

Jessica Maciel de Castro

Refinamento de Políticas para Auxiliar no Gerenciamento de Dispositivos IoT em Hospitais Inteligentes



UNIVERSIDADE FEDERAL DA PARAÍBA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

João Pessoa
2019

Refinamento de Políticas para Auxiliar no Gerenciamento de Dispositivos IoT em Hospitais Inteligentes

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal da Paraíba, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Computação Distribuída

Orientador: Fernando Menezes Matos

Coorientador: Eudisley Gomes dos Anjos

João Pessoa

2019

Catálogo na publicação
Seção de Catalogação e Classificação

C355r Castro, Jessica Maciel de.

Refinamento de Políticas para Auxiliar no Gerenciamento
de Dispositivos IoT em Hospitais Inteligentes / Jessica
Maciel de Castro. - João Pessoa, 2019.
68 f. : il.

Orientação: Fernando Menezes Matos.

Coorientação: Eudisley Gomes dos Anjos.

Dissertação (Mestrado) - UFPB/CI.

1. Internet das Coisas. 2. Hospital Inteligente. 3.
Gerenciamento de Política. 4. Refinamento de Política.
I. Menezes Matos, Fernando. II. Gomes dos Anjos,
Eudisley. III. Título.

UFPB/BC



UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA



Ata da Sessão Pública de Defesa de Dissertação de
Mestrado de Jessica Maciel De Castro, candidato ao título de
Mestre em Informática na Área de Sistemas de Computação,
realizada em 28 de agosto de 2019.

1 Aos vinte e oito dias do mês de agosto, do ano de dois mil e dezenove, às dezesseis horas,
2 no Centro de Informática da Universidade Federal da Paraíba, em Mangabeira, reuniram-se
3 os membros da Banca Examinadora constituída para julgar o Trabalho Final da Sr^a. Jessica
4 Maciel de Castro, vinculada a esta Universidade sob a matrícula nº 20171006180, candidata
5 ao grau de Mestre em Informática, na área de "Sistemas de Computação", na linha de
6 pesquisa "Computação Distribuída", do Programa de Pós-Graduação em Informática, da
7 Universidade Federal da Paraíba. A comissão examinadora foi composta pelos professores:
8 Fernando Menezes Matos (PPGI-UFPB) Orientador e Presidente da Banca, Gustavo
9 Henrique Matos Bezerra Motta (PPGI-UFPB), Examinador Interno, Eudisley Gomes dos
10 Anjos (UFPB), Examinador Externo ao Programa, Adriano Fiorese (UEL), Examinador
11 Externo à Instituição. Dando início aos trabalhos, o Presidente da Banca cumprimentou os
12 presentes, comunicou aos mesmos a finalidade da reunião e passou à palavra a candidata
13 para que a mesma fizesse a exposição oral do trabalho de dissertação intitulado:
14 "Refinamento de Políticas para Auxiliar no Gerenciamento de Dispositivos IoT em Hospitais
15 Inteligentes". Concluída a exposição, a candidata foi arguida pela Banca Examinadora que
16 emitiu o seguinte parecer: "**aprovada**". Do ocorrido, eu, Claurton de Albuquerque Siebra,
17 Coordenador do Programa de Pós-Graduação em Informática, lavrei a presente ata que vai
18 assinada por mim e pelos membros da banca examinadora. João Pessoa, 28 de agosto de
19 2019.

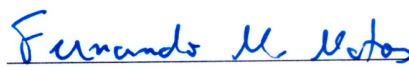
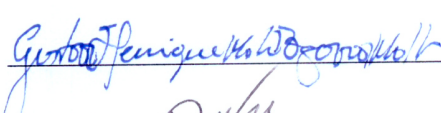

Prof. Dr. Claurton de Albuquerque Siebra

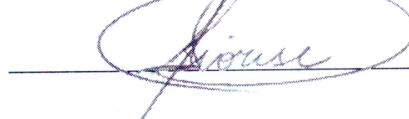
Prof. Dr. Fernando Menezes Matos
Orientador (PPGI-UFPB)

Prof. Dr. Gustavo Henrique Matos Bezerra Motta
Examinador Interno (PPGI-UFPB)

Prof. Dr. Eudisley Gomes dos Anjos
Examinador Externo ao Programa (UFPB)

Prof. Dr. Adriano Fiorese
Examinador Externo à Instituição (UEL)



*“Por vezes sentimos que aquilo que fazemos não é senão uma gota de água no mar.
Mas o mar seria menor se lhe faltasse uma gota.”
(Madre Teresa de Calcutá)*

Agradecimentos

Agradeço aos meus orientadores Dr. Fernando Menezes Matos e Dr. Eudisley Gomes dos Anjos por toda paciência ao me guiarem e orientarem no desenvolvimento deste trabalho. Sempre de forma sábia e amigável.

Agradeço a minha família, em especial a minha mãe que me deu suporte quando necessário e me ensinou o valor da educação, sempre se esforçando para que eu possa aprender e crescer.

Gostaria também de agradecer a minha namorada, Stephanie Soares, pelo companheirismo e apoio em todos os momentos durante esta etapa e pela força que me dá sempre para continuar. Obrigada também pelos desenhos de hospital inteligente que fez para que eu pudesse ilustrar minha dissertação.

Minha gratidão aos meus amigos que me acompanham e estão dispostos a me ajudar sempre que preciso. Ao meu querido amigo Iury Araújo que me deu suporte sempre quando preciso, principalmente nos momentos de angústia. E ao meu amigo Luiz Fernando pelo apoio e ajuda no desenvolvimento deste trabalho.

Por fim, gostaria de agradecer ao órgão de fomento a pesquisa, CAPES, ao apoio financeiro prestado para realização deste trabalho.

