialistas, interpretando classes de variáveis ou variáveis específicas da EDD. Um exemplo é a possibilidade do desenvolvimento de um sistema que faça a leitura de todos os parâmetros de diagnósticos dos dispositivos e após a leitura os armazene num banco de dados. A tecnologia FDT/DTM se limita neste requisito.

Pantoni e Brandão (2009) propõem um novo padrão chamado *Open-DD (Open Device description)*, para a descrição de instrumentos industriais, baseado em XML e outras tecnologias como XSLT (*Extensible Style sheet Language Transformations*), OO (*Object-oriented*) e XML Schema, é uma alternativa aberta e não proprietária para o modelo adotado hoje na indústria, a EDD (*Electronic Device Description*).

O novo padrão consiste em descrever as funcionalidades dos instrumentos industriais em arquivos XMLs, com o objetivo que aplicativos de configuração e IHM (Interface homem-máquina) possam operar e manipular os dispositivos. Informações como fabricante, tipo, revisão, blocos e parâmetros do instrumento, são descritos no arquivo XML e apresentadas num formato amigável ao usuário, aplicando-se o padrão HTML (*Hypertext Markup Language*).

Para descrever um dispositivo empregando a tecnologia EDDL é necessário que o fabricante do dispositivo possua habilidades de programação e investimentos são necessários

para aquisição do compilador de EDDL para a geração do arquivo de DD. Em contra partida a descrição em arquivo XML não requer habilidades de programação e muito menos investimentos são necessários na aquisição de compiladores por parte do fabricante do dispositivo. Para criar um arquivo do tipo XML é necessário apenas um editor de texto.

O desenvolvedor de aplicativos necessita de um interpretador proprietário para a leitura da DD, este interpretador requer investimento para a aquisição, enquanto que o arquivo do tipo XML é fácil de ser lido e manipulado por software.

No arquivo do tipo XML é ainda possível incorporar as características da EDDL como portabilidade, consistência, agrupamentos de parâmetros por classe e controle de revisão. A adoção do arquivo do tipo XML é uma alternativa viável e de custo efetivo para a descrição de dispositivos.

Padrão	Vantagens	Desvantagens	Aplicação
EDD	<ul style="list-style-type: none"> • Maior flexibilidade aos desenvolvedores de aplicativos para desenvolvimento de ferramentas especialistas. • Portabilidade, consistência, agrupamentos de parâmetros por classe e controle de revisão. 	<ul style="list-style-type: none"> • Apresenta distintas visualizações dependendo do aplicativo. • Necessita habilidades de programação. • Necessária aquisição de compilador de EDDL (proprietário). 	Industrial
FDT/DTM	<ul style="list-style-type: none"> • Visualização uniforme da UI (desde que sejam seguidas as orientações de estilos). 	<ul style="list-style-type: none"> • Limita os desenvolvedores de aplicativos para desenvolvimento de ferramentas especialistas. 	Industrial
XML	<ul style="list-style-type: none"> • Não requer habilidades de programação. • Sem necessidade de investimentos para aquisição de interpretadores. • Portabilidade, consistência, agrupamentos de parâmetros por classe e controle de revisão. 	<ul style="list-style-type: none"> • Dificuldades de interpretação quando arquivos complexos. 	Industrial Residencial Semáforos

Tabela 2.1 – Resumo comparativo entre as técnicas de descrição de dispositivos

2.5 Georeferenciamento em mapas na Internet

Georeferenciar os pontos de iluminação em mapas, facilita a visualização gráfica e espacial de todas as falhas/estado de operação dos pontos de iluminação de um município. Serviços de mapas são disponibilizados na Internet para uso público. Os mais conhecidos e relevantes são Google maps (2010), Yahoo! Maps (2011) e Bing Maps (2011).

O Yahoo! Maps (2011) apresenta uma grande desvantagem com relação a todos os outros serviços, pois somente um tipo de ícone é disponível para georeferenciar a coordenada geodésica no mapa. Essa limitação não permite distinguir, por exemplo, os estados de operação dos pontos de iluminação em um mapa.

A Tabela 2.2 apresenta a comparação das tecnologias de desenvolvimento para Internet entre o Google maps e Bing Maps. O Google maps só não suporta a linguagem de programação C# e em contrapartida o Bing Maps não suporta XSLT, JSON (*Java Script Object Notation*) e XML. Assim, O Google maps é uma alternativa viável e abrangente para se programar o georeferenciamento de pontos de iluminação em mapas na Internet. O Google maps é aplicado na solução da empresa *EpiSensor* que integra recursos de georeferenciamento conectados a Internet.

Tecnologia de desenvolvimento para Internet	Google maps	Bing Maps
Javascript	Suporta	Suporta
XMLHttpRequest	Suporta	Suporta
XSLT	Suporta	Não suporta
JSON	Suporta	Não suporta
XML	Suporta	Não suporta
C#	Não Suporta	Suporta

Tabela 2.2 – Comparação entre Google maps e Bing maps

Opcionalmente para extrair dados geográficos armazenados em banco de dados objetos-relacionais (SGBD-OR), pode se aplicar a biblioteca *TerraLib* (2010) mantida pelo Instituto Nacional de Pesquisas Espaciais (INPE) que é um recurso de software que facilita a manipulação e abstração dos dados geográficos em SGBD-OR.

A *TerraLib* possui uma parte central ou *kernel* que possibilita a decodificação de dados geográficos em formatos abertos através de rotinas que facilitam o desenvolvimento de protótipos para validação de novas técnicas.

2.6 Considerações finais

O objetivo da leitura e análise de trabalhos correlatos é estabelecer e definir a arquitetura para prover a comunicação para supervisão e controle entre um software disposto em um computador e o sistema de iluminação pública, para isso diversas soluções são apresentadas na literatura e no mercado.

Em muitos trabalhos correlatos a comunicação entre os nós sensores e centro de controle é estabelecida pela rede de celulares, aplicando o padrão GPRS para a troca de dados. O grande problema de se aplicar este tipo de comunicação é a cobertura plena de um centro urbano, a rede de celulares é limitada a uma determinada região. Outro ponto negativo é a tarifação por transferência de dados, tornando a operação do sistema oneroso.

Adotando-se padrões na concepção de um software permite a colaboração e expansibilidade da solução. Apenas uma solução estudada apresentou o uso de tecnologias abertas e não proprietárias. Atici et al. (2011) apresentam uma arquitetura centralizada no servidor OPC-DA - centralizando a comunicação é possível expandir a rede sem que sejam necessárias mudanças no aplicativo de gerenciamento.

O *CyberOPC* é um projeto de pesquisa acadêmico que se propõe a ser um sistema de comunicação aberto, baseado em HTTP (*Hyper Text Transfer Protocol*), especialmente

desenvolvido para controlar e supervisionar sistemas industriais remotamente através de redes IP (*Internet Protocol*) públicas (Torrise, 2011).

As técnicas de descrição de dispositivos são limitadas em trabalhos correlatos e soluções de mercado, por serem soluções proprietárias onde apenas um dispositivo pode ser integrado ao sistema. Na indústria é possível integrar em um único software vários dispositivos que medem ou controlam variáveis de natureza distintas, para resolver tal problema à indústria adotou os padrões EDDL e o FDT/DTM.

O principal objetivo ao se adotar um padrão de descrição é a possibilidade de integração transparente dos equipamentos no software. O fabricante descreve as funcionalidades do seu dispositivo e o sistema disponibiliza a operação do dispositivo com todas as funcionalidades descritas pelo fabricante, caracterizando o software como uma solução não proprietária e expansível.

3 Trabalho proposto

O estudo dos trabalhos correlatos permite propor uma arquitetura de sistema e também os requisitos necessários para prover a gestão do sistema de iluminação pública.

A arquitetura proposta é modular, pois o modelo baseado em células permite que novos conjuntos de dispositivos possam ser adicionados de acordo com a demanda e ao sistema operar efetiva e eficientemente independentemente da escala de nós sensores.

Os requisitos propostos são apresentados em função do estudo de soluções de mercado e da literatura. Como característica, os requisitos apresentados adotam padrões na sua concepção, o que torna o sistema aberto e expansível para adição de novas características, permitindo futuras contribuições ao trabalho.

3.1 Arquitetura

A arquitetura proposta neste trabalho baseia-se no modelo de células, aplicando o modelo arquitetural por região, já o modelo arquitetural por segmento não é aplicável, pois o trabalho adota rede de sensores sem fio. Os estudos de modelos arquiteturais apresentam a possibilidade de distribuir os controladores de comunicação via rede Ethernet, o que permite a distribuição dos controladores de comunicação geograficamente em um município.

Na arquitetura proposta, a célula tem como função prover o acesso de supervisão e escrita para um determinado número de nós sensores, ainda ser acessada remotamente via rede Ethernet pelos aplicativos de gerenciamento, permitindo a distribuição física das células. Porém, os controladores de comunicação não estão acoplados diretamente a rede Ethernet e sim ao computador que permite a comunicação remota.

Uma célula é composta por uma cabine, um computador industrial, uma antena GPS, um controlador de comunicação e nós sensores. Inúmeras células podem ser formadas com o objetivo da cobertura plena de um centro urbano.

O computador industrial é instalado na cabine e é conectado a um controlador de comunicação, que se comunicam utilizando um protocolo serial, enquanto que o controlador de comunicação comunica-se com os nós sensores pela rede sem fios.

O controlador de comunicação ainda possui a capacidade de ler periodicamente o horário do *Global Positioning System* (GPS), com o objetivo de se manter uma única base de tempo entre as várias células, assim alarmes ocorridos em diferentes células podem ser ordenados. Os aplicativos de gerenciamento (Configuração/Topologia, Servidor de operação remota e monitoração) são instalados no computador de Operação e comunicam-se via rede Ethernet com as células.

O acesso remoto é feito pelo navegador de Internet do computador Aplicativo de operação remota ao servidor de operação remota. A biblioteca de mapas do Google (Google maps) é adotada para plotar os pontos de iluminação em mapas.

A Figura 3.1 apresenta a arquitetura do sistema e ilustra a célula 01 responsável por agrupar alguns sensores, ainda mostra a capacidade de adição de novas células até que todos os pontos de iluminação de uma cidade sejam mapeados.

Figura 3.1 – Arquitetura do sistema

3.2 Requisitos do Aplicativo de operação remota

Nesta seção serão listados todos os requisitos propostos para a operação do sistema de iluminação pública. A seguir são listados os requisitos:

- 1) *Operação remota:* O usuário deve operar o sistema através de um navegador de Internet, nomeado como Aplicativo de operação remota;
- 2) *Alarmes:* Os nós sensores alarmados deverão ser listados e posicionados em mapas. Um alarme típico é lâmpada queimada. O usuário ainda poderá configurar a visualização de alarmes, ocultando alarmes indesejados. E todas as ocorrências deverão ser listadas em relatórios;
- 3) *Estado de operação:* O Aplicativo de operação remota deverá listar o estado de operação de todas as lâmpadas (acesa ou apagada) e posicionar cada um deles no mapa. O usuário

ainda poderá configurar a visualização dos estados de operação. Por exemplo, o usuário filtra da visualização todas as lâmpadas ligadas e assim apenas lâmpadas acesas serão mostradas e vice-versa. Com isso, é possível durante o dia detectar quais pontos de iluminação precisam de intervenção para reestabelecer o seu normal funcionamento;

- 4) *Operação individual:* Ao selecionar um nó sensor será possível supervisionar os valores de todos os seus parâmetros e configurá-los. Um exemplo de parâmetro configurável é o modo de operação do nó sensor, sendo automático ou manual. No modo automático, o relé fotocélula controla quando a lâmpada será ligada ou desligada, em função da presença ou não da luz. Já no modo manual, o usuário pode especificar os horários que a lâmpada irá ligar e desligar;
- 5) *Operação em grupo:* A operação em grupo é um requisito para configurar múltiplos nós sensores simultaneamente. Uma figura geométrica (círculo ou retângulo) é desenhada na interface gráfica e todos os nós sensores compreendidos pela área da figura serão configurados. Um caso típico de aplicação é quando se deseja apagar ou ligar as lâmpadas de uma rua/quadra. Sem este requisito o usuário teria que configurar individualmente os vários nós sensores da rua/quadra.
- 6) *Consumo de energia:* Medir e apresentar o consumo de energia em forma gráfica diariamente e mensalmente.
- 7) *Suportabilidade:* A capacidade de suportar diferentes modelos de nós sensores na solução é um requisito. O arquivo do tipo XML é adotado para descrever os dispositivos, cada modelo de nó sensor terá uma equivalência descritiva em arquivo. O fabricante descreve todas as características do modelo do nó sensor no arquivo e a suportabilidade ocorre quando, o arquivo de descrição é transferido e integrado a biblioteca dos modelos de nós sensores do sistema.

3.3 Escopo

O escopo do sistema de operação remota e supervisão de iluminação pública é dispor aplicativos no computador de operação e no computador industrial, estabelecer comunicação entre o computador industrial e o controlador de comunicação, possibilitar a integração de dispositivos com funcionalidades distintas nos aplicativos de gerenciamento e possibilitar, acesso remoto do sistema ao usuário.

Não é escopo do sistema, desenvolver ou descrever a rede sem fios que interconecta os nós sensores, os nós sensores e o controlador de comunicação. Porém, são trabalhos de pesquisa correlatos no laboratório e futuramente serão integrados e apresentados, como sendo a solução de um sistema de telecomando para a gestão da iluminação pública em grandes áreas.

O protocolo da rede sem fios que interconecta os nós sensores e o controlador de comunicação é o GGPSR (*Geocast Greedy Perimeter Stateless Routing*), projetado por Pantoni (2011).

4 Metodologia

A linguagem de programação adotada na metodologia é a C#. Esta linguagem apresenta as seguintes características: suporte a orientação a objetos baseado em herança simples de classes, herança múltipla de interfaces, gerenciamento de memória automática, tipagem forte e executa num ambiente gerenciado, no qual a segurança e integridade das operações efetuadas pelos programas podem ser garantidas (C#, 2006). A ferramenta adotada para se desenvolver as classes e componentes é o Visual C# 2008 Express Edition.

Na metodologia, arquivos do tipo XML são adotados. A linguagem XML se caracteriza por: fácil categorização, compreensível pelo ser humano, portabilidade, não é proprietário e pode ser usado para armazenamento de informação em qualquer plataforma, por aplicar um padrão internacional. Arquivos do tipo XML são aplicados para descrição dos dispositivos e definição da topologia da rede.

4.1 Suportabilidade

A suportabilidade é aplicada adotando-se para cada modelo de nó sensor um arquivo do tipo XML, nomeado como arquivo de descrição do nó sensor ou arquivo de descrição. No arquivo o fabricante do nó sensor descreve a coleção de parâmetros, grupos de variáveis de alarmes/processo/operação e opcionalmente grupos de IHM.

Os arquivos de descrição dos nós sensores possuem características e serviços que são consumidos pelo Servidor de Operação Remota, conector *IPGateway* do *CyberOPC* e o Aplicativo de Configuração / Topologia. Neste cenário, cada aplicação deve conhecer como o arquivo é estruturado e codificar individualmente a manipulação dos arquivos de descrição. Para evitar que cada aplicativo manipule os arquivos de descrição, o componente nomeado

XMLLibrary centraliza a manipulação dos arquivos e disponibilizam serviços aos aplicativos de gerenciamento, com isso os aplicativos de gerenciamento não precisam manipular os arquivos de descrição diretamente. Os aplicativos de gerenciamento aplicam o componente sem conhecer como o arquivo é estruturado e devem somente saber como os serviços do componente *XMLLibrary* são dispostos.

As próximas seções demonstram como o arquivo e o componente de software *XMLLibrary* são estruturados.

4.1.1 Especificação do arquivo de descrição

O arquivo do tipo XML de descrição do nó sensor é composto pelo nó principal *Sensor* e pelos nós aninhados *Parameters*, *Diagnostics*, *ProcessVariables*, *OperationStatuses* e *UserInterface*. Por definição apenas um nó *Sensor* deve existir por arquivo.