Resumo

O uso de uma rede de dispositivos que constituem a Internet das Coisas já é uma realidade nas mais diversas áreas de aplicação. No contexto de hospitais inteligentes diversos tipos de dispositivos têm sido utilizados para proporcionar uma melhor experiência para os seus usuários, nomeadamente pacientes e funcionários. O Uso de IoT permite uma melhoria nos serviços prestados e com isso melhor atendimento aos pacientes, agilizando os processos internos dos hospitais. Contudo, a customização dos dispositivos pertencentes à uma rede IoT ainda é uma etapa que pode dificultar, ou até mesmo impossibilitar, o uso da mesma. Essa característica fica ainda mais evidente em ambientes com mudanças constantes e modificações críticas, como hospitais. Neste caso, se o sistema não for facilmente configurável para as possíveis mudanças de políticas e requisitos, a forma de execução de serviços do hospital pode ficar defasada e perder a sua principal utilidade, dificultando ainda mais o trabalho dos funcionários. Portanto, sempre que houver uma mudança de regras de funcionamento do hospital, principalmente na execução de atividades críticas, é preciso que o sistema seja facilmente modificado levando aos dispositivos IoT da rede a se adaptarem às novas regras. Por este motivo, o presente trabalho propõe uma solução para facilitar o gerenciamento de dispositivos IoT em hospitais inteligentes utilizando refinamento de políticas, permitindo que os usuários criem e gerenciem as mesmas através de linguagens de programação mais abstratas e que não requerem um vasto conhecimento de tecnologias de programação. A solução proposta foi implementada e testada, alcançando resultados favoráveis para aplicações em um cenário hospitalar.

Palavras-chave: Internet das Coisas, Hospital Inteligente, Gerenciamento de Política, Refinamento de Política.

Abstract

The use of a network of devices that constitute the Internet of Things is already a reality in many application areas. In the Smart Hospital context, various types of devices have been used to provide a better experience for their users, as patients and staff. The use of IoT allows an improvement in the services provided, a better patient care and streamline hospital internal processes. However, the customization of devices belonging to one IoT network is still a step that may make it difficult, or even impossible, the use of it. This feature is even more evident in environments with constant change and critical modifications, such as hospitals. In this case, if the system is not easily configurable for policy and requirements changes, the hospital's services may become outdated and lose its main utility. That's make staff work even more difficult. Therefore, whenever there is a change in hospital operating rules, especially when performing critical activities, the system needs to be easily modified to the IoT devices to adapt to the new rules. For this reason, this work proposes a solution to facilitate IoT device management in smart hospitals using policy refinement. So, it allow users to create and manage them through more abstract programming languages that do not require extensive knowledge of technologies. The proposed solution was implemented and tested, achieving favorable results for the hospital scenario.

Keywords: Internet of Things, Smart Hospital, Policy management, Policy Refinement.

Lista de ilustrações

Figura 1 – Níveis de Camadas - Adaptado de (RIEKSTIN et al., 2016).	27
Figura 2 – Requisitos de Refinamento.	27
Figura 3 – Política para identificação de movimentação.	29
Figura 4 – Categorias de Refinamento de Política	30
Figura 5 – Hospital Inteligente (Autoria: Stephanie Soares)	32
Figura 6 – Visão Geral da Arquitetura	34
Figura 7 – Camadas de Abstração da Política	35
Figura 8 – Funcionamento da Solução	36
Figura 9 – Modelo de Dados dos Recursos	38
Figura 10 – Modelo de Dados das Políticas	39
Figura 11 – Modelo de Dados das Normas	40
Figura 12 – Modelo do XML	40
Figura 13 – Política de gerenciamento de porta automática	44
Figura 14 – Exemplo de <i>Parse Tree</i>	45
Figura 15 – Exemplo de Arquivo de Configuração de uma Política	46
Figura 16 – Saída do Validador de Normas	49
Figura 17 – Saída do Validador de Recursos	50
Figura 18 – Simulação	51
Figura 19 – Gráfico de Parser	52
Figura 20 – Gráfico de Validação de Normas	52
Figura 21 – Gráfico de Validação de Recursos	53
Figura 22 – Gráfico de Gerador de Arquivo de Configuração	53
Figura 23 – Gráfico de Processo Completo de Refinamento de Política	53
Figura 24 – XSD da Política	63
Figura 25 – Políticas 01 em XML	64
Figura 26 – Políticas 02 em XML	64
Figura 27 – Políticas 03 em XML	64
Figura 28 – Políticas 04 em XML	65

Figura 29 – Políticas 05 em XML	65
Figura 30 – Políticas 06 em XML	66
Figura 31 – Políticas 07 em XML	66
Figura 32 – Políticas 08 em XML	66
Figura 33 – Políticas 09 em XML	67
Figura 34 – Políticas 01 em CFILE	67
Figura 35 – Políticas 02 em CFILE	67
Figura 36 – Políticas 03 em CFILE	67
Figura 37 – Políticas 04 em CFILE	67
Figura 38 – Políticas 05 em CFILE	67
Figura 39 – Políticas 06 em CFILE	68
Figura 40 – Políticas 07 em CFILE	68
Figura 41 – Políticas 08 em CFILE	68

Lista de tabelas

Tabela 1 – Sumário dos Requisitos	28
Tabela 2 – Operadores	41
Tabela 3 – Resultado dos Requisitos	50

Lista de siglas

CFILE Arquivo de Configuração

BDs Bancos de Dados

BD Banco de Dado

IoT Internet of Things

MW *Middleware*

RFID *Radio Frequency Identification Devices*

RSSF Rede de Sensores Sem Fio

TI Tecnologia da Informação

VR Validador de Recursos

VN Validador de Normas

Sumário

1	INTRODUÇÃO	19
1.1	Objetivos	21
1.2	Estrutura da Dissertação	21
2	REFERENCIAL TEÓRICO	23
2.1	Internet das Coisas	23
2.2	Hospital Inteligente	24
2.3	Políticas	25
2.3.1	Refinamento de Políticas	26
2.4	Trabalhos Relacionados	28
3	REFINADOR DE POLÍTICAS	31
3.1	Ambiente Hospitalar	31
3.2	Visão Geral da Arquitetura	33
3.3	Funcionamento da Arquitetura	36
3.4	Modelo de Dados	37
3.5	Política de Gerenciamento	40
4	AVALIAÇÃO	43
4.1	Implementação	43
4.2	Ambiente de Teste	47
4.3	Experimentos e Resultados	48

5	CONCLUSÃO	55
	REFERÊNCIAS	57
	APÊNDICES	61
	APÊNDICE A – POLÍTICAS	63
A.1	Schema XML da Política	63
A.2	Políticas em XML	64
A.3	Políticas em CFILE	65

Introdução

A Internet das Coisas, do inglês Internet of Things (IoT), pode ser definida como um conjunto de dispositivos conectados à uma rede e que interagem entre si. Esses dispositivos podem conter sensores e atuadores embutidos em objetos do dia a dia, como refrigeradores, celulares, lâmpadas, pulseiras inteligentes, entre outros. Eles têm a habilidade de interconectar-se sem intervenção humana e obedecer regras predefinidas (ATZORI; IERA; MORABITO, 2010).

Como forma de melhorar a qualidade de vida dos usuários, o paradigma IoT pode ser implementado em diversos domínios, tais como: casas, escritórios, salas de aula, bancos, ambientes públicos, hospitais, além de inúmeros outros ambientes inteligentes (KAFLE; FUKUSHIMA; HARAI, 2016), (AHMED et al., 2016). Dessa forma, um ambiente inteligente é um meio em que diversos sensores, atuadores e elementos computacionais estão distribuídos e embarcados em objetos conectados à rede (CHEN; THOMAS; HARRISON, 2011). Além disso, eles utilizam diferentes tipos de sensores para implementar funcionalidades variadas, tais quais reconhecimento, monitoramento e gerenciamentos inteligentes. Assim, o ambiente busca trazer uma melhor experiência individual e coletiva dos indivíduos no espaço físico (YU; LU; ZHU, 2012).

Atualmente, a área da saúde e de cuidados com a saúde é uma das mais atrativas para IoT (PANG, 2013). Um hospital inteligente baseado em IoT é construído com diversos sensores, atuadores, tecnologias e aplicações que fornecem serviços inteligentes. Estes serviços integram as funções de diagnósticos, decisão, gerência e tratamento (DHARIWAL; MEHTA, 2017). Para isso, os sensores podem ser embarcados em pacientes, médicos, funcionários clínicos, equipamentos médicos e espaços físicos, para monitorar, perceber, agir e alertar dependendo das configurações previamente definidas (MORAIS et al., 2017).

Para que o ambiente inteligente ofereça a seus usuários as funcionalidades desejadas, os seus dispositivos precisam ser configurados para atuar conforme as necessidades daquele contexto. Caso haja mudanças nas políticas de funcionamento do ambiente, é

preciso reconfigurar os dispositivos para que eles estejam de acordo com as novas regras. Entretanto, reconfigurar esses dispositivos pode ser uma tarefa difícil para pessoas que não possuem o conhecimento técnico em áreas relacionadas à Tecnologia da Informação (TI) (REISINGER; SCHRAMMEL; FRÖHLICH, 2017). Por isso, caso os usuários/gerentes não consigam realizar as mudanças, faz-se necessário a solicitação do auxílio de um técnico, o que pode gerar atrasos e/ou interrupções nos serviços do ambiente, principalmente quando as mudanças são críticas e requerem decisões rápidas. Uma solução encontrada no mercado é o uso de *kits* compostos por dispositivos e sistemas configurados para funcionar com comportamentos pré estabelecidos, dificultando a atualização e mudança de comportamento dos dispositivos caso seja necessário. Outro problema que afeta a adoção de tecnologias IoT, principalmente em hospitais, ocorre devido a dependência de profissionais especializados para configurar e reconfigurar os dispositivos, causando hesitação na adoção de tais tecnologias. Usuários ficam receosos em adotar tecnologias que não compreendem ou não dominam completamente (EDGCOMB; VAHID, 2012).

Hospitais são ambientes críticos, por isso é importante que todos os seus sistemas funcionem com o mínimo de complicações, incluindo os sistemas IoT (ULLAH; FIEDLER; WAC, 2012). Sendo assim, é necessário fornecer ferramentas que facilitem as redefinições de comportamento dos dispositivos IoT pelos usuários. Uma abordagem utilizada para atender a essa necessidade é o uso de políticas. Uma política é uma representação de uma ou mais regras que determinam um ação a partir de um evento. Elas são comumente representadas no formato Evento-Condição-Ação (ECA) e podem ser utilizadas para ditar o comportamento dos dispositivos. O uso de política permite que os usuários possam criar e gerenciar seus próprios dispositivos com mais facilidade sem precisar programar cada um do ambiente (PONTE; GOMES, 2018). Isso traz maior flexibilidade para o sistema e garante que o comportamento dos dispositivos possam ser alterados sempre que necessário (SINGH; BACON, 2014). Com essa abordagem é possível prover aplicações em que os usuários possam criar suas próprias políticas em uma linguagem de programação de alto nível.

Entretanto, para proporcionar a criação de políticas em linguagens de alto nível, é necessário refinar a política criada pelo usuário transformando-a de um alto nível para baixo nível de abstração. Contudo, é importante garantir a consistência das políticas durante todo o processo de refinamento, isto é, é necessário realizar a tradução da política e garantir a validação da mesma de acordo com os recursos utilizados (RIEKSTIN et al., 2016). Além disso, para um hospital inteligente, é importante garantir que as políticas cumpram com as normas definidas pelo hospital e as agências reguladoras nacionais e internacionais, tais como: normas que estabelecem as condições de temperatura e umidade dentro dos ambientes hospitalares (Anvisa, 2003), temperatura da geladeira de medicamentos (Ebserh, 2017), iluminação (ABNT, 1991), entre outras.

Sendo assim, este trabalho apresenta uma solução para auxiliar no gerenciamento de dispositivos IoT usando refinamento de políticas em hospitais inteligentes. O objetivo é prover aos funcionários a autonomia necessária para gerenciar dispositivos de acordo com as mudanças existentes no hospital. Para isso, a solução executa a tradução da política do XML para um arquivo de configuração que pode ser executado pelo *Middleware* (MW). Então, o MW pode utilizar o arquivo para configurar o comportamento dos dispositivos IoT. Além disso, a solução deve garantir a validação das políticas em relação às normas do hospital e recursos disponíveis (CASTRO et al., 2019).