Os nós aninhados são os responsáveis por agrupar a coleção de parâmetros, os alarmes, as variáveis de processo e os grupos de IHM. A Figura 4.1 ilustra um exemplo do arquivo de descrição do nó sensor.

```
<?xml version="1.0" encoding="utf-8" ?>
<Sensor manufacturerID="1" modelID="1" manufacturerName="USP"
modelName="Sensor Modelo 01" revision="1">
  <Parameters>
  <Diagnostics>
  <ProcessVariables>
  <OperationStatuses>
  <UserInterface>
</Sensor>
```

Figura 4.1 - Arquivo de descrição do nó sensor

O nó principal *Sensor* é onde o fabricante do nó sensor dispõe as informações de identificação do modelo, como nome, número, revisão, nome do fabricante e número do fabricante.

As informações são mapeadas em forma de atributos no nó *Sensor*, o nome do modelo, por exemplo, é informado no atributo *modelName*. O atributo *modelName* apenas pode assumir valores textuais. A Tabela 4.1 descreve todos os atributos do nó principal *Sensor*.

Atributo	Descritivo	Entrada permitida
<i>modelName</i>	Nome do modelo do nó sensor	Texto
<i>manufacturerName</i>	Nome do fabricante do nó sensor	Texto
<i>modelID</i>	Identificador numérico do modelo	Número
<i>manufacturerID</i>	Identificador numérico do fabricante	Número
<i>revision</i>	Revisão do modelo	Número

Tabela 4.1 - Atributos do nó principal *Sensor*

Um fabricante de sensores pode descrever inúmeros modelos, por definição cada modelo é representado por um arquivo de descrição. É permitido que um mesmo modelo possua vários arquivos de descrição, desde que o atributo *revision* seja incrementado. Esta abordagem é aplicada no caso de um modelo de sensor sofrer mudanças nas suas características.

4.1.2 Coleção de parâmetros

O nó *Parameters* é responsável por agrupar todos os parâmetros do nó sensor. O nó *Parameter* representa um parâmetro, múltiplos nós *Parameter* podem ser aninhados ao nó *Parameters*, constituindo a coleção de parâmetros do dispositivo. A Figura 4.2 mostra um exemplo de coleção de parâmetros.

```

<?xml version="1.0" encoding="utf-8" ?>
<Sensor manufacturerID="1" modelID="1" manufacturerName="USP"
modelName="Sensor Modelo 01" revision="1">
  <Parameters>
    <Parameter index="1" name="LampActivity" type="2"
accessRights="1">
    <Parameter index="2" name="LampDimmerValue" type="3"
accessRights="1">
    <Parameter index="3" name="LampOperationTimeLimit" type="0"
accessRights="1">
    <Parameter index="4" name="LampOperationTimeCounter"
type="3" accessRights="1">
    <Parameter index="5" name="BatteryLevel" type="0"
accessRights="0">
    <Parameter index="6" name="BatteryLevelAlarms" type="2"
accessRights="0">
    <Parameter index="7" name="LampOperationAlarms" type="1"
accessRights="0">
    <Parameter index="8" name="LampConsumption" type="3"
accessRights="0">
  </Parameters>
  <Diagnostics>
  <ProcessVariables>
  <OperationStatuses>
  <UserInterface>
</Sensor>

```

Figura 4.2 – Coleção de parâmetros

O nó *Parameter* possui os seguintes atributos: *accessRights*, *type*, *name* e *index*. O atributo *accessRights* define o tipo de acesso ao parâmetro. Se configurado com o valor “0” é permitida somente a leitura, quando “1” é permitida a leitura e a escrita. A Tabela 4.2 enumera os valores de configuração do atributo *accessRights*.

Valor	Permissão de acesso ao parâmetro
0	Apenas leitura
1	Leitura e escrita

Tabela 4.2 – Enumeração do atributo *accessRights*

O parâmetro pode ser classificado como número inteiro, bit enumerado, enumerador, número com ponto flutuante, texto e data. A classificação limita a possibilidade de submeter valores inconsistentes a um parâmetro do nó sensor, como exemplo, um parâmetro definido como número inteiro não pode aceitar valores de data.

O atributo *type* é o campo responsável pela classificação, a Tabela 4.3 enumera todos os valores possíveis de configuração para o atributo.

Valor	Tipo do parâmetro
0	Número inteiro
1	Bit enumerador
2	Enumerador
3	Ponto flutuante
4	Texto
5	Data

Tabela 4.3 – Enumeração do atributo *type*

O nome do parâmetro é definido no atributo *name* e identifica o unicamente na coleção de parâmetros e é um texto que não possui espaços entre os caracteres. O atributo *index* é um índice numérico que identifica unicamente o parâmetro na coleção de parâmetros.

A Figura 4.3 ilustra um exemplo de parâmetro nomeado *LampActivity*, tipo de acesso leitura e escrita, classificado como enumerador e índice “1”.

```
<Parameter index="1" name="LampActivity" type="2" accessRights="1">
```

Figura 4.3 – Atributos do nó *Parameter*

O nó *Parameter* ainda pode aninhar outros nós para caracterizar o parâmetro. Os seguintes nós podem ser aninhados *Description*, *Enumerations*, *Range* e *Unit*.

O nó *Description* é utilizado para descrever a funcionalidade do parâmetro em múltiplos idiomas. O nó possui os seguintes atributos *language*, *help* e *label*. O atributo *language* define o idioma do conteúdo dos atributos *help* e *label*. O atributo *help* descreve a funcionalidade do parâmetro e o *label* apresenta um descritivo breve do parâmetro.

Para descrever a funcionalidade do parâmetro em múltiplos idiomas é necessário aninhar nós do tipo *Description* cada qual com o seu idioma, ao nó *Parameter*. Por definição o conteúdo do atributo *language* não pode se repetir entre os vários nós do tipo *Description* aninhados, ou seja, apenas uma ocorrência por idioma é permitida.

A Figura 4.4 mostra dois nós do tipo *Description* aninhados ao nó *Parameter*, a sigla en-US é empregada no atributo *language* do primeiro nó e determina o idioma dos atributos *help* e *label* como inglês e no segundo nó a sigla pt-BR fixa o conteúdo dos atributos como português do Brasil.

```

<Parameter index="8" name="LampConsumption" type="3" accessRights="0">
  <Description language="en-US" label="Lamp consumption" help="The
consumption of the lamp" />
  <Description language="pt-BR" label="Consumo da lâmpada"
help="Consumo da lâmpada" />
  <Unit>Kwh</Unit>
</Parameter>

```

Figura 4.4 – Nós *Description* do nó *Parameter*

A função do nó *Enumerations* é agrupar as enumerações do parâmetro. O nó *Enumeration* representa uma enumeração do parâmetro, múltiplos nós do tipo *Enumeration* podem ser aninhados ao nó *Enumerations*, formando a coleção de enumerações do parâmetro.

O nó *Enumerations* somente deve ser aninhado ao nó *Parameter* quando o parâmetro for classificado como enumerador ou bit enumerador. A Figura 4.5 mostra o nó *Enumerations* aninhado ao nó *Parameter*.

```

<Parameter index="1" name="LampActivity" type="2" accessRights="1">
  <Description language="en-US" label="Lamp activity" help="Controls
lamp activity" />
  <Description language="pt-BR" label="Estado de operação"
help="Controla a atividade da lâmpada" />
  <Enumerations>
    <Enumeration value="0" language="en-US">Off</Enumeration>
    <Enumeration value="0" language="pt-BR">Desliga</Enumeration>
    <Enumeration value="1" language="en-US">On</Enumeration>
    <Enumeration value="1" language="pt-BR">Liga</Enumeration>
  </Enumerations>
</Parameter>

```

Figura 4.5 – Nó *Enumerations*

O nó *Enumeration* representa uma possibilidade de valor válido do parâmetro e pode ser descrito em múltiplos idiomas. *Language* e *value* são os atributos do nó, sendo *language* o atributo que define o conteúdo do idioma da descrição da enumeração e o *value* o valor da enumeração. O descritivo da enumeração é definido no próprio nó *Enumeration*.

A Figura 4.5 apresenta quatro nós do tipo *Enumeration* aninhados ao nó *Enumeration*. Sendo os dois primeiros a representação da enumeração do valor “0”, traduzidos para o inglês e português respectivamente e os dois outros representam a enumeração do valor “1”, também traduzidos para os dois idiomas.

Os nós *Range* e *Unit* podem ser aninhados ao nó *Parameter* quando o parâmetro for classificado como numérico, ou seja, quando o conteúdo do atributo *type* do nó *Parameter* assumir o valor “0” ou “3”. O nó *Range* permite definir o valor mínimo e/ou o valor máximo aceitável pelo parâmetro, sendo o atributo *min.* responsável pelo menor valor e o atributo *max.* pelo maior. A

Figura 4.6 ilustra o nó *Range* aninhado com o nó *Parameter*, restringindo a faixa em operação, em “0” como mínimo e “100” como máximo.

```
<Parameter index="2" name="LampDimmerValue" type="3" accessRights="1">
  <Description language="en-US" label="Brightness of the bulb"
    help="Controls the brightness of the bulb" />
  <Description language="pt-BR" label="Luminosidade da lampada"
    help="Controla a luminosidade da lampada" />
  <Range min="0" max="100" />
  <Unit>%</Unit>
</Parameter>
```

Figura 4.6 – Nó *Range* e *Unit*

O nó *Unit* é adicionado ao nó *Parameter* quando se deseja especificar a unidade de operação do parâmetro. A Figura 4.6 apresenta o nó *Unit* aninhado ao nó *Parameter* especificando a unidade “%” como unidade do parâmetro.

Uma vez definida a coleção de parâmetros com todas as suas características, o arquivo de descrição permite ao fabricante do sensor, categorizar os parâmetros como alarmes, variáveis de processo e operação, sendo os nós *Diagnostics*, *ProcessVariables* e *OperationStatuses* os responsáveis pela categorização. As próximas seções demonstram como o arquivo programa o agrupamento.

4.1.3 Agrupamento de alarmes

O nó *Diagnostics* possui a atribuição de agrupar os parâmetros de alarmes do sensor. O nó *Diagnostic* representa um parâmetro de alarme, onde vários nós *Diagnostic* podem ser aninhados ao nó *Diagnostics*, constituindo a coleção de alarmes do sensor. O nó *Diagnostic* possui um único atributo *parameterName*.

O atributo *parameterName* é configurado com um nome de parâmetro da coleção de parâmetros e apenas parâmetros enumeradores e bit enumeradores podem ser configurados como alarmes. Como regra cada enumerador do parâmetro é considerado um item de alarme. O item de alarme é representado pelo nó *DiagnosticSettings*. O nó *DiagnosticSettings* possui três atributos *value*, *severity* e *filter*.

O atributo *value* é o valor do enumerador do parâmetro, ele atrela o item de alarme com a enumeração correspondente do parâmetro. Já o atributo *severity* é usado para definir a severidade do alarme em uma escala numérica entre zero e três, onde zero é o menos severo enquanto o três é o mais severo.

O atributo *filter* classifica a enumeração como um item de alarme ou não. Para não considerar a enumeração como um alarme, basta configurá-lo com o valor “y”. Caso contrário, o valor do atributo deve ser configurado com o valor “n”.

A Figura 4.7 apresenta um exemplo de coleção de alarmes com os seguintes parâmetros *BatteryLevelAlarms* e *LampOperationAlarms*.

```
<?xml version="1.0" encoding="utf-8" ?>
<Sensor manufacturerID="1" modelID="1" manufacturerName="USP"
modelName="Sensor Modelo 01" revision="1">
  <Parameters>
    <Diagnostics>
      <Diagnostic parameterName="BatteryLevelAlarms">
        <DiagnosticSettings value="0" severity="0" filter="y"/>
        <DiagnosticSettings value="1" severity="2" filter="n"/>
        <DiagnosticSettings value="2" severity="3" filter="n"/>
      </Diagnostic>
      <Diagnostic parameterName="LampOperationAlarms">
        <DiagnosticSettings value="1" severity="0" filter="y"/>
        <DiagnosticSettings value="2" severity="1" filter="n"/>
        <DiagnosticSettings value="4" severity="3" filter="n"/>
        <DiagnosticSettings value="8" severity="2" filter="n"/>
      </Diagnostic>
    </Diagnostics>
    <ProcessVariables>
    <OperationStatuses>
    <UserInterface>
  </Sensor>
```

Figura 4.7 – Coleção de alarmes

4.1.4 Agrupamento dos parâmetros de processo

O nó *ProcessVariables* possui a atribuição de agrupar os parâmetros de processo do nó sensor. O nó *ProcessVariable* representa um parâmetro de processo, vários nós *ProcessVariable* podem ser aninhados ao nó *ProcessVariables*, constituindo a coleção de parâmetros de processos do sensor. O nó *ProcessVariable* possui um único atributo *parameterName*.

O atributo *parameterName* é configurado com um nome de parâmetro da coleção de parâmetros e apenas parâmetros numéricos podem ser configurados como parâmetro de processo. A Figura 4.8 ilustra a coleção de parâmetros de processo com o parâmetro *LampConsumption*.

```
<?xml version="1.0" encoding="utf-8" ?>
<Sensor manufacturerID="1" modelID="1" manufacturerName="USP"
modelName="Sensor Modelo 01" revision="1">
  <Parameters>
  <Diagnostics>
  <ProcessVariables>
    <ProcessVariable parameterName="LampConsumption" />
  </ProcessVariables>
  <OperationStatuses>
  <UserInterface>
</Sensor>
```

Figura 4.8 – Coleção de parâmetros de processo

4.1.5 Agrupamento dos parâmetros de operação

O nó *OperationStatuses* possui a atribuição de agrupar os parâmetros de operação do nó sensor. O nó *OperationStatus* representa um parâmetro de operação, vários nós *OperationStatus* podem ser aninhados ao nó *OperationStatuses*, constituindo a coleção de parâmetros de operação do sensor. O nó *OperationStatus* possui dois atributos *name* e *parameterName*. O atributo *name* identifica unicamente o tipo de operação no agrupamento.

O atributo *parameterName* é configurado com um nome de parâmetro da coleção de parâmetros e apenas parâmetros do tipo enumerador podem ser configurados como parâmetro de operação. A Figura 4.9 ilustra a coleção de parâmetros de operação com o parâmetro *LampOperationStatus*.

```
<?xml version="1.0" encoding="utf-8" ?>
<Sensor manufacturerID="1" modelID="1" manufacturerName="USP"
modelName="Sensor Modelo 01" revision="1">
  <Parameters>
  <Diagnostics>
  <ProcessVariables>
  <OperationStatuses>
    <OperationStatus name="LampOperationStatus"
      parameterName="LampOperationStatus"/>
  </OperationStatuses>
  <UserInterface>
</Sensor>
```

Figura 4.9 – Coleção de parâmetros de operação do sensor

O nó *OperationStatus* pode aninhar dois nós o *Description* e o *Enumerations*. O nó *Description* é aplicado para descrever a operação em múltiplos idiomas. O nó possui os seguintes atributos *language*, *help* e *label*. O atributo *language* define o idioma do valor dos atributos *help* e *label*. O atributo *help* determina a funcionalidade do grupo e o *label* apresenta um descritivo breve do grupo.

Para descrever a funcionalidade a operação em múltiplos idiomas é necessário aninhar nós do tipo *Description* cada qual com o seu idioma, ao nó *OperationStatus*. Por definição o conteúdo do atributo *language* não pode se repetir entre os vários nós do tipo *Description* aninhados, ou seja, apenas uma ocorrência por idioma é permitida.

A Figura 4.10 apresenta dois nós do tipo *Description* aninhados ao nó *OperationStatus*, determinando o idioma do primeiro nó como inglês e o segundo como português.

```
<OperationStatus name="lampOperationStatus"
  parameterName="LampOperationStatus"/>
  <Description language="en-US" label=" Lamp operation status"
    help="" />
  <Description language="pt-BR" label=" Estado de operação das
    lâmpadas" help="" />
  <Enumerations>
</OperationStatuses>
```

Figura 4.10 – Nós *Description* do nó *OperationStatus*

O nó *Enumerations* possui a atribuição de agrupar os estados de operação do parâmetro. E cada estado de operação é representado pelo nó *Enumeration*, que deve ser aninhado ao nó *Enumerations*. Como regra cada enumerador do parâmetro é considerado um estado de operação. O nó *Enumeration* possui dois atributos *value* e *icon*.

O atributo *value* é configurado com o valor do enumerador do parâmetro, ele atrela um estado de operação com a enumeração correspondente do parâmetro. Já o atributo *icon* é usado para definir uma representação gráfica para o estado de operação. A Figura 4.11 apresenta os estados de operação de um parâmetro.

```
<Enumerations>
  <Enumeration value="1" icon="orange-dot.png"/>
  <Enumeration value="0" icon="white-dot.png"/>
</Enumerations>
```

Figura 4.11 – Estados de operação de um parâmetro

4.1.6 Agrupamento de IHM

O nó *UserInterface* pode ser aninhado ao nó principal *Sensor* opcionalmente quando se deseja definir grupos de IHM para o sensor. Os parâmetros do sensor podem ser agrupados por

funcionalidades, por exemplo, parâmetros da lâmpada podem ser agrupados e disponibilizados pelo aplicativo de gerenciamento em um grupo de IHM.

O nó *Groups* é aninhado ao nó *UserInterface*, define a coleção de grupos de IHM. O nó *Group* representa um grupo, múltiplos nós *Group* podem ser adicionados ao nó *Groups*, definindo a coleção de IHM do sensor.

O nó *Group* possui um único atributo *name* e é utilizado para identificar unicamente o grupo na coleção. A Figura 4.12 demonstra o nó *UserInterface* com dois grupos, *battery* e *lamp*.

```
<?xml version="1.0" encoding="utf-8" ?>
<Sensor manufacturerID="1" modelID="1" manufacturerName="USP"
modelName="Sensor Modelo 01" revision="1">
  <Parameters>
  <Diagnostics>
  <ProcessVariables>
  <OperationStatuses>
  <UserInterface>
    <Groups>
      <Group name="battery">
      <Group name="lamp">
    </Groups>
  </UserInterface>
</Sensor>
```

Figura 4.12 – Coleção de IHM

O nó *Group* pode aninhar dois nós o *Description* e o *Parameters*. O nó *Description* é aplicado para descrever o grupo em múltiplos idiomas. O nó possui os seguintes atributos *language*, *help* e *label*. O atributo *language* define o idioma do valor dos atributos *help* e *label*. O atributo *help* determina a funcionalidade do grupo e o *label* apresenta um descritivo breve do grupo.

Para descrever a funcionalidade do grupo em múltiplos idiomas é necessário aninhar nós do tipo *Description* cada qual com o seu idioma, ao nó *Group*. Por definição o conteúdo do atributo *language* não pode se repetir entre os vários nós do tipo *Description* aninhados, ou seja, apenas uma ocorrência por idioma é permitida.

A Figura 4.13 apresenta dois nós do tipo *Description* aninhados ao nó *Group*, determinado o idioma do primeiro nó como inglês e o segundo como português.

```
<Group name="battery">
  <Description language="en-US" label="Battery" help="" />
  <Description language="pt-BR" label="Bateria" help="" />
  <Parameters>
</Group>
```

Figura 4.13 – Nós *Description* do nó *Group*

O nó *Parameters* possui a atribuição de agrupar os parâmetros do grupo IHM. O nó *Parameter* representa um parâmetro, vários nós *Parameter* podem ser aninhados ao nó *Parameters*, constituindo a coleção de parâmetros do grupo. O nó *Parameter* possui o atributo *parameterName*, sendo configurado com um nome de parâmetro da coleção de parâmetros. A Figura 4.14 demonstra um grupo de IHM nomeado como *battery*, com dois parâmetros agrupados *BatteryLevel* e *BatteryLevelAlarms*.

```
<Group name="battery">
  <Description language="en-US" label="Battery" help="" />
  <Description language="pt-BR" label="Bateria" help="" />
  <Parameters>
    <Parameter parameterName="BatteryLevel" />
    <Parameter parameterName="BatteryLevelAlarms" />
  </Parameters>
</Group>
```

Figura 4.14 – IHM *Battery*

4.1.7 Componente de software *XMLLibrary*

O componente de software *XMLLibrary* tem como função centralizar a manipulação dos arquivos de descrição e disponibilizar serviços aos aplicativos de gerenciamento. A manutenção é facilitada, pois caso mudanças estruturais no arquivo sejam aplicadas, o componente será adequado ao novo cenário e os serviços são mantidos aos aplicativos de gerenciamento.

O *XMLLibrary* é estruturado em classes de software e codificado na linguagem C#, onde cada classe dispõe de um subconjunto de serviços. As principais funcionalidades são listar os modelos de sensores de um determinado diretório do computador e disponibilizar as informações dos arquivos de descrição. As classes do componente são *Library*, *Parameter*, *UIGroup* e *AssetDescriptor*.

A classe *Library* é a responsável por listar os modelos de sensores de um determinado diretório. Ao se construir um objeto da classe é requerido que o caminho do diretório dos arquivos de descrição seja informado. O método *readLibrary* executa a leitura no diretório previamente informado e disponibiliza a coleção dos modelos de sensores.

A Figura 4.15 apresenta o diagrama de classe UML (*Unified Modeling Language*) da classe *Library*, sendo *models* a propriedade pública que disponibiliza a coleção dos modelos de sensores.

Library
-libraryPath : string
+models : Dictionary<string,XMLLibrary.AssetDescriptor> = new Dictionary<string, AssetDescriptor>()
+Library(in libraryPath : string)
+readLibrary()
+getDescriptor(in descriptorID : string) : AssetDescriptor

Figura 4.15 – Diagrama de classe UML - *Library*

Na classe *Library* é possível a partir da lista de modelos selecionar um determinado modelo e obter todas as informações de descrição do mesmo. A propriedade *models* recebe como

critério o identificador do sensor e retorna um objeto do tipo *AssetDescriptor* que disponibiliza as informações de descrição do sensor. O identificador do sensor é composto pelos atributos *manufacturerID* e *modelID* do nó principal *Sensor* do arquivo de descrição.

Opcionalmente, o método *getDescriptor* também disponibiliza tal serviço, como valor de entrada recebe o identificador do sensor e retorna um objeto do tipo *AssetDescriptor*.

A Figura 4.16 mostra o diagrama de classe UML da classe *AssetDescriptor*, e como as informações de um modelo de nó sensor são disponibilizadas. Ao se construir um objeto da classe *AssetDescriptor* é requerido que o idioma seja informado.

Figura 4.16 – Diagrama de classe UML - *AssetDescriptor*

A propriedade *parameterList* é a coleção de parâmetros disponíveis no nó sensor, *diagnosticList* dispõe a coleção de alarmes, *processVariablesList* lista dos parâmetros de processo e *userInterfaceGroupList* expõe os grupos de IHM. As informações de identificação do modelo são armazenadas nas seguintes propriedades *ManufacturerID*, *ModelID*, *ManufacturerName*, *ModelName* e ID.

A Tabela 4.4 mapeia as propriedades públicas da classe *AssetDescriptor* com as informações contidas no arquivo de descrição. Por exemplo, a propriedade *ManufacturerID* da classe é equivalente ao atributo *manufacturerID* do nó principal *Sensor*.

Propriedade da classe	Arquivo de descrição
<i>parameterList</i>	Coleção de parâmetros, disponível no nó <i>Parameters</i>
<i>diagnosticList</i>	Coleção de alarmes, disponível no nó <i>Diagnostics</i>
<i>processVariablesList</i>	Coleção de parâmetros de processo, disponível no nó <i>processVariables</i>
<i>userInterfaceGroupList</i>	Coleção de IHM, disponível no nó <i>UserInterface</i>
<i>ManufacturerID</i>	Atributo <i>manufacturerID</i> do nó principal <i>Sensor</i>
<i>ModelID</i>	Atributo <i>modelID</i> do nó principal <i>Sensor</i>
<i>ManufacturerName</i>	Atributo <i>manufacturerName</i> do nó principal <i>Sensor</i>
<i>ModelName</i>	Atributo <i>modelName</i> do nó principal <i>Sensor</i>
<i>ID</i>	Identificador do sensor: Composto pelos atributos <i>manufacturerID</i> e <i>modelID</i> do nó principal <i>Sensor</i>

Tabela 4.4 - Equivalência entre os atributos da classe *AssetDescriptor* e o arquivo de descrição

Na classe *AssetDescriptor* é possível a partir da coleção de parâmetros, selecionar um determinado parâmetro e obter as informações de descrição do mesmo. Este serviço é disponível pela propriedade *parameterList*. A propriedade recebe como critério o nome do parâmetro, retornando um objeto do tipo *Parameter* que disponibiliza as informações de descrição.

O serviço se estende para as propriedades *diagnosticList* e *processVariablesList*, a partir de um nome de parâmetro de alarme ou processo é possível obter a descrição do parâmetro. As

três coleções *parameterList*, *diagnosticList* e *processVariablesList* são coleções da classe *Parameter*, ou seja, são coleções do tipo *Parameter*.

A classe *Parameter* representa um parâmetro, e mapeia os atributos do nó *Parameter* do arquivo de descrição em forma de propriedades na classe. Ao se construir um objeto da classe é requerido que seja informado os seguintes valores *name*, *accessRigths*, *dataType*, *label*, *help* e *index*. Na Figura 4.17 apresenta o diagrama de classe UML da classe *Parameter* e como dispõe as informações de um parâmetro do sensor.

Figura 4.17 – Diagrama de classe UML - *Parameter*

A classe dispõe do serviço de inicialização das informações de identificação e das coleções a partir do arquivo de descrição do sensor. O método *bindAssetObject* programa este serviço, recebe como parâmetro de entrada o arquivo e preenche as propriedades do objeto.

A Tabela 4.5 mapeia as propriedades públicas da classe *Parameter* com as informações de descrição do nó *Parameter*. Como exemplo, a propriedade *Name* da classe é equivalente ao atributo *name* do nó *Parameter*.

Propriedade da classe	Arquivo de descrição
<i>Name</i>	Atributo <i>name</i> do nó <i>Parameter</i>
<i>AccessRights</i>	Atributo <i>accessRights</i> do nó <i>Parameter</i>
<i>DataType</i>	Atributo <i>type</i> do nó <i>Parameter</i>
<i>Index</i>	Atributo <i>index</i> do nó <i>Parameter</i>
<i>Label</i>	Atributo <i>label</i> do nó <i>Description</i> aninhado ao nó <i>Parameter</i>
<i>Help</i>	Atributo <i>help</i> do nó <i>Description</i> aninhado ao nó <i>Parameter</i>
<i>enumeration</i>	Coleção de enumeradores, disponível no nó <i>Enumerations</i>

Tabela 4.5 – Equivalência entre os atributos da classe *Parameter* e o arquivo de descrição

A propriedade *userInterfaceGroupList* da classe *Library* é a representação dos grupos de IHM presentes no arquivo de descrição do sensor. A coleção armazena objetos do tipo *UIGroup*.

A classe *UIGroup* representa um grupo de IHM, e mapeia os atributos do nó *Group* do arquivo de descrição em propriedades. Ao se construir um objeto da classe é solicitado os seguintes valores *name*, *label* e *help*. A Figura 4.18 mostra o diagrama de classe UML da classe *UIGroup* com todas as suas propriedades.

UIGroup
-name : string -label : string -help : string +parameterList : Dictionary<string,XMLLibrary.Parameter> = new Dictionary<string, Parameter>() +Name() : string +Label() : string +UIGroup(in name : string, in label : string, in help : string)

Figura 4.18 – Diagrama de classe UML - *UIGroup*

A Tabela 4.6 mapeia as propriedades públicas da classe *UIGroup* com as informações de descrição do nó *Group*. Como exemplo, a propriedade *Name* da classe é equivalente ao atributo *name* do nó *Group*.

Propriedade da classe	Arquivo de descrição
<i>parameterList</i>	Coleção de parâmetros do grupo, disponível no nó <i>Parameters</i>
<i>Name</i>	Atributo <i>name</i> do nó <i>Group</i>
<i>Label</i>	Atributo <i>label</i> do nó <i>Description</i> aninhado ao nó <i>Group</i>

Tabela 4.6 - Equivalência entre os atributos da classe *UIGroup* e o arquivo de descrição

Resumidamente para, o componente funciona da seguinte forma um objeto da classe *Library* deve ser criado e o método *readLibrary* deve ser invocado, para disponibilizar os serviços. A Figura 4.19 exemplifica a criação do componente aplicando-se a linguagem de programação C#.

No exemplo, a primeira linha cria um objeto da classe *Library* e nomeia como *library*. O construtor da classe *Library* recebe como parâmetro o diretório “C:\\Library”, neste caso todos os arquivos de modelos de sensores presentes no diretório são listados. Na segunda linha, o método *readLibrary* é invocado e os serviços são disponibilizados.

```
Library library = new Library("C:\\Library");  
library.readLibrary();
```

Figura 4.19 – Criando o componente *XMLLibrary*

Para acessar os serviços do componente basta acessar as propriedades das classes listadas durante esta seção. A Figura 4.20 ilustra um laço *foreach* iterando por todos os modelos de sensores disponíveis. O laço varre a coleção de modelos, propriedade *models* do objeto *library*, e um modelo de sensor é disponibilizado pelo objeto *assetDescriptor* a cada iteração do laço. Adicionalmente, através do objeto *assetDescriptor* os serviços da classe *AssetDescriptor* podem se invocados.

```
foreach (KeyValuePair<string,AssetDescriptor> item in library.models)
{
    AssetDescriptor assetDescriptor = item.Value;
}
```

Figura 4.20 – Iterando sobre a lista de modelos

4.2 Comunicação entre o sistema e os nós sensores

A comunicação funciona da seguinte forma: os aplicativos de gerenciamento dispostos no computador de operação solicitam dados de supervisão via rede Ethernet ao computador industrial instalado na cabine. A solicitação é recebida pelo computador da cabine e repassada ao controlador de comunicação conectado a porta serial.

O controlador de comunicação por sua vez transmite via rede sem fio a solicitação, identificando o pedido. Os nós sensores recebem as notificações, processam a requisição e transmitem a resposta via rede sem fio até o controlador. A resposta é recebida pelo controlador via rede sem fio e repassada via porta serial ao computador, o computador industrial envia a resposta ao computador de operação pela rede Ethernet.

4.2.1 CyberOPC

A arquitetura do *CyberOPC* é estruturada como cliente e servidor. O servidor disponibiliza as variáveis de supervisão e o cliente se conecta ao servidor para requisitar os dados. Múltiplos clientes podem se conectar a um único servidor simultaneamente.

O *CyberOPC* pode programar no seu servidor o suporte SSL (*Secure Socket Layer*), assim os dados são criptografados e acessados via HTTPS (*Hyper Text Transfer Protocol Secure*), possibilitando um canal seguro de comunicação entre servidor e clientes.

O *CyberOPC* instalado no servidor é estruturado da seguinte forma: *Application Server* e *Module Library*. O *Application Server* é a infraestrutura responsável por receber as notificações via HTTPS direcionar ao *module library* correspondente e disponibilizar a informação pela rede Ethernet via padrão *JSON-RPC* (*Java Script Object Notation - Remote Procedure Call*).

O *module library* é uma coleção de conectores, cada conector representa um tipo de dispositivo, ou seja, em um único servidor é possível integrar múltiplos equipamentos. O protocolo de comunicação entre o computador e o dispositivo é embutido no conector. Assim, o cliente *CyberOPC* ao solicitar informações de supervisão ao servidor não necessita conhecer o funcionamento do protocolo de comunicação entre computador e dispositivo. A Figura 4.21 apresenta a arquitetura do servidor *CyberOPC*.

Figura 4.21 – Arquitetura do servidor *CyberOPC*

Fonte: TORRISI (2011)

4.2.2 Protocolo de comunicação entre o *CyberOPC* e o controlador de comunicação

No computador disposto na cabine é disponibilizado um servidor *CyberOPC* com um conector do controlador de comunicação integrado a coleção *module library*, nomeado como conector *IPGateway*. Os aplicativos de gerenciamento são clientes *CyberOPC*.

Para solicitar dados de leitura e escrita, os aplicativos devem informar o identificador do sensor e índice do parâmetro e enviar ao servidor *CyberOPC*. No caso da escrita, o valor desejado de escrita também deve ser informado. Ao receber a solicitação, o pedido é direcionado ao conector do controlador de comunicação.

O conector recebe o pedido, um pacote de solicitação padronizado é formatado e transmitido ao controlador de comunicação via porta serial. E por fim, o controlador transmite a requisição via rede sem fio.

No caso de alarme, o processo é reverso, o nó sensor envia a requisição do pacote de alarme via rede sem fio direcionada ao controlador de comunicação correspondente da célula.