1.1 Objetivos

Como objetivo geral deste projeto, pretende-se proporcionar autonomia no gerenciamento dos dispositivos IoT aos funcionários dos hospitais inteligentes, garantindo que as políticas de gerenciamento sejam traduzidas e validadas de acordo com as normas hospitalares e recursos existentes. Como objetivos específicos, têm-se:

- ❑ Estabelecer uma representação de políticas de configuração de dispositivos. É preciso definir a representação das políticas, ou seja, como elas serão retratadas e armazenadas, permitindo que o *middleware* possa configurar os dispositivos do sistema IoT.
- ❑ Definir mecanismo de tradução e validação de políticas de configuração. Definição do mecanismo que será responsável por realizar o processo de tradução e validação de políticas submetidas pelos usuários, buscando realizar o processo com eficiência satisfatória para o usuário.

1.2 Estrutura da Dissertação

Este trabalho é dividido em cinco capítulos. O Capítulo 2 disserta sobre os termos utilizados e trabalhos relacionados com a pesquisa. O Capítulo 3 apresenta um exemplo de ambiente hospitalar inteligente, a visão geral do modelo proposto, explica o seu funcionamento, o modelo dos dados e o modelo definido da política de gerenciamento. No Capítulo 4 é mostrado como a implementação da proposta foi realizada, contendo a descrição do cenário de teste e os resultados encontrados. Por fim, o Capítulo 5 contém as considerações finais e os trabalhos futuros.

Referencial Teórico

Este capítulo é composto por quatro seções, cujo objetivo é apresentar os conceitos básicos para auxiliar na compreensão do trabalho. A primeira seção traz uma explanação sobre Internet das Coisas, a segunda contextualiza o Hospital Inteligente, a terceira explica o que são políticas, onde atuam e como funciona o refinamento delas. Na quarta e última seção será feita uma descrição dos trabalhos relevantes para esta pesquisa.

2.1 Internet das Coisas

O paradigma de IoT pode ser definido como a interconectividade de diversos tipos de objetos a uma rede, interagindo e cooperando entre si para que possam atingir objetivos em comum (ATZORI; IERA; MORABITO, 2010). Isso se deu devido ao avanço da tecnologia e expansão do acesso à internet, tendo um aumento no uso de dispositivos móveis e serviços ubíquos (BORGIA, 2014). Os objetos são conhecidos como objetos inteligentes ou "coisas" e são definidos por entidades que têm um mínimo de funcionalidade de comunicação, como a capacidade de receber e responder mensagens, possuindo um identificador único e possuir alguns recursos básicos de computação, podendo variar entre os dispositivos. Além disso, eles podem ter a capacidade de sentir fenômenos físicos (sensores) ou realizar ações que tenham efeitos sobre a realidade física (atuadores) (MIORANDI et al., 2012). Eles podem ser de diversos tipos, como geladeiras, televisões, câmeras, robôs, sensores de temperatura, umidade, luzes, entre outros. E, por conta disso, o uso de IoT permite automação para uma grande variedade de áreas (BAKER; XIANG; ATKINSON, 2017).

Na rede de sistemas IoT, existem diversos dispositivos de diversas marcas e tipos. Eles estão conectados, coletando e compartilhando informações do meio que estão inseridos. Essa característica de heterogeneidade dos dispositivos é um aspecto que precisa ser considerado nos sistemas IoT. Para isso, a solução encontrada para tratar a heterogeneidade é a utilização de middleware que fornece uma camada de abstração que age

como um mediador entre as camadas de aplicativos e infraestruturas de rede, ajudando a recuperar os dados dos sensores e comunicá-los aos aplicativos (PERERA et al., 2013). O middleware é comumente descrito como sistema de colagem, pois é crucial para auxiliar na comunicação e no gerenciamento dos sistemas IoT (SINGH; BACON, 2014).

Para a realização deste trabalho, tomou-se como base o middleware apresentado no artigo de (SINGH; BACON, 2014). Esse *middleware* além de apresentar o papel tradicional de permitir a interoperabilidade entre diversos dispositivos e aplicações, ele é pensado para atuar na área de serviços para a saúde. Ele faz uso de políticas para orquestrar e reconfigurar os componentes dos sistemas. Dessa forma, apresenta-se mais ativo ao controlar as interações dos sistemas e dos dispositivos para poder atingir o objetivos de cada usuário. A definição de política é apresentada na Seção 2.3.

2.2 Hospital Inteligente

Um ambiente inteligente é um espaço físico em que as tecnologias, como sensores, atuadores e elementos computacionais, estão embarcadas e conectadas a objetos do dia a dia através de uma rede. Elas adquirem e gerenciam de forma inteligente as informações relevantes dos indivíduos para que possam oferecê-los uma melhor experiência (CHEN; THOMAS; HARRISON, 2011).

Existem diversos ambientes inteligentes, como: prédios que contém aplicações como vigilância de vídeo, gerenciamento de energia, controle de ambiente interno; mobilidade que utiliza de sensores e monitoramento de veículos e vias para transportar o maior número de pessoas de forma mais eficiente. Há também a área de saúde inteligente, que contém ambientes e aplicações voltadas para a saúde e o auxílio da mesma, como hospitais, biossensores, dispositivos e sistemas de emergência e ambulância inteligentes, entre outros (MOHANTY; CHOPPALI; KOUGIANOS, 2016).

Os hospitais inteligentes utilizam de dispositivos e tecnologias, como biossensores, *smartphones*, *Radio Frequency Identification Devices* (RFID) e Rede de Sensores Sem Fio (RSSF), para adquirir informações sobre localizações dos pacientes e funcionários, sinais vitais dos pacientes, monitorar inteligentemente os equipamentos e medicamentos. Ao utilizar tais tecnologias, um sistema inteligente consegue fornecer dados úteis como informações dos pacientes atualizadas apresentadas em painéis, computadores ou no próprio *smartphone* (CHEN et al., 2016). Para que haja a comunicação entre os dispositivos e o oferecimento dos serviços, os hospitais utilizam de IoT. Como afirma (MOHANTY; CHOPPALI; KOUGIANOS, 2016), IoT é a coluna vertebral dos ambientes inteligentes e, junto com *big data*, os fazem mais eficiente e responsivo. Os dispositivos que fazem parte da rede são capazes de se conectarem e interagirem entre si sem a intervenção humana (ATZORI; IERA; MORABITO, 2010).