4.2.3 Formatação dos pacotes de requisição e resposta dos serviços de leitura, escrita e notificação de alarme.

Para definir a comunicação entre o sistema (Aplicativos e *CyberOPC / IPGateway*) e o controlador de comunicação, todos os serviços e seus respectivos formatos de requisição e resposta são apresentados nesta seção.

Os seguintes serviços são estabelecidos: leitura/escrita individual ou em grupo e alarmes. Leituras e escritas são requisições feitas pelo sistema para os nós sensores e alarmes são requisições do nó sensor ao sistema. Dependendo do tipo de serviço, pode haver ou não resposta atrelada. Para identificação única do serviço, é estabelecido o número de função do serviço. A

Tabela 4.7 lista os serviços do protocolo de comunicação e o seu número de função correspondente.

Função	Serviço
1	Leitura individual
2	Leitura em grupo
3	Escrita individual
4	Escrita em grupo
5	Alarme

Tabela 4.7 – Tipos de serviços do protocolo de comunicação e o seu número de função correspondente

Leitura individual: O serviço de leitura individual é aplicado na leitura do valor de um parâmetro para um determinado nó sensor. O funcionamento é baseado na requisição e resposta, o sistema envia o pacote de requisição identificando o nó sensor e o parâmetro, e o nó sensor retransmite ao sistema a resposta formatada com o valor do parâmetro requisitado.

O pacote de requisição possui os seguintes campos: função, coordenada do nó sensor, sequencia do pacote, índice do parâmetro e CRC. O campo função é preenchido com o número de serviço da leitura individual, ou seja, “1”.

O campo sequencia do pacote é utilizado pra controle de requisições, para cada nova requisição, um novo número é criado e o campo coordenada do nó sensor é configurado com as coordenadas geodésicas do nó sensor destino, a latitude e longitude. O campo índice do parâmetro armazena o índice do parâmetro de leitura. E o último campo, o CRC, é utilizado pelo controlador de comunicação para averiguar se houve falha na transmissão. A Tabela 4.8 apresenta os campos do pacote de requisição para leitura individual, além do tamanho em *bytes* de cada campo.

Campo	Tamanho (<i>bytes</i>)
função	1
coordenada do nó sensor	4 <i>bytes</i> para latitude 4 <i>bytes</i> para longitude
sequencia do pacote	1
índice do parâmetro	1
CRC	2

Tabela 4.8 – Campos do pacote de requisição para leitura individual

Por sua vez, o pacote de resposta é formatado com os seguintes campos: função, coordenada do nó sensor, sequencia do pacote, comprimento do dado, dado e CRC.

O campo função também é preenchido com o valor “1”, e o valor da coordenada do nó sensor é configurado com as coordenadas geodésicas do nó sensor, a latitude e longitude.

O valor do campo sequencia do pacote é configurado com o mesmo valor do campo de sequencia do pacote da requisição. Com este artifício, ao receber um pacote de resposta do serviço de leitura individual, o sistema tem condições de resgatar o pacote que originou aquela resposta recém-chegada. Sem o pacote original, o sistema, não é capaz de identificar qual é o índice do parâmetro no pacote de resposta, pois o mesmo não possui tal campo.

O campo comprimento do dado é atribuído com o tamanho do campo dado. O campo dado é configurado com o valor do parâmetro requisitado, e o seu tamanho depende do tipo do parâmetro. Por exemplo, parâmetros do tipo ponto flutuante ocupam quatro bytes.

O campo CRC é utilizado pelo sistema, para averiguar se houve falha na comunicação serial entre o controlador de comunicação e o sistema. A Tabela 4.9 apresenta os campos do pacote de resposta para leitura individual, além do tamanho em *bytes* de cada campo.

Campo	Tamanho (bytes)
função	1
coordenada do nó sensor	8
sequencia do pacote	1
comprimento do Dado	1
dado	Definido pelo tipo do parâmetro
CRC	2

Tabela 4.9 – Campos do pacote de resposta para leitura individual

Leitura em grupo: O serviço de leitura em grupo é aplicado na leitura do valor de um parâmetro para múltiplos nós sensores. O funcionamento é baseado em uma requisição e múltiplas respostas, o sistema envia o pacote de requisição identificando uma região geográfica e o índice do parâmetro, e todos os nós sensores posicionado dentro da região, formatam a resposta para retransmitir o valor do parâmetro requisitado.

O pacote de requisição possui os seguintes campos: função, coordenada da região, sequencia do pacote, índice do parâmetro e CRC. O campo função é preenchido com o número de serviço da leitura em grupo, ou seja, “2”.

Dois formatos de figuras geométricas são empregados para o serviço de leitura em grupo, o retângulo e o círculo. O campo coordenada da região é formatado em função da figura geométrica adotada.

No caso do retângulo, o campo é formatado obedecendo a seguinte regra: todos os quatro vértices do retângulo são adicionados ao campo e cada deles é representado por uma coordenada geodésica. O primeiro vértice do campo a ser adicionado é o superior direito, e os demais são anexados ao campo seguindo a orientação anti-horária até completar o quarto e último. Cada vértice utiliza oito *bytes*, quatro para latitude e mais quatro para longitude, assim o campo

coordenada da região é preenchido na sua totalidade com os trinta e dois *bytes*. A Figura 4.22 ilustra um retângulo e os seus vértices enumerados obedecendo à regra para formatação do campo Coordenada da região para leitura em grupo.

Figura 4.22 – Retângulo e os seus vértices enumerados obedecendo à regra para formatação do campo Coordenada da região para leitura em grupo

No caso do círculo, o campo coordenada da região é formatado obedecendo a seguinte sequencia: a coordenada geodésica do centro do círculo e qualquer coordenada geodésica que esteja ao longo da circunferência do mesmo. As duas coordenadas geodésicas totalizam dezesseis bytes e outros dezesseis bytes do campo são preenchidos com valores zeros. A Figura 4.23 ilustra como o campo Coordenada da região é formatado para leitura em grupo utilizando o círculo.

Figura 4.23 – Regra para formatação do campo Coordenada da região para leitura em grupo utilizando o círculo

O campo sequencia do pacote é utilizado pra controle de requisições, para cada nova requisição, um novo número é criado e o campo Índice do parâmetro armazena o índice do parâmetro de leitura. E o último campo, o CRC, é utilizado pelo controlador de comunicação para averiguar se houve falha na transmissão.

A resposta da requisição é transmitida pelos nós sensores posicionados na área da figura, e o formato do pacote de resposta empregado é o mesmo da leitura individual, estipulado na Tabela 4.9.

Escrita individual: O serviço de escrita individual é aplicado na escrita do valor de um parâmetro para um determinado nó sensor. O funcionamento é baseado somente na requisição, o sistema envia o pacote de requisição identificando o nó sensor, o parâmetro e valor da escrita.

O pacote de requisição possui os seguintes campos: função, coordenada do nó sensor, sequencia do pacote, índice do parâmetro, comprimento do dado, dado e CRC. O campo função é preenchido com o número de serviço da escrita individual, ou seja, “3”.

Os campos coordenada do nó sensor, sequencia do pacote e índice do parâmetro são análogos aos campos de requisição do serviço de leitura individual e são configurados com os mesmos valores. O campo comprimento do dado é atribuído com o tamanho do campo dado. O campo dado é configurado com o valor de escrita do parâmetro, e o seu tamanho depende do tipo do parâmetro.

E o último campo, o CRC, é utilizado pelo controlador de comunicação para averiguar se houve falha na transmissão. A Tabela 4.10 apresenta os campos do pacote de requisição para escrita individual, além do tamanho em *bytes* de cada campo.

Campo	Tamanho (<i>bytes</i>)
função	1
coordenada do nó sensor	8
sequencia do pacote	1
índice do parâmetro	1
comprimento do Dado	1
dado	Definido pelo tipo do parâmetro
CRC	2

Tabela 4.10 – Campos do pacote de requisição para escrita individual

Por não possuir uma resposta atrelada a requisição, duas formas podem ser aplicadas para obter a confirmação da escrita. A primeira é enviar uma solicitação de leitura individual para averiguar se o valor foi escrito. E a segunda é mapear o parâmetro como alarme, ocorrendo à escrita no parâmetro, o alarme é enviado automaticamente pelo nó sensor ao sistema, condicionado somente aos parâmetros do tipo enumerador ou bit enumerador.

Escrita em grupo: O serviço de escrita em grupo é aplicado na escrita do valor de um parâmetro para múltiplos nós sensores. O funcionamento é baseado somente na requisição, o sistema envia o pacote de requisição identificando uma região geográfica, o índice do parâmetro e o valor de escrita, e todos os nós sensores posicionados dentro da região recebem a requisição de escrita.

O pacote de requisição possui os seguintes campos: função, coordenada da região, sequencia do pacote, índice do parâmetro, comprimento do dado, dado e CRC. O campo função é preenchido com o número de serviço da leitura em grupo, ou seja, “4”.

Os campos coordenada da região, sequencia do pacote e índice do parâmetro são análogos aos campos de requisição do serviço de leitura em grupo e são configurados com os

mesmos valores. O campo comprimento do dado é atribuído com o tamanho do campo dado. O campo dado é configurado com o valor de escrita do parâmetro, e o seu tamanho depende do tipo do parâmetro.

E o último campo, o CRC, é utilizado pelo controlador de comunicação para averiguar se houve falha na transmissão. A Tabela 4.11 apresenta os campos do pacote de requisição para escrita em grupo, além do tamanho em *bytes* de cada campo.

Campo	Tamanho (<i>bytes</i>)
função	1
coordenada da região	32
sequencia do pacote	1
índice do parâmetro	1
comprimento do Dado	1
dado	Definido pelo tipo do parâmetro
CRC	2

Tabela 4.11 – Campos do pacote de requisição para escrita em grupo

Alarme: O serviço de alarme é aplicado para notificar ocorrências no nó sensor sem a necessidade de leituras periódicas nos parâmetros do sensor. O funcionamento é baseado somente na requisição, o nó sensor envia para o sistema o pacote de requisição identificando o nó sensor e a nova condição de alarme.

O pacote de requisição possui os seguintes campos: função, coordenada do nó sensor, sequencia do pacote, comprimento do dado, alarme anterior, alarme atual e CRC. O campo função é preenchido com o número de serviço de alarme, ou seja, “5”.

O campo coordenada do nó sensor é configurado com as coordenadas geodésicas do nó sensor fonte e o campo sequencia do pacote é atribuído pelo nó sensor um número de controle interno.

Na sequencia, o campo comprimento do dado é atribuído com o valor quatro, pois representa a soma dos tamanhos em bytes dos campos alarme anterior e alarme atual, cada qual com dois bytes de tamanho. O campo alarme anterior é configurado com o valor posterior do alarme e o campo alarme atual indica a atual condição de alarme do nó sensor que enviou a requisição ao sistema.

E o último campo, o CRC, é utilizado pelo sistema para averiguar se houve falha na transmissão. A Tabela 4.12 apresenta a formatação dos campos do pacote de notificação de alarme, além do tamanho em *bytes* de cada campo.

Campo	Tamanho (<i>bytes</i>)
função	1
coordenada do nó sensor	8
sequencia do pacote	1
comprimento do Dado	1
alarme anterior	2
alarme atual	2
CRC	2

Tabela 4.12 – Campos do pacote de notificação de alarme

4.3 Aplicativos de gerenciamento

O sistema propõe dois aplicativos para a sua configuração e operação, sendo o aplicativo de Configuração / Topologia e o Servidor de operação remota. O aplicativo de Configuração é o responsável por distribuir os sensores em células através do arquivo de topologia.

O Servidor de operação remota possui duas atribuições disponibilizar a operação do sistema em páginas HTML, para acesso remoto através de navegadores de Internet e manter os serviços de monitoração ativos. Os serviços de monitoração são subdivididos em três: monitoração de alarmes, monitoração de estado de operação e monitoração do consumo.

4.3.1 Aplicativo de Configuração / Topologia

A principal funcionalidade deste aplicativo é criar a topologia do sistema. A topologia é a organização dos nós sensores por controlador de comunicação (células). Esta distribuição é representada por um arquivo do tipo XML, chamado arquivo de topologia.

A segunda funcionalidade deste aplicativo é a capacidade de configuração, ou seja, enviar comandos de leitura / escrita aos nós sensores. Para isso, o mesmo deve ser um cliente do tipo *CyberOPC*.

4.3.2 Arquivo de Topologia

Nesta seção é apresentada a especificação do arquivo de topologia. O Aplicativo de Configuração / Topologia deverá possuir a capacidade de manipular este arquivo, com o objetivo de facilitar a distribuição dos nós sensores por controlador de comunicação.

O arquivo de topologia lista os nós sensores e controladores disponíveis de um município ou região e define as células. O arquivo é estruturado da seguinte forma: nó principal *Configuration* e os nós aninhados *Sensors*, *Gateways* e *Topology*. A Figura 4.24 apresenta a estrutura do arquivo de topologia.

```
<?xml version="1.0" encoding="utf-8" ?>
<Configuration>
  <Sensors>
  <Gateways>
  <Topology>
</Configuration>
```

Figura 4.24 – Estrutura do arquivo de topologia

O nó principal *Configuration* é somente um agrupador de nós. O nó aninhado *Sensors* forma a coleção de nós sensores disponível. O nó *Sensor* representa um sensor, múltiplos nós *Sensor* podem ser aninhados ao nó *Sensors*, assim formando a coleção de nós sensores. A Figura 4.25 apresenta a coleção de nós sensores.

```
<Sensors>
  <Sensor id="1" descriptorID="0101" x="23,12" y="12,25" />
  <Sensor id="2" descriptorID="0101" x="23,14" y="12,43" />
  <Sensor id="3" descriptorID="0101" x="23,78" y="12,34" />
  <Sensor id="4" descriptorID="0101" x="23,67" y="12,54" />
  <Sensor id="5" descriptorID="0101" x="23,87" y="12,32" />
  <Sensor id="6" descriptorID="0101" x="23,98" y="12,21" />
  <Sensor id="7" descriptorID="0101" x="23,65" y="12,34" />
  <Sensor id="8" descriptorID="0101" x="23,56" y="12,54" />
  <Sensor id="9" descriptorID="0101" x="23,12" y="12,81" />
  <Sensor id="10" descriptorID="0101" x="23,56" y="12,32" />
  <Sensor id="11" descriptorID="0101" x="23,34" y="12,21" />
  <Sensor id="12" descriptorID="0101" x="23,87" y="12,34" />
  <Sensor id="13" descriptorID="0101" x="23,45" y="12,78" />
</Sensors>
```

Figura 4.25 – Coleção de nós sensores do arquivo de topologia

O nó *Sensor* possui os seguintes atributos *x*, *y*, *descriptorID* e *id*. Os atributos *x* e *y* representam as coordenadas geodésicas do sensor, onde *x* é latitude e *y* é a longitude, são

números do tipo ponto flutuante, onde o separador da casa decimal é a vírgula. O atributo *descriptorID* é o que relaciona o sensor com o seu arquivo de descrição e o id é utilizado para identificar unicamente o nó sensor.

O atributo *descriptorID* é composto pelos números do fabricante e modelo, assim é possível estabelecer um vínculo com o arquivo de descrição do sensor. Como exemplo, o primeiro nó sensor da coleção da Figura 4.25, apresenta o atributo *descriptorID* configurado com o valor “0101”, sendo o primeiro conjunto “01”, o número do fabricante e o segundo conjunto “01” o identificador do modelo de sensor. Com as informações de identificação, números do fabricante e modelo, é possível utilizar o componente *XMLLibrary* para obter os detalhes de descrição do sensor.

O nó aninhado *Gateways* forma a coleção de controladores de comunicação disponível. O nó *Gateway* representa um controlador, múltiplos nós *Gateway* podem ser aninhados ao nó *Gateways*, assim formando a coleção de controladores. A Figura 4.26 apresenta a coleção de controladores de comunicação.

```
<Gateways>
  <Gateway id="1" descriptorID="0102" x="23,12" y="12,34" />
  <Gateway id="2" descriptorID="0102" x="23,56" y="12,54" />
  <Gateway id="3" descriptorID="0102" x="23,67" y="12,67" />
  <Gateway id="4" descriptorID="0102" x="23,76" y="12,32" />
  <Gateway id="5" descriptorID="0102" x="23,56" y="12,1" />
  <Gateway id="6" descriptorID="0102" x="23,67" y="12,12" />
</Gateways>
```

Figura 4.26 - Coleção de controladores de comunicação do arquivo de topologia

O nó *Gateway* é análogo ao nó *Sensor*, apenas representa um controlador ao invés de um nó sensor. Como exemplo, o primeiro controlador da coleção da Figura 4.26, apresenta o atributo *descriptorID* configurado com o valor “0102”. O controlador de comunicação deve possuir um arquivo de descrição correspondente.

A coleção de células é representada pelo nó aninhado *Topology*. Um novo nó *Gateway* é apresentado, este nó corresponde a uma célula, nós deste tipo podem ser aninhados ao nó *Topology* formando a lista de células. A Figura 4.27 ilustra a coleção de células de um arquivo de topologia.

```
<Topology>  
  <Gateway id="1">  
  <Gateway id="6">  
</Topology>
```

Figura 4.27 – Coleção de células

O nó que representa a célula possui o atributo *id*. O atributo é configurado com um identificador da coleção de controladores de comunicação. Como regra o valor do atributo *id* não pode se repetir entre os vários nós *Gateway*, ou seja, um controlador de comunicação é responsável por apenas uma célula.

A coleção de sensores agrupados em uma célula é representada pelo nó *Sensors*. Nó do tipo *Sensor* representa um sensor, múltiplos nós do tipo *Sensor* anexados ao nó *Sensors* configura a coleção de sensores de uma célula. A Figura 4.28 mostra grupo de sensores de uma célula.

```
<Topology>
  <Gateway id="1">
    <Sensors>
      <Sensor id="1" />
      <Sensor id="2" />
      <Sensor id="3" />
      <Sensor id="4" />
      <Sensor id="5" />
    </Sensors>
  </Gateway>
  <Gateway id="6">
</Topology>
```

Figura 4.28 – Sensores de uma célula

O nó que representa um sensor na célula especifica o atributo id. O id é preenchido com um identificador da coleção de nós sensores do arquivo de topologia. Por definição o valor do atributo não pode se repetir entre os vários nós *Sensor* dentro do arquivo, assim um sensor só pode estar presente em uma única célula.

4.3.3 Servidor de operação remota

Nas duas próximas seções são apresentadas as funcionalidades do Servidor de operação remota.

4.3.3.1 Operação remota

A operação remota opera da seguinte forma: o Aplicativo do Servidor de operação remota é ativo, o cliente abre o seu navegador de Internet, requisita uma página HTML armazenada no Servidor de operação remota, o Servidor envia a página ao navegador do cliente e a mesma é disponibilizada para o cliente com os recursos do sistema.

O Servidor de operação remota é um servidor HTTP. O cliente HTTP (navegador de Internet) requisita a página ao Servidor de operação remota e o mesmo envia a página como resposta ao cliente.

Páginas em HTML são estáticas, uma vez solicitadas ao servidor, elas são processadas e o seu conteúdo não é mais atualizado. Ocorrendo qualquer mudança no estado de operação dos nós sensores, o usuário não será notificado.

Para que o processo de notificação seja automatizado, as páginas HTMLs incorporam códigos JavaScript que possibilitam efetuar requisições HTTP ao Servidor. O objeto do tipo *XMLHttpRequest* é criado e configurado no JavaScript, para enviar periodicamente requisições HTTP ao Servidor de operação remota.

Serviços são dispostos no Servidor de operação remota para consumo do objeto do tipo *XMLHttpRequest*. O Servidor formata a resposta no padrão XML indicando as mudanças. E o código JavaScript manipula a resposta enviada pelo Servidor e atualiza a interface gráfica.

Assim, uma vez requisitada à página HTML ao Servidor de operação remota, o usuário será notificado das mudanças ocorridas nos nós sensores. A Figura 4.29 ilustra o funcionamento do processo de verificação de atualização baseado em requisições HTTP. As setas com origem no Aplicativo de operação remota em direção ao Servidor são as requisições. E as setas em sentido contrário são as respostas no formato XML do Servidor ao Aplicativo, o processo só finaliza quando o Aplicativo é fechado.

Figura 4.29 – Processo de verificação de atualização baseado em requisições HTTPs

4.3.3.2 Monitoramento

Uma vez ativo o Servidor de operação remota, automaticamente os serviços de monitoramento são iniciados. São três os serviços de monitoramento são: de alarmes, do estado de operação e do consumo.

Para um melhor entendimento, ao inicializar o serviço de monitoramento os seguintes passos devem ser seguidos: leitura do arquivo de topologia para identificar as células e seus sensores, execução da leitura dos arquivos de descrição para obter os parâmetros equivalentes a cada tipo de serviço de monitoramento e assim executar a monitoração dos parâmetros no servidor *CyberOPC*.

O serviço de monitoramento de alarmes mantém uma base de informação com os sensores alarmados, para mante-la atualizada, solicitações de leitura são executadas periodicamente ao servidor *CyberOPC*. Esta base é compartilhada com o Servidor de operação remota. O Servidor de operação de remota a utiliza, quando recebe uma requisição de serviço da página para verificar se houve mudança dos nós sensores alarmados. Outra funcionalidade do

serviço de monitoramento de alarmes é armazenar as ocorrências de alarme em arquivos XMLs, possibilitando a geração de relatórios.

O serviço de monitoramento de estado de operação é análogo ao serviço de alarmes, os estados de operação dos nós sensores são mantidos em uma base de informação pela leitura periódica e compartilhada com o Servidor de operação remota, que por sua vez a utiliza para manter a interface gráfica atualizada. Ainda, o serviço armazena as mudanças dos estados de operação em arquivos XMLs para a geração de relatórios.

O serviço de monitoramento de consumo é programado para executar a leitura do consumo de energia de todos os nós sensores. A leitura é escalonada ao longo do dia, com nove horários pré-determinados: 09:00, 10:00, 11:00, 12:00, 13:00, 14:00, 15:00, 16:00 e 17:00. Exemplificando, considerando uma rede com noventa nós sensores, nove leituras serão requisitadas a cada horário. As leituras são armazenadas em arquivos XMLs para a geração de gráficos de consumo.

5 Ensaaios e Resultados

Os primeiros resultados obtidos foram utilizando o Simulador de nós sensores, o mesmo auxiliou no desenvolvimento dos recursos do sistema e validação da arquitetura. Alguns testes preliminares de integração entre os projetos correlatos foram executados e são mostrados.

As coordenadas geodésicas utilizadas no trabalho e nos experimentos são da cidade de Mairiporã, estado de São Paulo e foram fornecidas pela concessionária de iluminação pública e energia Elektro.

5.1 Recursos do sistema

Nesta subseção são apresentados todos os recursos disponíveis do sistema como o *CyberOPC*, o simulador, os aplicativo de configuração e topologia, o servidor de operação remota e o aplicativo de operação remota.

5.1.1 *CyberOPC*

O conector *IPGateway* foi desenvolvido e integrado ao *module library* do servidor *CyberOPC*. Através de um cliente *CyberOPC* usual foi possível estabelecer a conexão local e remota entre clientes e o conector disposto no servidor. Assim validando a integração do conector *IPGateway* no servidor *CyberOPC*. A Figura 5.1 ilustra o *IPGateway* integrado ao servidor *CyberOPC*.

No *IPGateway* é possível configurar a porta serial, as configurações de comunicação serial, habilitar os eventos de comunicação, o período de reenvio de mensagens de leituras e Abrir/Fechar a comunicação. Os eventos de comunicação são armazenados no painel direito da interface gráfica.

Como característica, logo ao iniciar a comunicação, todas as variáveis de alarmes e operação são enviadas sequencialmente pela porta serial, o objetivo é obter a condição de inicial de operação do sistema de iluminação. Após a primeira leitura, os valores são mantidos pelas notificações de alarmes gerados pelos próprios nós sensores.

Para um melhor entendimento, supondo que o computador industrial no qual o *CyberOPC* está instalado é desligado e neste intervalo de tempo, uma lâmpada localizada na célula do *CyberOPC* se queima, a notificação deste evento não será recebida pelo *CyberOPC*. Na abordagem desenvolvida, assim que o computador for ligado e o *CyberOPC* for ativado, uma leitura será direcionada ao nó sensor que está com a lâmpada queimada e nova condição será obtida pelo *CyberOPC*.

Figura 5.1 - Conector *IPGateway* integrado ao servidor *CyberOPC*

Uma característica implementada no conector *IPGateway* é o controle de leitura para casos de não respostas dos nós sensores. O controle de leitura funciona da seguinte forma: ao enviar uma requisição aos nós sensores, um limite máximo de tempo é estabelecido para o aguardo da resposta da requisição, caso a resposta não chegue dentro do limite de tempo

estipulado, uma nova requisição é enviada. Sendo o limite de tempo configurável no conector *IPGateway* (Reenvio de mensagens de leituras), o valor adotado/configurado para reenvio de novas requisições é de 2 minutos.

5.1.2 Simulador

O simulador juntamente com o *CyberOPC* representam uma célula na arquitetura proposta. O objetivo é simular as condições de operação de um sistema de iluminação pública, como ligar/desligar lâmpadas em horários pré-estipulados, atender requisições de leitura/escrita do *CyberOPC* e gerar alarmes. A motivação para o desenvolvimento de um simulador foi a não disponibilidade de nós sensores físicos para realização de testes com o sistema.

A simulação de operação do sistema de iluminação pública ocorre da seguinte forma: o Aplicativo de Configuração / Topologia gera arquivos de simulação para cada célula, este arquivo é utilizado pelo simulador da célula para obter a lista de nós sensores que estão disponíveis para simulação. O arquivo é importado, a lista de nós é apresentada ao usuário, o mesmo escolhe os sensores que serão simulados e inicia-se a simulação. Opcionalmente, o usuário pode inicializar os valores de simulação dos seguintes parâmetros: potência da lâmpada, tempo em operação e limite de operação.

A base de tempo da simulação é o ciclo do dia. Este é a representação de um ciclo de um único dia, ou seja, o simulador irá acender e apagar a lâmpada neste ciclo e é configurado em segundos para possibilitar simulações com maior frequência. A simulação é contínua e só se finaliza mediante a requisição do usuário.

A cada ciclo de simulação, os parâmetros internos dos nós sensores em simulação são modificados. O valor do parâmetro de consumo de energia é calculado e atualizado, o valor do parâmetro tempo de operação é incrementado em doze horas e os alarmes tempo de operação da lâmpada e de lâmpada queimada são averiguados.

O parâmetro que representa o alarme tempo de operação da lâmpada excedido é ativado, caso o valor do parâmetro tempo de operação supere o valor do parâmetro limite de operação estabelecido. O parâmetro que apresenta o alarme lâmpada queimada é habilitado, quando o valor do parâmetro tempo de operação for superior a 10% do limite de operação. Caso um alarme seja ativado o mesmo deve ser transmitido ao *CyberOPC* automaticamente.

A Figura 5.2 apresenta o Aplicativo de Simulação de nós sensores. A interface gráfica do Simulador é dividida em dois painéis, um que trata da simulação e o outro da comunicação serial.

No painel de Simulação é possível configurar o tipo de simulação, o ciclo de um dia, o atraso da rede e a taxa de entrega. Adicionalmente, pode-se especificar o início e o fim da Alvorada e do Ocaso. Por fim, uma lista ilustra os nós sensores da célula.

A frequência da simulação pode ser configurada em tempo real quando se deseja simular o ciclo completo de um dia completo (vinte e quatro horas) ou especificando o ciclo de um dia em segundos.

Atraso e taxa de entrega da rede são parâmetros que simulam condições de uma rede sem fio, o protocolo GGPSR não é simulado. O atraso é configurado em segundos e simula as condições de latência de uma rede sem fio. O simulador só envia as respostas para requisições depois de ultrapassado o período estipulado.

A taxa de entrega é especificada em percentagem e simula pacotes de requisições que são perdidos ao longo da rede sem fio e que não são entregues ao seu destino final. Por exemplo, uma taxa de entrega de 90% indica que a cada 100 pedidos, 90 terão respostas e os outros 10 não. No simulador as requisições são negadas aleatoriamente. Esta característica permite o teste de comportamento do sistema perante a não entrega de requisições.

Ainda, para cada nó sensor, o simulador atribui aleatoriamente os horários de desligamento e acionamento da lâmpada. A faixa de horários da alvorada limita os horários de

desligamento das lâmpadas. Ao adicionar um nó sensor para simulação, o horário de desligamento do nó sensor é estipulado aleatoriamente dentro da faixa de início e fim da alvorada. A mesma regra vale para o acionamento da lâmpada, o horário é estipulado aleatoriamente pelo simulador, obedecendo à faixa de valores de início e fim do ocaso.

No painel de comunicação serial os recursos configuráveis são: a porta serial, as propriedades da comunicação e Abrir/Fechar a comunicação com a porta serial. E uma lista de eventos de comunicação é apresentada.

Figura 5.2 – Aplicativo de Simulação de nós sensores

Outro recurso disponível no simulador é a possibilidade de modificação dos valores dos parâmetros dos nós sensores em simulação. Ao selecionar um nó sensor a partir da lista de nós sensores da célula, outra lista é apresentada. Essa lista é composta pelos parâmetros, provenientes do arquivo de descrição, e os valores calculados e mantidos pelo simulador para o nó sensor selecionado.

O usuário pode selecionar um parâmetro na lista, modificar o valor e submeter à alteração ao simulador. A Figura 5.3 ilustra a lista de parâmetros disponíveis para edição de um sensor.

Index	Variável	Label	Acesso	Tipo	Valor traduzido
1	OperationMode	Modo de operação	RW	Enumerated	Automático
2	LampDimmerValue	Luminosidade da lâmpada	RW	Float	100 %
3	LampOperationTimeLimit	Tempo limite de operação	RW	Float	10000 h
4	LampOperationTimeCounter	Contador do tempo de operação	RW	Float	97,36 h
5	BatteryLevel	Nível da bateria	RO	Integer	5 V
6	BatteryLevelAlarms	Alarme do nível da bateria	RO	Enumerated	Boa
7	LampOperationAlarms	Alarmes das lâmpadas	RO	BitEnumerated	Operação normal;
8	LampConsumption	Consumo da lâmpada	RO	Float	38,944 Kwh
9	SerialNumber	Número serial	RO	Integer	2783
10	LampPower	Potência da lâmpada	RO	Float	400 W
11	LampOperationStatus	Estado de operação da lâmpada	RO	Enumerated	Acesa
12	ManualStartTime	Ligar às (manual)	RW	Time	18:00
13	ManualTurnOffTime	Desligar às (manual)	RW	Time	07:00

Figura 5.3 – Aplicativo de Simulação, lista dos parâmetros de um nó sensor

Depois de desenvolvido o simulador era necessário disponibilizar um canal de comunicação entre o *CyberOPC* e o simulador. Para integrar o simulador com o *CyberOPC*, uma conexão de *loopback* virtual é criada. A conexão permite que informações sejam trocadas entre portas seriais bidirecionalmente, ou seja, solicitações de leituras/escritas provenientes do *CyberOPC* são transmitidas para o simulador e alarmes gerados no simulador são transferidos para o *CyberOPC*.

A Figura 5.4 mostra o aplicativo *Null-modem emulator* (2012) utilizado para criar a conexão de *loopback* entre portas seriais. No lado esquerdo do aplicativo é mostrado um par de portas virtuais “COM1” e “COM2” conectadas em *loopback*. Com o caminho bidirecional estabelecido entre o Simulador e o servidor *CyberOPC* foi possível transmitir e receber pacotes entre as partes. Assim, foi possível desenvolver e testar o protocolo de comunicação entre o simulador e o conector *IPGateway*. Outro objetivo desta atividade foi criar um ponto de

integração, pois uma vez disponível a solução do sistema de telecomando para gestão de iluminação pública, o simulador pode ser substituído pelos equipamentos físicos.

Figura 5.4 – Aplicativo *Null-modem emulator* (2012) utilizado para criar a conexão de *loopback* entre portas seriais

A Figura 5.5 ilustra o Simulador comunicando através da porta serial “COM1”. As mensagens recebidas são provenientes do *CyberOPC* conectado a porta serial “COM2”, como apresenta a Figura 5.6.