Desta forma, os hospitais podem usar IoT para manter informações diversas sobre os dispositivos, funcionários e pacientes (CARNAZ; NOGUEIRA, 2016). Há um grande número de sensores para monitorar os pacientes, o ambiente dos hospitais e as condições dos dispositivos médicos, tendo uma vasta variedade deles (ZHANG et al., 2018), como diversos para avaliar diferentes sinais vitais (por exemplo, temperatura, taxa respiratória, pressão sanguínea, ECG, glicose, batimento cardíaco, entre outros) (MORAIS et al., 2017). Alguns exemplos de caso de uso de hospitais inteligentes baseados em IoT são: identificação automática e rastreamento de pessoas e dispositivos médicos em hospitais, o monitoramento do paciente, a correta associação entre droga e paciente e também monitoramento em tempo real de parâmetros fisiológicos para detecção precoce de deterioração clínica (CATARINUCCI et al., 2015).

2.3 Políticas

Uma política define o comportamento desejado dos sistemas e seus componentes. Ela é formada por um conjunto de regras ou metas que são utilizados para gerenciar ou controlar acesso a um conjunto de recursos e serviços de um sistema, como o sistema IoT. Diversos autores dividem as políticas em dois tipos: Políticas de Gerenciamento e Políticas de Autorização. O primeiro tipo estabelece o que deve ser feito dada uma certa condição, enquanto o segundo define o que é permitido ou não, restringindo acesso ao sistema (RIEKSTIN et al., 2016). Uma política é comumente representada no formato de Evento-Condição-Ação (ECA) (SINGH; BACON, 2014).

(SINGH; BACON, 2014) apresenta vantagens da utilização de abordagens orientadas a políticas. Como a redução considerável de carga em desenvolvimento de serviços ou aplicações e a probabilidade de erros. Isso se deve porque os desenvolvedores não necessitam considerar os requisitos individuais dos usuários e não precisam lidar com todos os contextos potenciais. No caso do hospital, cada instituição irá descrever suas próprias políticas de acordo com o seu funcionamento. Desta forma, aumentando o reuso e facilitando a integração com novos componentes. Além disso, a utilização de políticas permite quebrar aplicações de silo, onde as aplicações não conseguem interagir entre elas e se mantém fechadas. Assim, podendo operar através dos limites do sistema, permitindo mais flexibilidade e funcionalidade e o sistema pode ser personalizado e customizado para responder a diferentes circunstâncias.

De acordo com (BOUTABA; AIB, 2007), o uso de políticas na informática começou pela necessidade de melhorar a segurança quando surgiram os computadores compartilhados, sendo utilizadas inicialmente como políticas de segurança em que definia-se regras em alto-nível para regulamentar o controle de acesso dos sistemas. Com o uso de políticas para segurança em controle de acesso, junto com o surgimento do DNS e sistemas

baseados em controle de acesso para sistemas distribuídos, percebeu-se a necessidade de fornecer uma abordagem mais abstrata para gerenciar as políticas. Tal abordagem ficou definida como Gerenciamento Baseado em Política, do inglês *Policy-Based Management* (PBM). Eles são conjuntos de regras para alcançar o gerenciamento escalonável de um sistema distribuído, tendo também como objetivo a otimização de uso de recursos, custo, receita e desempenho. Consequentemente, com o crescimento da internet, foi necessário a utilização de política para impor a Qualidade de Serviço, do inglês *Quality of Service* (QoS), e possibilitar selecionar rotas em ordem de restringir o uso de redes a certas classes de clientes. Paralelamente ao início do uso de políticas para redes ou gerenciamento de redes (PBNM), surgiu a discussão de como tais políticas devem ser representadas apropriadamente. Inicialmente as políticas eram representadas utilizando linguagens em baixo nível, e com o tempo, trabalhos com linguagens de alto nível também foram propostas.

Como o uso de políticas permite sistemas alcançarem os requisitos de escalabilidades, flexibilidade, uniformidade, entre outros (BOUTABA; AIB, 2007), elas se tornaram uma boa alternativa para gerenciar, configurar, programar os dispositivos e sistemas IoT. (SICARI et al., 2017) afirma que a adoção de um mecanismo de aplicação de políticas oferece um controle de acesso flexível e efetivo aos recursos IoT. Outra vantagem na utilização de uma linguagem que gera política é a programação e reprogramação dos dispositivos IoT no ambiente sem a necessidade de mudar seus códigos fontes (PONTE; GOMES, 2018).

2.3.1 Refinamento de Políticas

O processo de tradução de uma política de alto nível para baixo nível é definido como Refinamento de Políticas (RF). Segundo (RIEKSTIN et al., 2016), RF é o processo de tradução de uma política descrita em uma abstração de alto nível (linguagem de operação) para a linguagem de configuração específica do dispositivo. Uma política pode ser visualizada como uma hierarquia de diferentes níveis, de acordo com a forma que ela é expressa, comunicada ou automatizada. (KOCH; KRELL; KRAMER, 1996) define que um mínimo de três camadas são necessários para preencher todo o processo entre o nível mais alto ao mais baixo. É possível visualizar na Figura 1 as definições de camadas baseadas no trabalho (SLOMAN, 1994). Sendo assim, o operador define a política em alto nível e ela precisa passar pelo processo de refinamento, ou seja, de tradução de uma camada para outra até o nível em que a política pode ser interpretada e aplicada pela instância final.

Além da etapa de tradução, é importante garantir a consistência da política durante todo o processo. O trabalho de (RIEKSTIN et al., 2016) apresenta uma *Survey* em métodos de refinamento de políticas para redes sustentáveis. Apesar do trabalho ser focado em cenários de redes de computadores sustentáveis, as definições de requisitos para

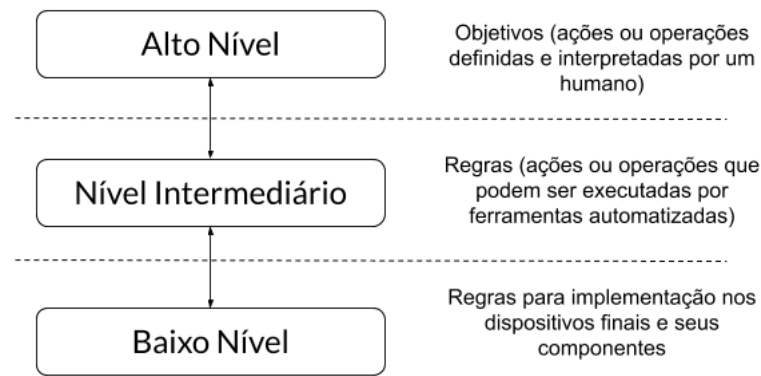


Figura 1 – Níveis de Camadas - Adaptado de (RIEKSTIN et al., 2016).