Para mostrar a troca de pacotes entre o Simulador e o *CyberOPC*, o pacote “01 E8 87 BA C1 D1 62 3A C2 07 0B 98 00” é recebido pelo Simulador as 23h52min55seg, identificado com a sigla “[In]”, como mostra a Figura 5.5, é uma requisição de leitura individual, pois o primeiro *byte* é marcado com “01”. A origem da requisição é mostrada na Figura 5.6, no mesmo horário as 23h52min55seg, identificado com a sigla “[Out]” o pacote é transferido do *CyberOPC*. Assim, requisições de leitura individuais, leituras em grupo e requisição de escrita com origem do *CyberOPC* em direção ao Simulador é validado.

O pacote “05 ED 8D BA C1 1C 56 3A C2 01 04 00 00 09 00 60 12” é enviado pelo simulador as 23h52min57seg, identificado com a sigla “[Out]”, como ilustra a Figura 5.5. A

recepção do pacote pelo conector *IPGateway* do *CyberOPC* é ilustrado na Figura 5.6, no mesmo horário as 23h52min57seg, identificado com a sigla “[In]”. Validando a comunicação bidirecional entre ambos.

Figura 5.5 – Simulador recebendo solicitações de serviços e respondendo aos mesmos

Figura 5.6 – *CyberOPC* recebendo solicitações de serviços e respondendo aos mesmos

5.1.3 Aplicativo de configuração e topologia

A motivação para a criação do aplicativo era automatizar a manipulação do arquivo de topologia conforme descrito na seção 4.3. O aplicativo possibilita ao usuário abrir, salvar, editar, criar e apagar arquivos de topologia, além de permitir a criação das células com seus respectivos nós sensores.

Dois arquivos de descrição foram elaborados conforme a seção 4.1.1 para este teste, sendo um arquivo de descrição do controlador de comunicação e o outro do nó sensor. O componente *XMLLibrary* é aplicado e os modelos de sensores e controladores são disponibilizados ao aplicativo em forma de lista.

Para adicionar novos sensores ou controladores é necessário selecionar o dispositivo na lista de modelos, informar as coordenadas geodésicas e o identificador, clicar no botão adicionar e assim o novo dispositivo é agregado ao arquivo de topologia. A Figura 5.7 apresenta o Aplicativo de configuração e topologia desenvolvido.

Figura 5.7 - Aplicativo de configuração e topologia

A partir da lista o usuário é capaz de abrir um modelo e visualizar a descrição do dispositivo graficamente. Esta visualização é composta pelos agrupamentos de alarmes, parâmetros de processos e grupos IHM. Cada agrupamento ainda lista os seus parâmetros. A Figura 5.8 apresenta graficamente o sensor descrito para teste.

Variável	Label	Acesso	Tipo	Ajuda
LampActivity	Estado de operação	RW	Enumerated	Controla a atividade da lâmpada
LampDimmerValue	Luminosidade da lampada	RW	Float	Controla a luminosidade da lampada
LampOperationTimeLimit	Tempo limite de operação	RW	Integer	Define o limite do alarme de tempo de...
LampOperationTimeCounter	Contador do tempo de o...	RW	Float	Indica o tempo de operação da lâmpa...
BatteryLevel	Nível da bateria	RO	Integer	Indica o nível da bateria (0-5V)
BatteryLevelAlarms	Alarme do nível da bateria	RO	Enumerated	Alarme de nível da bateria
LampOperationAlarms	Alarmes da lâmpada	RO	BitEnumerated	Estado geral de operação da lampada
LampConsumption	Consumo da lâmpada	RO	Float	Consumo da lâmpada

Figura 5.8 – Descrição gráfica de um modelo de sensor

5.1.4 Servidor de operação remota

O funcionamento do Servidor se dá da seguinte forma: o usuário configura a porta HTTP que o cliente irá se conectar, seleciona o arquivo de topologia que será utilizado para monitoração, define o intervalo de monitoração dos serviços de monitoração e ativa o funcionamento do Servidor clicando no botão Iniciar. O Servidor de operação remota é apresentado na Figura 5.9.

Figura 5.9 –Servidor de operação remota

Depois de ativado o Servidor de operação remota, requisições HTTPs podem ser requisitadas na porta especificada. Uma página foi desenvolvida com os requisitos propostos deste trabalho e disponibilizada no Servidor.

A Figura 5.10 apresenta como é feito o acesso à página no Servidor, o endereço é digitado no navegador de Internet e a página é aberta com os recursos do sistema. Sendo “localhost”, o endereço do computador, neste caso o Servidor de operação remota e o navegador de Internet estão no mesmo computador, “9090” a porta HTTP configurada no servidor e “MapaAlarmesGoogleV3.htm” a página HTML requisitada.

Figura 5.10 –Acesso da página de operação do sistema ao Servidor de operação remota

A arquitetura deste trabalho prevê o acesso ao Servidor de operação remota através da Internet. Para validar tal característica, o Servidor de operação remota foi instalado em um servidor na USP e o mesmo foi disponibilizado na Internet. A Figura 5.11 apresenta o Servidor de operação remota ativo no Laboratório de Automação Industrial da USP, São Carlos - SP.

Figura 5.11 – Acesso remoto ao Servidor de operação remota ativo no Laboratório de Automação Industrial na USP, São Carlos - SP

A Figura 5.12 apresenta o acesso via Internet da página de operação do sistema ao Servidor de operação remota instalado na USP. Sendo “143.107.235.5”, o endereço do computador, a porta HTTP é a 80 (omitida pelo navegador de Internet) e “MapaAlarmesGoogleV3.htm” é a página requisitada.

Figura 5.12 – Acesso via Internet da página de operação do sistema ao Servidor de operação remota instalado na USP

No Servidor de operação remota é previsto dispor serviços para consumo do objeto do tipo *XMLHttpRequest*. O serviço “getAlarms.xml” é disposto no Servidor de operação remota e ciclicamente a página requisita o serviço.

A Figura 5.13 ilustra o navegador de Internet em modo de depuração, indicando as múltiplas requisições executadas pela página ao Servidor de operação ao serviço “getAlarms.xml”. O intervalo entre cada requisição é de vinte segundos.

Figura 5.13 – Navegador de Internet em modo de depuração, indicando as múltiplas requisições do serviço “getAlarms.xml” executadas pela página ao Servidor de operação remota

A Figura 5.14 apresenta o navegador de Internet em modo de depuração, mostrando detalhes de uma requisição HTTP ao serviço “getAlarms.xml”.

Figura 5.14 – Navegador de Internet em modo de depuração, mostrando detalhes de uma requisição HTTP ao serviço “getAlarms.xml”

A Figura 5.15 mostra o navegador de Internet em modo de depuração, mostrando o XML de resposta à solicitação ao serviço “getAlarms.xml”. O XML é manipulado pela página e verifica-se a necessidade de atualização da interface gráfica.

Figura 5.15 – Navegador de Internet em modo de depuração, mostrando o XML de resposta à solicitação ao serviço “getAlarms.xml”

Ao receber a requisição HTTP do serviço “getAlarms.xml”, a mesma é indicada pelo Servidor de operação remota no seu painel de eventos. A Figura 5.16 ilustra a recepção de três requisições provenientes do navegador. O intervalo de tempo entre cada uma delas é de vinte segundos. Assim, o funcionamento do processo de verificação de atualização baseado em requisições HTTP entre o navegador de Internet e Servidor de operação remota é validada.

Figura 5.16 – Indicação dos eventos de requisições HTTP recebidos no Servidor de operação remota enviados pela página HTML

5.1.5 Aplicativo de operação remota

Como apresentado na seção anterior, uma página “MapaAlarmesGoogleV3.htm” é disponibilizada para a operação do sistema de iluminação pública.

A página é organizada em duas partes, um painel à esquerda com o mapa da cidade posicionando os nós sensores e no painel à direita com os menus que irão ser explorados um a um ao decorrer desta seção. A Figura 5.17 apresenta o Aplicativo de operação remota.

Figura 5.17 – Aplicativo de operação remota

5.1.5.1 Alarme

As funcionalidades de alarmes são subdivididas em três requisitos: alarmes ativos, filtro e relatórios. Os alarmes ativos são mostrados em duas formas: em listas e posicionados no mapa. A lista é mostrada na Figura 5.18. No exemplo da figura, o nó sensor identificado com número “1” possui dois alarmes ativos.

Cada item de diagnóstico possui uma severidade atrelada ao mesmo, com o objetivo de facilitar a visualização dos itens de diagnósticos no mapa, uma tabela de cores é adotada, assim cada nível de severidade possui uma cor correspondente. A Tabela 5.1 apresenta o nível de severidade com a sua cor correspondente. Na Figura 5.18, o nó sensor é representado pelo ícone amarelo, representando o alarme de maior severidade entre os alarmes ativos.

Severidade	Cor
1
2
3

Tabela 5.1 – Nível de severidade do alarme com a sua cor correspondente.

Figura 5.18 – Vistas de alarmes

O usuário ainda poderá configurar a visualização de alarmes, ocultando alarmes indesejados. Todos os itens de alarmes descritos no arquivo XML de descrição são mostrados ao usuário, como mostra a Figura 5.19.

Figura 5.19 – Filtro de alarmes

Todas as ocorrências de alarmes são armazenadas no Servidor de operação remota, e são disponibilizados em forma de lista na página de operação. O usuário deve selecionar o dia inicial e final para obter os eventos de alarme do período. O serviço “getAlarmsReport.xml” é chamado no Servidor de operação remota e a lista de ocorrências é mostrada. A Figura 5.20 ilustra todos os eventos de alarmes ocorridos entre os dias 16/12/2012 e 18/12/2012, para um sistema de iluminação pública em simulação.

Figura 5.20 – Histórico de alarmes

5.1.5.2 Estado de operação

O estado de operação dos nós sensores é disponibilizado no submenu “Estado de operação das lâmpadas” do menu Operação. Tal submenu só está disponível por ter sido descrito no arquivo de descrição do nó sensor. São dois os estados de operação descritos, “Apagada” e “Acesa” e para cada estado um ícone é atrelado. O trabalho apresenta duas formas de visualização dos estados de operação m forma de lista e a outra posicionando os nós sensores no mapa. Ambas as visualizações são mostradas na Figura 5.21.

Figura 5.21 – Menu de operação com o estado de operação das lâmpadas do Aplicativo de operação remota

O usuário ainda poderá configurar a visualização dos estados de operação, como mostra a Figura 5.22. Ao retirar a seleção das lâmpadas apagadas as mesmas são removidas de ambas as vistas. É apenas uma forma de configuração da visualização, o alarme permanece ativo.

Figura 5.22 – Visualização somente das lâmpadas acesas

5.1.5.3 Operação individual

A operação individual se dá, quando o usuário seleciona no mapa um nó sensor e todos os parâmetros são apresentados com seus respectivos valores. Os parâmetros que possibilitam escrita, podem ser alterados e submetidos ao Servidor de operação remota, que transmite o pedido ao *CyberOPC*. A Figura 5.23 apresenta a lista dos parâmetros do nó sensor de identificação “5528”. O serviço “getMonitorAsset.xml” disposto no Servidor de operação remota notifica as alterações dos valores da supervisão individual. Os valores apresentados na figura são meramente ilustrativos.

Figura 5.23 – Menu de operação individual do Aplicativo de operação remota

5.1.5.4 Operação em grupo

A opção de Operação em grupo é disponibilizada no submenu Grupo do menu Operação.

A Figura 5.24 apresenta o menu de Operação e o submenu Grupo ativados.

Figura 5.24 – Menu de operação em grupo do Aplicativo de operação remota

O círculo e o retângulo são as formas geométricas disponíveis no aplicativo. Para adicionar a figura geométrica basta clicar no botão Adicionar da figura correspondente. Assim a figura é adicionada ao mapa. A manipulação das figuras é um recurso disponível na biblioteca de mapas do Google maps (2010).

Ao adicionar a figura geométrica no mapa, uma lista é apresentada ao usuário com os parâmetros de escrita dos modelos disponíveis na biblioteca de modelos.

O usuário deve posicionar a figura compreendendo os nós sensores no qual deseja submeter à configuração em grupo, selecionar o parâmetro, configurar o valor desejado do parâmetro e enviar o comando. A Figura 5.25 apresenta o retângulo desenhado no mapa e a lista de parâmetros de escrita.

Figura 5.25 – Operação em grupo com retângulo

A Figura 5.26 mostra o círculo desenhado no mapa e o questionamento se deseja alterar a configuração dos nós sensores da região para o modo automático. O usuário confirmando a operação, o comando é enviado para o Servidor de operação remota através do serviço “writeInBatch.xml” que repassa a solicitação ao *CyberOPC*.

A dificuldade encontrada durante o desenvolvimento deste requisito era definir pra qual servidor *CyberOPC*, o comando deve ser enviado, pois o comando por região pode compreender nós sensores de diferentes células. Por enquanto, foi definido que o comando por região será enviado para o controlador de comunicação mais próximo da região.

Figura 5.26 – Operação em grupo com círculo

5.1.5.5 Consumo de energia

O consumo de energia é medido e armazenado no Servidor de operação remota e visualizado na página HTML. O menu de Consumo dispõe os gráficos de consumo ao usuário. Existem duas formas de visualização, uma vista de consumo diário, mostrada na Figura 5.27 e a outra consolidada por mês, apresentada na Figura 5.28. Para traçar o gráfico, o Servidor de operação remota dispõe do serviço “getConsumption.xml”, que formata um XML de resposta com os dados do gráfico.

Figura 5.27 – Gráfico de consumo diário do mês de dezembro de 2012

Figura 5.28 – Gráfico de consumo mensal do ano de 2012

5.1.5.6 Suportabilidade

Esta funcionalidade foi validada nos demais requisitos apresentados nesta seção, a suportabilidade é utilizada por quase todos os componentes do sistema. No total, foram descritos três arquivos modelos de nós sensores, um para os primeiros testes com o simulador, outro para o controlador de comunicação e o último para o teste piloto que será apresentado.

Além dos arquivos, o trabalho foi facilitado pelo componente de software *XMLLibrary*, pois várias mudanças estruturais foram implementadas nos arquivos de descrição ao longo do projeto. Com a biblioteca, a cada nova funcionalidade adicionada na especificação, o componente era estendido. Centralizando a manipulação dos arquivos em único lugar, facilitou o desenvolvimento da suportabilidade em todos os aplicativos clientes da biblioteca.

5.2 Integrações com o controlador de comunicação e os nós sensores

reais

O teste piloto foi configurado com dois computadores. No primeiro foi instalado o sistema com toda a infraestrutura para estabelecer a comunicação com o controlador de comunicação instalado na porta serial, o sistema de operação remota e supervisão de iluminação pública. Para a integração foi desenvolvido um arquivo XML de descrição para o modelo do nó sensor e um arquivo de topologia composto por apenas uma célula e os três nós sensores.

O modelo de nó sensor desenvolvido possui somente dois parâmetros: *LampOperation* e *LampConsumption*. O primeiro é utilizado para apagar e acender um LED, simulando uma lâmpada e o segundo é utilizado para indicar o consumo do nó sensor.

Já no segundo computador, foi instalado o dispositivo “MC1322 USB Dongle” da empresa *FreeScale* acoplado a porta USB, com a capacidade de capturar mensagens que trafegam na rede sem fio, entre o controlador de comunicação e os nós sensores. O software *Wireshark* (2012) também é instalado para visualizar as capturas executadas.

O objetivo do dispositivo de captura é certificar que um comando requerido pelo sistema é transmitido para o nó sensor destino através do controlador de comunicação, e também se o nó sensor responde ao comando de requisição.

O endereço adotado para o controlador de comunicação foi “1”. Os nós sensores foram configurados com os endereços “5”, “6” e “7”. A Figura 5.29 apresenta a configuração do experimento para o teste de integração de comunicação entre o sistema de operação remota e supervisão de iluminação pública e o controlador de comunicação e os nós sensores reais.

Figura 5.29 – Configuração do experimento para o teste de integração de comunicação entre o sistema e os nós sensores reais

Ao ativar o sistema, algumas requisições de leituras são enviadas ao controlador de comunicação e esse por sua vez retransmite a rede de sensores sem fio, neste momento foi possível visualizar através das capturas a solicitação do controlador em direção ao nó sensor de destino com endereço “7”.