refinamentos descritas nele serviram como base para a definição do processo de refinamento no contexto de sistemas e dispositivos IoT. Os autores apresentaram sete requisitos que julgaram ser necessários para o cenário de redes sustentáveis, sendo eles: Tradução, Recursos, Cobertura/Verificação, Conflitos, Dinamicidade, Representação de Sustentabilidade e Capacidade de Orquestração. Na Figura 2 são apresentados os requisitos que foram considerados relevantes para este trabalho e para o cenário de IoT. Os requisitos de Tradução, Recursos, Cobertura, Conflitos foram considerados relevantes para este trabalho e para o cenário de IoT. Além desses, um novo requisito chamado Normas foi identificado como necessário para aplicações IoT em Hospitais. Os requisitos circulados na figura foram considerados na implementação deste trabalho.

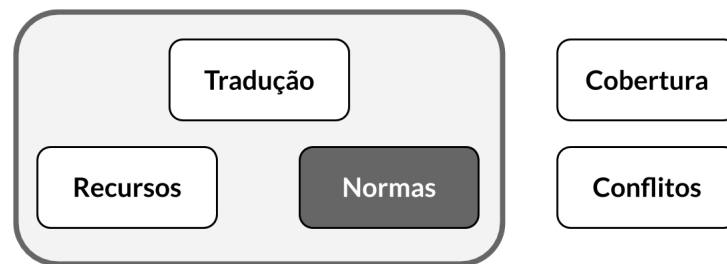


Figura 2 – Requisitos de Refinamento.

A Tabela 1 apresenta o sumário dos requisitos relevantes para ambientes inteligentes e sistemas IoT. O requisito (1) apresenta que é preciso que as etapas de tradução sejam garantidas pelo método de refinamento. Ou seja, a partir de uma política em nível mais alto é preciso aplicar uma tradução que considere a semântica e/ou sintática das políticas para transformá-las em políticas de mais baixo nível. O (2) é a verificação de que todos os recursos que serão utilizados pela política existem, estão disponíveis e podem realizar as tarefas descritas. Por exemplo: se a política utiliza a temperatura de uma determinada sala, é necessário que exista um sensor de temperatura no local e que ele esteja fornecendo os dados necessários. O (3) é a verificação de que a política refinada está de acordo com os requisitos originais. Ou seja, se o que está descrito em alto nível

trata da mesma coisa que a de baixo nível. O (4) é etapa que verifica se existem outras políticas que podem entrar em conflito com a que está sendo refinada, resolvendo-as e garantindo que o sistema continue a funcionar de forma correta. Por fim, o requisito (5) é responsável por garantir que a política esteja obedecendo a todas as regras do hospital. Ou seja, se existe uma norma que estabelece que as temperaturas das salas não podem ser menores que 18 graus Celsius, uma política submetida que tem como ação alterar a temperatura para 10 graus não pode ser aprovada.

Tabela 1 – Sumário dos Requisitos

Requisito	Sumário
(1) Tradução	Necessita ter a(s) etapa(s) que garante(m) a tradução de uma política de alto nível para baixo nível
(2) Recursos	Garantir que durante o processo de transformação os recursos no ambiente do sistema IoT são considerados
(3) Cobertura	Verificar se as política refinada mantém os seus requisitos originais
(4) Conflitos	Lidar com conflitos ao detectá-los e resolvê-los
(5) Normas	Durante o processo de transformação é preciso garantir que as normas do hospital foram consideradas

2.4 Trabalhos Relacionados

Nesta seção serão apresentados alguns dos trabalhos utilizados e considerados relevantes para o desenvolvimento deste trabalho. Eles podem ser divididos em duas categorias: trabalhos voltados para linguagens de programação de ambientes inteligentes e trabalhos para a programação de ambientes inteligentes com uso de políticas.

Desde o início dos anos 2000s são produzidas pesquisas a respeito da facilitação de programação de sistemas inteligentes, como é possível observar nos trabalhos de (JAHNKE; D'ENTREMONT; STIER, 2002) e (HUMBLE et al., 2003). O primeiro busca facilitar a programação de casas inteligentes, apresentando uma solução diferente dos paradigmas tradicionais de programação. Também traz a discussão a respeito de quais paradigmas podem ser utilizados para facilitar a customização, tais como linguagem visual, componente de *software* e programação baseada em conexão, em que são utilizados fluxogramas para realizar as conexões entre os dispositivos inteligentes. Entretanto, nesse trabalho não são realizados testes com usuários, nem há uma apresentação da proposta desenvolvida. O segundo trabalho traz a ideia de quebra cabeça para tentar realizar as programações, tentando representar nas peças um componente da programação. Também não foi realizada nenhuma avaliação.

Os autores do (EDGCOMB; VAHID, 2012) apresentam a MNFL, uma linguagem de fluxo de monitoração e notificação para monitoramento assistido. Eles mencionam que os usuários que não conseguem modificar as tecnologias para suas necessidades acabam

abandonando-as. Então, um sistema é desenvolvido para facilitar a programação para usuários finais leigos em relação à computação, para que eles possam programar e alterar o sistema quando preciso. Para isso, utiliza-se uma linguagem de fluxo organizada por blocos, onde colocam-se as condições que desejam. Como pode ser observado na Figura 3 Foram realizados testes de usabilidade para verificar o sucesso de usuários finais em encontrar e utilizar os blocos existentes necessários. Antes dos testes, os participantes tiveram acesso a um tutorial sobre a linguagem e, para o experimento, receberam os desafios de programação, a partir dos quais foram analisados o tempo necessário para realização e a solução submetida. As soluções são de baixa complexidade, exigindo entre 1-2 blocos. A partir dos resultados, identificou-se que em 50% das vezes as pessoas conseguiram realizar as tarefas pedidas. Verificou-se, a partir disso, que talvez seja necessário outras abordagens de programação para programas mais complexos.