Na Figura 5.30 é possível visualizar na primeira linha, o pacote enviado pelo controlador com endereço “1” para a rede (o endereço esta disponível no terceiro *byte* na coluna *Source*). Logo em seguida o nó sensor com endereço “7” disponibiliza a resposta, e assim sucessivamente solicitações são feitas pelo controlador a algum nó sensor e os mesmos respondem. Em alguns casos é possível visualizar duas respostas seguidas a uma requisição, é um mecanismo de retransmissão do nó sensor.

Figura 5.30 – Software *Wireshark* (2012) dispondo das capturas executadas durante a integração

O produto final da integração é mostrado na Figura 5.31, com os nós sensores posicionados no mapa e a indicação dos seus estados de operação em forma de lista. Ainda foi possível ligar e desligar o LED que simulava a lâmpada, através da operação individual. O

requisito de medição do consumo não foi testado, pois tal característica não foi plenamente implementado pelo nó sensor.

Figura 5.31 – Integração do sistema de operação remota e supervisão de iluminação pública com dispositivos físicos

5.3 Comportamento do sistema perante latência e a não entrega de leituras e escritas

As características de latência e não entrega das leituras e/ou escritas estão presentes em sistemas de iluminação pública, pois as mesmas adotam as redes de sensores sem fio. O objetivo dos testes é averiguar o comportamento do sistema de operação e supervisão proposto perante tais condições, além da detecção de possíveis problemas nos requisitos propostos.

Para executar os testes propostos, os resultados em simulação obtidos por Pantoni (2011) com uma rede com 1000 nós sensores são adotados. Os resultados apresentados foram 80% (pior caso) para taxa de entrega e 35 segundos em média para a latência em para tráfego divergente. O tráfego divergente é definido como as requisições de leituras e escritas por parte do sistema com destino aos nós sensores. Ainda, é apresentada a latência para tráfego convergente. O tráfego convergente é definido como envio de notificações dos nós sensores em direção ao controlador de comunicação. O atraso médio apresentado no pior caso, de transmissões simultâneas, é de no máximo 110 segundos com 200 nós sensores transmitindo simultaneamente e 70 segundos para transmissões não simultâneas. O protocolo GGPSR adota estratégias de roteamento para entrega das mensagens de forma independente.

Segundo o estudo de Pantoni (2011), o valor de latência atende aos requisitos especificados na RFC 5548 (2009) na ordem de dezenas de segundos e a taxa de entrega é satisfatória quando comparada aos trabalhos correlatos na literatura.

O aplicativo de simulação simula as condições de latência do tráfego divergente e a taxa de entrega referente à rede de sensores sem fio. Entende-se que a simulação da latência do tráfego convergente é específico do protocolo e não é o objetivo deste trabalho se aprofundar em todas as características do protocolo GGPSR. Porém, é notório que a latência do tráfego convergente deve ser testada para validar a solução por completo e para isso é sugerido que exista uma integração futura entre o simulador apresentado e o simulador do protocolo GGPSR.

Para o experimento, o simulador foi configurado com 1018 nós sensores representando uma célula, com taxa de entrega com 80% e latência de 35 segundos, assim o comportamento dos requisitos do sistema foram avaliados. Os resultados são mostrados nas próximas seções. A Figura 5.32 ilustra as configurações para o experimento do aplicativo simulador.

The screenshot shows a software window titled 'Simulador'. It contains a 'Simulação' (Simulation) section with various input fields and buttons. Below this is a table with simulation data and two buttons at the bottom: '[Selecionar todos]' and '[Limpar todos]'.

Simulação Configuration:

- ☒ Tempo real
- Ciclo de um dia (s): 1200
- Atraso da rede (s): 35
- Taxa de entrega (%): 80
- Alvorada Início: 06:00, Fim: 06:50
- Ocaso Início: 18:00, Fim: 18:50
- Nó sensor: Em operação 0 h, Limite operação 10000 h, Potência 400 W
- Buttons: Iniciar, Parar

Simulation Data Table:

Simulação	Sensor	Mod.	Alvorada	Ocaso	x	y
☒	2783	0101	6:03	18:41	-23,32566	-46,59472
☒	2785	0101	6:49	18:23	-23,32564	-46,59212
☒	2786	0101	6:49	18:48	-23,32563	-46,59084
☒	2787	0101	6:40	18:24	-23,32562	-46,59311
☒	2788	0101	6:36	18:15	-23,32561	-46,59651
☒	2790	0101	6:25	18:26	-23,32555	-46,59637
☒	2791	0101	6:27	18:21	-23,32554	-46,59051
☒	2792	0101	6:20	18:28	-23,32553	-46,59678
☒	2793	0101	6:10	18:09	-23,3255	-46,59399

Buttons: [Selecionar todos], [Limpar todos]

Figura 5.32 – Configurações do simulador para o experimento do comportamento do sistema perante latência e a não entrega de leituras e escritas

5.3.1 Desempenho da partida para obtenção do estado inicial de operação e alarmes de uma célula

Este teste indica o desempenho da leitura inicial das condições de operação e alarmes de uma célula, nas configurações ilustradas na Figura 5.32. Duas variáveis são correlacionadas neste teste o número de parâmetros de alarme/operação e também se a partida ocorre no período do ocaso ou da alvorada.

O número de parâmetros de alarme e operação tem ligação direta com o desempenho, pois quanto maior o número de parâmetros, maior será o número de requisições de leituras aos nós sensores.

No caso da partida ocorrer nos períodos de ocaso ou da alvorada, o desempenho é otimizado porque automaticamente alarmes são enviados nestes períodos, indicando o estado de alarme/operação. Um alarme proveniente do simulador em direção ao *CyberOPC* indica em um único pacote as informações do estado de operação e também das condições de alarmes, assim poupando a necessidade de envio de comandos de leituras para averiguar tais condições. Para exemplificar, uma porção de lâmpadas irá ligar num determinado horário e os alarmes são gerados para cada nó sensor correspondentes. Ao receber os alarmes o *CyberOPC* não irá solicitar leituras para os nós sensores que já enviaram tais notificações de alarmes, otimizando o fluxo de solicitações.

Além do desempenho, três resultados foram medidos. O número de requisições enviadas, o número de pacotes recebidos e o número de requisições negadas. Foram executados quatro testes, a Tabela 5.2 resume o experimento com os cenários dos testes e os resultados.

No primeiro e no segundo testes, os nós sensores foram descritos com dois parâmetros de alarme e um de operação, totalizando três parâmetros de leitura por sensor. A diferença entre os dois testes é que o primeiro foi executado fora dos períodos do ocaso e da alvorada. Comparando os dois testes, o resultado do desempenho obtido no segundo teste foi superior ao do primeiro, pois a partida no segundo teste foi aproximadamente de 45 minutos e no primeiro por volta de 1 hora e 30 minutos. Nota-se que no segundo teste o número de requisições enviadas é inferior ao do primeiro, indicando a influencia das notificações de alarmes no processo, pois para cada notificação recebida três leituras são evitadas.

Nos testes três e quatro, os nós sensores foram descritos com apenas um parâmetro de alarme e o teste três foi executado fora dos períodos de ocaso/alvorada. O teste de número de três

foi sete minutos mais lento comparados com o número quatro, indicando a influência das notificações de alarmes nos testes.

Número do teste	Parâmetros de operação/alarme por sensor	Durante o Ocaso ou Alvorada	Número de requisições enviadas	Número de pacotes recebidos	Número de requisições negadas	Tempo total de partida (hora:minuto:segundo)
1	3	Não	3820	3054	766	01:27:26
2	3	Sim	1808	2398	361	00:44:49
3	1	Não	1274	1018	256	00:32:06
4	1	Sim	986	1315	197	00:25:04

Tabela 5.2 – Cenários e resultados dos testes executados para obtenção do tempo de partida de uma célula

O desempenho dos testes é ilustrado graficamente na Figura 5.33.

Figura 5.33 – Resumo dos resultados obtidos no teste de partida de uma célula

Conclui-se que o desempenho da partida está relacionado diretamente ao número de parâmetros de alarmes e de operação descritos no nó sensor, quando comparados os conjuntos de testes um e dois com o conjunto três e quatro, sendo o segundo mais eficiente. Para um melhor

desempenho é sugerido um novo serviço no protocolo de comunicação entre o controlador de comunicação e o *CyberOPC*. Este novo serviço, enviaria somente uma requisição e como resposta a este serviço uma notificação de alarmes seria enviada com todas as informações de alarmes/estado de operação do nó sensor. Assim, nos testes um e dois, apenas um comando por nó sensor seria enviado e não três, diminuindo drasticamente o número de requisições aos nós sensores. Acredita-se que com este novo serviço, o desempenho nos testes um e dois seriam iguais ou superiores aos obtidos nos testes três e quatro.

Quando a partida ocorre no período do ocaso ou da alvorada, os resultados de desempenho da partida são melhores, pois as notificações dos alarmes tornam o processo de leitura mais efetivo. Ainda, entende-se que os resultados de desempenho obtidos são de vital importância para os operadores do sistema, pois no caso de reestabelecimento de uma célula, o tempo deve ser conhecido.

5.3.2 Desempenho de leitura e escrita na operação individual

O teste de operação individual foi executado para averiguar o desempenho do requisito nas condições de latência e taxa de entrega, sendo o teste subdividido em dois, a supervisão e a escrita. A Tabela 5.3 resume os resultados dos testes executados para este requisito.

No teste de supervisão, ao selecionar um nó sensor foi tomado o tempo total até que todos os valores dos 13 parâmetros fossem totalmente preenchidos na interface gráfica, obtendo-se uma média de 4 minutos e 50 segundos em quatro amostras.

Teste	Média (minuto:segundo)	Melhor caso (minuto:segundo)	Pior caso (minuto:segundo)	Amostras
Supervisão	4:50	3:07	5:14	4
Escrita	2:47	2:15	3:52	4

Tabela 5.3 – Desempenho do requisito operação individual perante a latência e taxa de entrega

O comportamento do sistema perante a latência e não entrega de requisições, referente à supervisão, foi bem sucedido, pois todos os valores em todas as quatro amostras foram obtidos em função da resposta do simulador.

O aplicativo de operação remota apresenta durante o processo de supervisão o progresso, indicando valores lidos e não lidos. A Figura 5.34 destaca dois parâmetros (Índices 6 e 7) ainda não lidos e os demais já disponíveis ao usuário, sendo a indicação de “*Reading...*” usada para marcar os parâmetros não lidos.

Figura 5.34 – Indicação de progresso da supervisão, indicando valores lidos e não lidos

No teste de escrita, ao submeter à escrita de um valor ao parâmetro foi tomado o tempo total até que o novo valor fosse atualizado na interface gráfica, obtendo-se uma média de 2 minutos e 47 segundos.

Conclui-se que o sistema se comporta adequadamente perante a latência e não entrega de pacotes quando processa as leituras, pois independente do tempo que os pedidos são executados pelo simulador, as informações são disponibilizadas ao operador. No mecanismo de escrita não existe a confirmação do processamento do pedido pelo nó sensor, provocando dúvidas se a solicitação foi entregue ou não. A solução de contorno encontrada para este problema é o envio de solicitações de leituras para verificação do novo valor do parâmetro a cada 1 minuto, tornando o requisito de escrita funcional nas condições de não entrega. Porém, até mesmo os pedidos de leituras podem não ser entregues, o que impacta o desempenho, assim é sugerida como melhoria futura que seja adicionado uma resposta ao serviço de escrita para aperfeiçoar o desempenho.

5.4 Validação da arquitetura

Para a validação da arquitetura baseada em células, o experimento foi configurado conforme a Figura 5.35 com três computadores. Duas células em simulação foram criadas, uma com 976 nós sensores e a outra com 255 nós sensores, totalizando 1231 nós sensores, sendo cada célula representada por um computador. No terceiro computador, o Servidor de operação remota foi instalado e ativado, assim a supervisão e operação do sistema foram disponibilizadas aos operadores. Para a divisão em duas células, uma topologia com dois controladores de comunicação foi estabelecida.

Figura 5.35 – Configuração do experimento para validação da arquitetura

Após configurar o experimento e ativar as células, o primeiro teste foi averiguar a comunicação entre o Servidor de operação remota e as células, para isso um cliente *CyberOPC* usual disposto no servidor foi utilizado, para envios de comandos de leitura para as células. A Figura 5.36 apresenta o cliente *CyberOPC* usual configurado para acesso ao valor de um parâmetro na célula 1, sendo (1) o endereço do computador da célula 1, (2) o certificado utilizado para autenticação entre cliente e servidor *CyberOPC*, (3) a porta de conexão do servidor do servidor *CyberOPC*, (4) o parâmetro que será requisitado a leitura, “IPGateway.2368.11”, sendo IPGateway o conector que será dirigida a leitura, 2368 o identificador do nó sensor e 11 o índice do parâmetro e (5) é a resposta ao comando de leitura.

A resposta é composta pelos campos *ReceiveTime* (horário que o servidor *CyberOPC* recebeu o pedido), *ReplyTime* (horário que o servidor respondeu ao pedido), *ServerState* (indica o estado de operação do servidor *CyberOPC*), *ItemID* (composição do identificador do nó sensor

e o parâmetro separados por ponto), *TimeStamp* (indica o horário da última atualização do parâmetro) e *Value* (valor do parâmetro).

Figura 5.36 – Cliente *CyberOPC* usual disposto no Servidor de operação remota enviando comando de leitura a célula 1

A comunicação entre a célula 1 e o Servidor de operação remota foi validada, na Figura 5.37 o servidor *CyberOPC* disposto na célula 1 recebe o pedido de leitura, e a resposta ao pedido é mostrado na Figura 5.36 no cliente *CyberOPC* disposto no Servidor de operação remota. Nota-se que os valores dos campos mostrados no servidor *CyberOPC* são exatamente os valores dos campos mostrados no cliente *CyberOPC*, assim validando a comunicação entre ambos.

Figura 5.37 – Servidor *CyberOPC* disposto na célula 1 respondendo ao comando de leitura solicitado pelo cliente usual *CyberOPC* disposto no Servidor de operação remota

A Figura 5.38 mostra o cliente *CyberOPC* disposto no Servidor de operação remota enviando comando de leitura a célula 2.

Figura 5.38 – Cliente *CyberOPC* usual disposto no Servidor de operação remota enviando comando de leitura a célula 2

A comunicação entre a célula 2 e o Servidor de operação remota foi validada, na Figura 5.39 o servidor *CyberOPC* disposto na célula 2 recebe o pedido de leitura, e a resposta ao pedido é mostrado na Figura 5.38 no cliente *CyberOPC* disposto no Servidor de operação remota. Nota-se que os valores dos campos mostrados no servidor *CyberOPC* são exatamente os valores dos campos mostrados no cliente *CyberOPC*, assim validando a comunicação entre ambos.

Figura 5.39 – Servidor *CyberOPC* disposto na célula 2 respondendo ao comando de leitura solicitado pelo cliente usual *CyberOPC* disposto no Servidor de operação remota

Após a validação da comunicação entre o Servidor de operação remota e os computadores das células, o próximo passo foi ativar o aplicativo de operação remota para disponibilizar a operação e supervisão do sistema remotamente. A página foi acessada e o resultado é mostrado na Figura 5.40. A figura ilustra os dois conjuntos de nós sensores, cada qual pertencentes às células 1 e 2.

Figura 5.40 – Experimento com duas células apresentado no mapa pelo Aplicativo de operação remota

Neste teste conclui-se que a arquitetura adotada foi validada, permitindo que inúmeras células possam ser adicionadas para atender grandes áreas urbanas para iluminação pública.

6 Considerações finais e conclusões

Características de propostas acadêmicas e soluções de mercado foram pesquisadas e analisadas, e concluiu-se que a solução proposta atende a todos os requisitos para operação e supervisão de um sistema de iluminação pública. Dentre as propostas estudadas, nenhuma delas apresenta a possibilidade de integração de nós sensores utilizando arquivo do tipo XML para suportabilidade de nós sensores, o que torna uma contribuição deste trabalho.

Disponibilizar a operação do sistema de operação remota e supervisão de iluminação pública em aplicativos através de navegadores de Internet e estabelecer a comunicação entre os nós sensores simulados e físicos e o sistema foram os principais resultados obtidos, além da operação individual, operação em grupo, medição do consumo de energia, recepção de alarmes e a indicação da operação.

O simulador permitiu a criação de um cenário próximo a um sistema de iluminação pública, onde foi possível medir o desempenho e averiguar pontos de melhorias. O desempenho foi medido para o cenário de uma célula com 1018 nós sensores. Caso o desempenho apresentado não seja adequado para operação de uma célula, o número de nós sensores por célula pode ser reduzido. A arquitetura proposta permite tal condição, pois inúmeras células podem ser criadas até a cobertura plena de um município.

O teste piloto mostrou a capacidade de integração de nós sensores à solução, pois todos os recursos descritos no arquivo XML de descrição foram disponibilizados à operação, permitindo ao operador, por exemplo, apagar/acender o LED disposto no nó sensor simulando o acionamento/desligamento de uma lâmpada. O que comprova a contribuição deste trabalho, pois fabricantes de nós sensores podem descrever os seus nós sensores utilizando arquivos do tipo

XML. Ainda foi possível no teste validar o padrão de comunicação estabelecido entre o controlador de comunicação e o sistema.

A arquitetura proposta foi validada, permitindo que células possam ser criadas para atender grandes áreas urbanas para iluminação pública e o sistema apresentado é viável, pois os resultados validam a metodologia e arquitetura proposta.

6.1 Publicações

- Fonseca, Cleber; Brandão, Dennis. Sistema de controle e monitoramento de iluminação pública. In: XIX Congresso Brasileiro de Automática, 2012, Campina Grande - PB. Sistema de controle e monitoramento de iluminação pública, 2012.
- Pantoni, Rodrigo; Fonseca, Cleber; Brandão, Dennis. Street Lighting System Based on Wireless Sensor Networks. In: Moustafa Eissa. (Org.). (Org.). Street Lighting System Based on Wireless Sensor Networks. 01ed.Rijeka - Croatia: InTech - Open Access Publisher, 2012, v. , p. 337-352.
- Fonseca, Cleber; Pantoni, Rodrigo; Brandão, Dennis. Aplicativo de Controle e Monitoramento para Sistemas de Iluminação Pública. In: XVIII Congresso Brasileiro de Automática, 2010, Bonito/MS. Aplicativo de Controle e Monitoramento para Sistemas de Iluminação Pública, 2010.

6.2 Trabalhos futuros

A partir dos resultados do trabalho, nota-se a possibilidade da evolução, adição de novas funcionalidades e propostas de trabalhos futuros, destacando-se:

- Aplicar o protocolo HTTPS no Servidor de operação remota e adicionar um mecanismo de autenticação para acesso à página de operação do sistema;
- Módulo de manutenção com a capacidade de traçar rotas orientadas pelos pontos de iluminação defeituosos, com a possibilidade de sugerir o material requerido para efetuar a intervenção em cada ponto de iluminação;
- Desenhar no mapa as regiões que estão com falta de energia;

- Adicionar um novo serviço ao protocolo de comunicação, que enviaria somente uma requisição ao nó sensor e como resposta a este serviço uma notificação de alarme seria enviada com todas as informações de alarmes/estado de operação do sensor, para diminuir o número de requisições de leituras para obter as condições iniciais dos nós sensores de uma célula;
- Adicionar ao serviço de escrita individual e em grupo a resposta de confirmação do processamento da escrita;
- Integração entre o simulador apresentado e o simulador do protocolo GGPSR de Pantoni (2011);
- Integração com banco de dados;

Referências bibliográficas

Ackerman, W.J.; Block, W.R. (1992). Understanding supervisory systems. IEEE Computer Applications in Power, Vol. 5, Issue 4, pp. 37-40

Agência Nacional de Energia Elétrica - ANEEL. Resolução Normativa N° 414: Estabelece as Condições Gerais de Fornecimento de Energia Elétrica de forma atualizada e consolidada, de 9 de Setembro de 2010. Disponível em: < <http://www.aneel.gov.br/cedoc/bren2010414.pdf> >. Acesso em 01 nov. 2011.

Atici, Ç., Özçelebi, T., Lukkien, J.J. (2011). System architecture for road lighting. Consumer Electronics (ICCE), 2011 IEEE International Conference on, pp.681-682, 9-12 Jan. 2011

Bing Maps (2011). Bing Maps APIs. Disponível em: <<http://msdn.microsoft.com/en-us/library/dd877180.aspx>>. Acesso em: Acesso em 13 set. 2011

C# (2006). Standard ECMA-334 C# Language Specification. Disponível em: <<http://www.ecma-international.org/publications/standards/Ecma-334.htm>>. Acesso em 10 fev. 2011.

Chen, Y., Liu, Z. (2009). Distributed Intelligent City Street Lamp Monitoring and Control System Based on Wireless Communication chip nRF401. Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC '09. International Conference on , vol.2, pp.278-281, 25-26 April 2009

Central Software - Streetlight Intelligence Inc (2008). Lumen IQ™ Central. Disponível em: <<http://www.streetlightiq.com/Products/Overview/NetworkedLampControl/CentralSoftware/tabid/112/Default.aspx>>. Acesso em 20 ago. 2011

Chunguo, J., Dongmei, S., Deying, G. (2007). Design of Streetlight Monitoring and Control System Based on Wireless Sensor Networks. Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on, pp.57-62, 23-25 May 2007

Confederação Nacional de Municípios – CNM (2012). Iluminação Pública. Disponível em: < <http://www.cnm.org.br/iluminacao/> >. Acesso: 18 dez. 2012

CyberOPC (2010). CyberOPC JSON-RPC-DA 1.0 Specification. Disponível em: < https://docs.google.com/Doc?docid=0AcheE_w0kX9uZGh0Z2Y4czlfMmZndzRibmNu&hl=en >. Acesso em 18 fev. 2011

Denardin, G.W., Barriquello, C.H., Campos, A., Do Prado, R.N. (2009). An intelligent system for street lighting monitoring and control. Power Electronics Conference, 2009. COBEP '09. Brazilian , vol., no., pp.274-278, Sept. 27 2009-Oct. 1 2009

Denysyuk, P.; Teslyuk, V.; Khimich, I.; Farmaga, I. (2007). XML application for microfluidic devices description. CAD Systems in Microelectronics, 2007. CADSM '07. 9th International Conference - The Experience of Designing and Applications of, pp.567-569, 19-24 Feb. 2007

Duncan, B.K. (2009). Relay parameterization made easy with a universal XML interface. Power Systems Conference, 2009. PSC '09., pp.1-6, 10-13 March 2009

EDDL (2011). White paper - EDDL Makes System Life-Cycle Management Easier - Keeping system up-to-date. Disponível em : <http://www.eddl.org/SiteCollectionDocuments/TechPapers/wp_EDDL%20easy%20admin.pdf>. Acesso 10 ago. 2011

EpiSensor (2009). Sica Studio Software. Disponível em: <<http://www.episensor.com/products/software/sica-studio>>. Acesso em 10 ago. 2011

Faure, D.; Vanderdonckt, J. (2010). User interface eXtensible markup language. EICS'10 - Proceedings of the 2010 ACM SIGCHI Symposium on Engineering Interactive Computing Systems , pp. 361-362

Fonseca, Cleber; Brandão, Dennis. (2012) Sistema de controle e monitoramento de iluminação pública. In: XIX Congresso Brasileiro de Automática - CBA 2012, 2012, Campina Grande – PB.

Fonseca, C. C.; Pantoni, R. P.; Brandão, D. (2010). Aplicativo de controle e monitoramento para sistemas de iluminação pública. In: CONGRESSO BRASILEIRO DE AUTOMÁTICA - CBA 2010, 2010, Bonito – MS

Google maps (2010). API do Google Maps. Disponível em: <<http://code.google.com/intl/pt-BR/apis/maps/documentation/javascript/>>. Acesso em 13 set. 2011

Guerrero-Garcia, J.; Gonzalez-Calleros, J.M.; Vanderdonckt, J.; Muoz-Arteaga, J. (2009). A Theoretical Survey of User Interface Description Languages: Preliminary Results. Web Congress, 2009. LE-WEB '09. Latin American, pp.36-43, 9-11 Nov. 2009

Hong, S.-I.; Lyu, H.-S.; Yoon, D.-H.; Lin, C.-H. (2011). Development of road light gateway with sensor network. International Conference on Advanced Communication Technology, ICACT , pp. 839-842

Iordache, C., Gavat, S., Mada, C., Stanciu, D., Holban, C. (2008). Streetlight monitoring and control system Part I: System structure. Automation, Quality and Testing, Robotics, 2008. AQTR 2008. IEEE International Conference on , vol.3, pp.183-186, 22-25 May 2008

JavaScript (2011). JavaScript Reference. Disponível em: <<https://developer.mozilla.org/en/JavaScript/Reference>>. Acesso em 18 fev. 2011

Jiang, Y.; Jiang, W.; Ren, H. (2010). Research on XML-based network lighting device description approach. Computer Science and Education (ICCSE), 2010 5th International Conference on, pp.990-995, 24-27 Aug. 2010

Lavric, A.; Popa, V.; Males, C.; Finis, I. (2012). "A performance study of ZigBee wireless sensors network topologies for street lighting control systems," Mobile and Wireless Networking (iCOST), 2012 International Conference on Selected Topics in , vol., no., pp.130-133, 2-4 July 2012

- Leccese, F.; Leonowicz, Z. (2012), "Intelligent wireless street lighting system," Environment and Electrical Engineering (EEEIC), 2012 11th International Conference on , vol., no., pp.958-961, 18-25 May 2012
- Lee, J.D.; Nam, K.Y.; Jeong, S.H.; Choi, S.B.; Ryoo, H.S.; Kim, D.K. (2006). Development of Zigbee based Street Light Control System. Power Systems Conference and Exposition, 2006. PSCE '06. 2006 IEEE PES, pp.2236-2240, Oct. 29 2006-Nov. 1 2006
- Li, Q.; SHu, Y.; Tao, C.; Wang, L. (2008). Device Common Resource Description and Management with CIM in Industry. Computer and Electrical Engineering, 2008. ICCEE 2008. International Conference on, pp.459-464, 20-22 Dec. 2008
- Lobashov, M., Sauter, T. (2006). Vertical Communication from the Enterprise Level to the Factory Floor - Integrating Fieldbus and IP-based Networks. Emerging Technologies and Factory Automation, 2006. ETFA '06. IEEE Conference on , pp.1214-1221, 20-22 Sept. 2006
- Long, L., Xiumin, C., Yong, W., Qing, W. (2009). The Development of Road Lighting Intelligent Control System Based on Wireless Network Control. Electronic Computer Technology, 2009 International Conference on, pp.353-357, 20-22 Feb. 2009
- Louise, J. (2011). Smartening up the city. New Electronics, United Kington, Volume 44, Issue 2, 25 January 2011, Pages 14-16
- Mendalka, M.; Gadaj, M.; Kulas, L.; Nyka, K. (2010). WSN for intelligent street lighting system. 2010 2nd International Conference on Information Technology, ICIT 2010, pp. 99-100
- Mori, G.; Paternò, F.; Spano, L.D. (2008). Exploiting web services and model-based user interfaces for multi-device access to home applications. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5136 LNCS, pp. 181-193
- The National Institute of Standards and Technology – NIST. (2012). Standards Information Resources from NIST and SGIP. Disponível em: <http://www.nist.gov/smartgrid/standards_info_resources.cfm>. Acesso em 04 jan. 2012
- Null-modem emulator (2012). Null-modem emulator. Disponível em: <<http://sourceforge.net/projects/com0com/>>. Acesso em 15 abr. 2011
- OPC-DA (1998). OPC Overview White Paper. Disponível em: <<http://www.opcfoundation.org/Archive/2a83366b-da1c-490a-99d2-f321e2d4ce51/General/OPC%20Overview%201.00.pdf>>. Acesso em 30 ago. 2011
- OPC XML-DA (2004). OPC XMLDA 1.01 Specification. Disponível em: <<http://www.opcfoundation.org/Downloads.aspx?CM=1&CN=KEY&CI=278&CU=55>>. Acesso em 30 ago. 2011
- Pantoni, R. P. (2011). Estratégia de roteamento e de controle de acesso ao meio para redes de sensores sem fio destinadas a redes urbanas. Tese (Doutorado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2011.

Pantoni, R.P.; Brandao, D. (2009). Developing and implementing an open and non-proprietary device description for FOUNDATION fieldbus based on software standards. *Computer Standards & Interfaces*, vol. 31, Issue 2, pp.504-514

Pantoni, Rodrigo ; Fonseca, Cleber; Brandao, Dennis (2012). Street Lighting System Based on Wireless Sensor Networks. In: Moustafa Eissa. (Org.). (Org.). Street Lighting System Based on Wireless Sensor Networks. 01ed.Rijeka - Croatia: InTech - Open Access Publisher, 2012, v. , p. 337-352.

Programa PROCEL RELUZ (2000). Programa Nacional de Iluminação Pública e Sinalização Semafórica Eficientes. Disponível em: <

RFC 5548 (2009). Routing Requirements for Urban Low-Power and Lossy Networks. Disponível em: <<http://tools.ietf.org/html/rfc5548>> Acesso em 10 dez. 2012.

Roe, S.A. (2010). Ajax, XSLT and SVG: Displaying ATLAS conditions data with new web technologies. *Journal of Physics: Conference Series* 219 (1 PART 4), art. no. 042020

Rose, I., Welsh, M. (2010). Mapping the urban wireless landscape with Argos. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys '10)*. ACM, New York, NY, USA, 323-336

Shu, Y.-Z.; Tao, C.-G.; Quan, L.; Xiaohong, P. (2009). Research of Optimizing Device Description Technology Based on XML in EPA. *Electronic Commerce and Security*, 2009. ISECS '09. Second International Symposium on , vol.1, pp.561-564, 22-24 May 2009

Strategic Telemetry (2009). SmartLights 2.0. Disponível em: <<http://www.strategictelemetry.net/OurProducts.aspx>>. Acesso em 01 set. 2011

Streetlight.Vision (2009). Streetlight.Vision Monitoring Software. Disponível em: <<http://www.streetlightmonitoring.com/slvwebsite/index.php/slvsolution>>. Acesso em 07 set. 2011

Suzuki, K.; Ideguchi, T. (2009). Proposal of device capability profiling for manufacturing application service interface. *Industrial Technology*, 2009. ICIT 2009. IEEE International Conference on, pp.1-6, 10-13 Feb. 2009

TerraLib (2010). TerraLib project. Disponível em: <<http://www.terralib.org/docs/papers/cap12.pdf>>. Acesso em 12 jan. 2010

Torrise, N.M. (2011). Monitoring Services for Industrial. *Industrial Electronics Magazine*, IEEE , vol.5, no.1, pp.49-60, March 2011

Utterback, T.; Tritschler, M. (2007). Getting a fix on street lighting assets. *Lighting Design and Application: LD and A*, vol.37, pp.53-55

Web Services (2002). Web Services White Paper. Disponível em: <<http://www.medialab.sonera.fi/workspace/WebServicesWhitePaper.pdf>>. Acesso em 04 abr. 2011

Wei, Z.; Ni, X.; Jia, D. (2010). Research on software reuse of user interface for mobile computing devices based on XML. Software Engineering and Service Sciences (ICSESS), 2010 IEEE International Conference on, pp.146-149, 16-18 July 2010

Wireshark (2012). Wireshark. Disponível em: <<http://www.wireshark.org>>. Acesso em 19 dez. 2012.

XMLHttpRequest (2010). W3C Candidate Recommendation. Disponível em: <<http://www.w3.org/TR/XMLHttpRequest/>>. Acesso 04 ago. 2011

Yahoo! Maps (2011). Yahoo! Maps Web Services - AJAX API Getting Started Guide. Disponível em: <<http://developer.yahoo.com/maps/>>. Acesso em: Acesso em 13 set. 2011

Yao, Y., Zhang, D., Wang, C., Sun, Y., Xu, D. (2006). Design of Remote Street Lamp Monitoring and Communicating Node Controller Based on Spread Spectrum Carrier. Industrial Electronics, 2006 IEEE International Symposium on , vol.3, pp.1642-1645, 9-13 July 2006

Zotos, N.; Stergiopoulos, C.; Anastasopoulos, K.; Bogdos, G.; Pallis, E.; Skianis, C. (2012). "Case study of a dimmable outdoor lighting system with intelligent management and remote control," Telecommunications and Multimedia (TEMU), 2012 International Conference on , vol., no., pp.43-48, July 30 2012-Aug. 1 2012