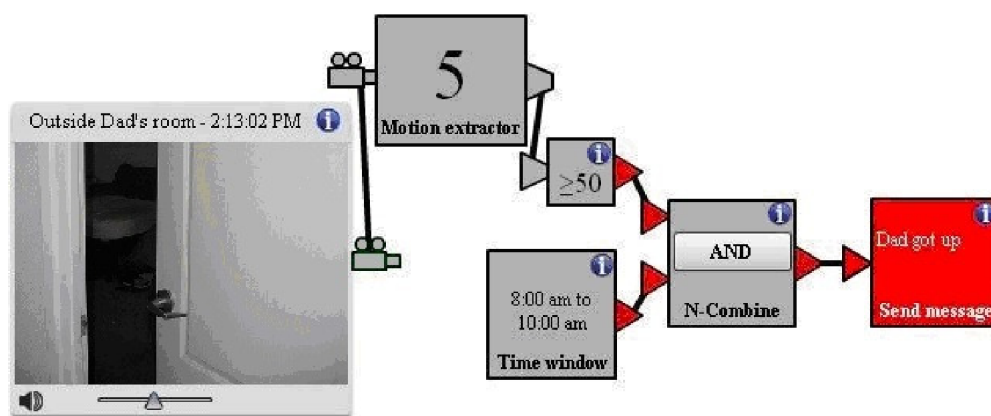


Figura 3 – Política que identifica a movimentação e emite alerta caso seja entre 8 e 10 horas da manhã. Fonte: (EDGCOMB; VAHID, 2012)

No trabalho de (HA; LINDH, 2017) é apresentado um modelo de gerenciamento de desempenho para IoT como um serviço para cuidadores na área de *e-Health*. O modelo é baseado em políticas de baixa complexidade, elas são implementadas com um código de programa que roda no *gateway* e é inspirada por uma abordagem que é construída como evento-condição-ação. No artigo é apresentado o modelo em que um cuidador ou provedor de serviço possa adicionar novas políticas, porém o trabalho é focado em construir políticas para gerenciar as performances dos dispositivos e as políticas são construídas com essa finalidade de análise, como a avaliação sobre a perda de pacotes dos dispositivos.

No artigo (PONTE; GOMES, 2018) os autores apresentam uma linguagem de programação de dispositivos IoT para automação residencial. Eles trazem a arquitetura desenvolvida para isso, bem como testes realizados. Apesar de não utilizar o termo refinamento, eles realizam a tradução e a validação da política, conferindo que os dispositivos utilizados estão disponíveis e podem executar as tarefas impostas. Como é um trabalho para ser implantado em uma residência, eles utilizam uma arquitetura centralizada.

Como os trabalhos da literatura geralmente são voltados para aplicações de sistemas IoT para ambientes residenciais, e nesse contexto não é necessário obedecer normas, eles não exploraram esse tipo de validação.

Por ser limitado o número de trabalhos de refinamento de políticas para ambientes inteligentes ou sistemas IoT, estudou-se os trabalhos relacionados de PBM e PBNM. Desta forma, a Figura 4 apresenta três tipos de abordagens de refinamento de política, tendo como base o trabalho de (RIEKSTIN et al., 2016). Essas abordagens são Transformação baseada em Regra, Refinamento baseado em Classificação e Abordagens baseadas em Lógica. A primeira, Transformação baseada em Regra, traz os trabalhos que mais se assemelham com a proposta deste trabalho, como (CARVALHO et al., 2012) e (VERMA, 2000).

Os autores de (CHEMINOD et al., 2019) apresentam uma abordagem para refinamento e verificação automáticos de políticas de acesso de controle. A verificação de política se assemelha as validações de políticas tratadas neste trabalho. Eles utilizam um modelo de twofold apresentados por eles para realizar o refinamento e verificação das políticas. Apesar do escopo do trabalho ser diferente, como também o processo de refinamento, dessa forma não podendo ser aplicado para o refinamento de políticas em hospitais inteligentes e sistemas IoT, ele foi interessante para verificar formas para avaliar a proposta deste trabalho.

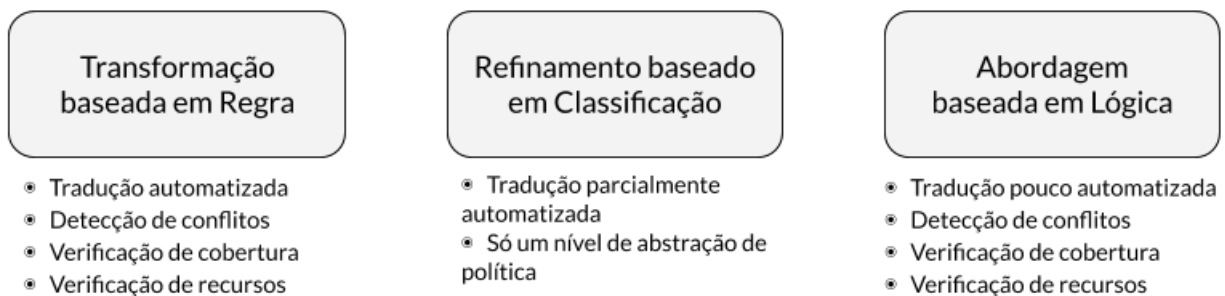


Figura 4 – Categorias de Refinamento de Política

Refinador de Políticas

Este capítulo tem como objetivo apresentar detalhadamente a solução proposta. Na primeira seção encontra-se a apresentação do ambiente hospitalar pensado para a implementação da solução, na segunda, a visão geral da proposta deste trabalho, mostrando a arquitetura do modelo e explicando cada componente que a compõe. Na Seção 3.3 será descrito o funcionamento da proposta, enquanto a quarta seção apresentará o modelo de dados utilizado. Por fim, a última seção discorrerá sobre a política de gerenciamento definida para este trabalho.

3.1 Ambiente Hospitalar

Os hospitais inteligentes utilizam de tecnologias, como sensores e atuadores, para otimizar os serviços de saúde oferecidos e melhorar a vida dos seus pacientes e funcionários. Ele fornece soluções automatizadas dos serviços hospitalares ao capturar as informações do ambiente, analisando-as e agindo de acordo com as políticas pré definidas pelo hospital. As áreas do hospital estão integradas por meio dos sistemas IoT e diversas soluções são ofertadas, como *check-in* automatizado, controle de acesso inteligente, farmácia inteligente, rastreamento de bens e pessoas, telemedicina, monitoramento de pacientes em tempo real, entre outros. Para a construção dos exemplos que serão descritos a seguir, foram realizadas pesquisas na literatura e entrevistas com profissionais que atuam em hospitais. A Figura 41 apresenta alguns ambientes de um hospital inteligente.

1. **Equipamentos Médicos:** O hospital inteligente tem diversos dispositivos que são rastreados em tempo real. A localização e circunstância em que se encontra cada dispositivo é monitorada automaticamente. Por exemplo, sabe-se onde é possível encontrar ferramentas cirúrgicas aptas para uso ou em que etapa se encontra ferramentas que precisam de higienização. Além disso, alguns equipamentos com mais poder tecnológicos gerenciam e permitem o seu uso a partir de políticas. Como monitores localizados dentro do quarto do paciente cujas informações exibidas só

podem ser visualizadas pelo médico responsável ou desfibriladores externos automáticos que só podem ser manuseados por enfermeiros e médicos. As políticas podem definir a ação que deverá ser tomada de acordo com os eventos que ocorram com os equipamentos. Por exemplo: caso haja uma cirurgia marcada e for constatado que não há mais ferramentas cirúrgicas adequadas para uso, uma mensagem de aviso é enviada para o funcionário responsável; quando o monitor identificar uma nova tentativa de acesso, só ocorrerá a liberação caso a identidade do usuário for equivalente à do médico responsável pelo paciente; entre outros.

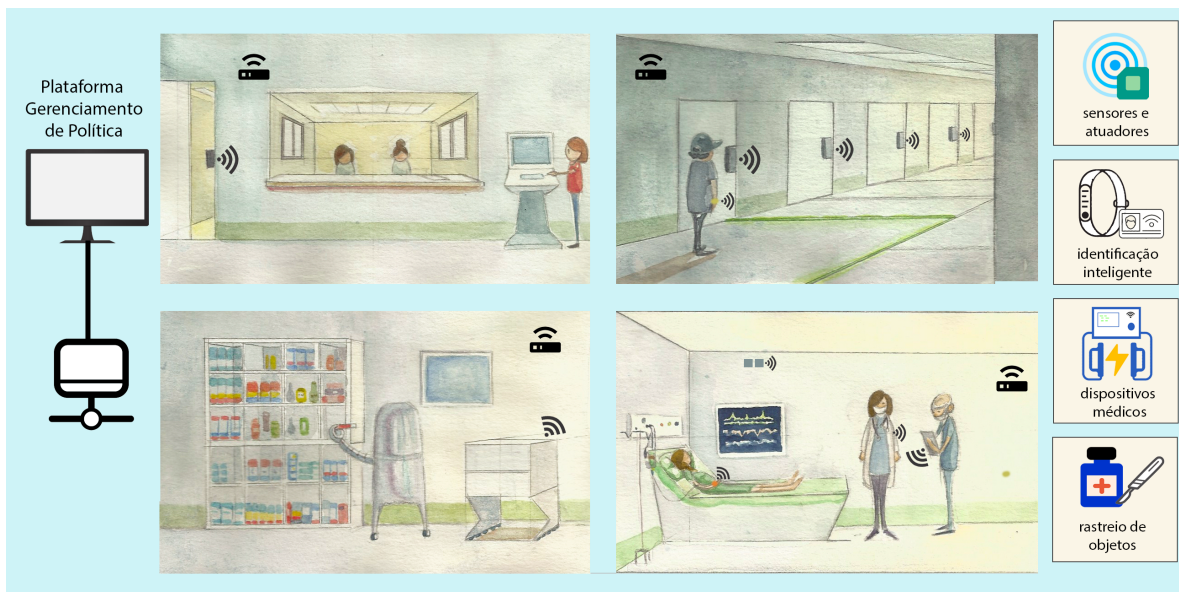


Figura 5 – Hospital Inteligente (Autoria: Stephanie Soares)

2. **Farmácia Inteligente:** A farmácia utiliza um sistema online integrado aos outros setores do hospital. Ela é composta por diferentes robôs, geladeiras, armários e remédios. Os remédios são catalogados e identificados com tecnologias de identificação (como etiqueta RFID e código de barra). Desta forma, os estoques são monitorados para que não ocorra falta de medicamento ou acúmulo de produtos fora da validade. O robô, a partir de uma solicitação, retira o medicamento do estoque e adiciona informações sobre o paciente que será medicado. Um outro robô, com capacidade de transitar pelo hospital, leva os medicamentos para os respectivos pacientes. Desta forma, quando um médico prescrever o medicamento de um paciente, a farmácia será atualizada em tempo real e poderá retirar e encaminhar o remédio sem necessidade de intervenção humana e com eficiência. É possível rastrear os medicamentos que necessitam de refrigeração adequada e verificar se as condições de armazenamento estão sendo obedecidas, utilizando-se de políticas para definir as ações que deverão ser tomadas. Ou, quando for agendado um procedimento médico, como endoscopia, uma política define quais os medicamentos serão utilizados e o robô precisará pegar e encaminhar.

3. **Navegação pelo hospital:** Os pacientes e visitantes têm mais facilidade de locomoção na área hospitalar. A realização do *check-in* é mais eficiente, podendo ser realizada, por meio de aplicações móveis, até mesmo antes da chegada ao hospital. As portas são automatizadas e o acesso é feito a partir de verificação em tempo real. Cada porta contém um sensor com a capacidade de se comunicar com tecnologias de identificação (RFID, NFC, entre outros) e verificar se o sujeito pode ter acesso ao ambiente. O paciente pode utilizar de navegação disponível por dispositivo móvel, como também utilizar de indicações visuais instaladas nos corredores, como LEDs que acendem quando necessário.

Um exemplo de uso de política na navegação do usuário é: ao chegar em um hospital e realizar o check-in, o paciente recebe uma pulseira com um chip, no qual estarão contidas as informações pessoais do usuário e a sua categoria de gravidade. A gravidade é definida de acordo com os sintomas do paciente e é dividida em cinco tipos, são elas: Emergencial e Imediato (vermelha), Urgente e prioritário (laranja), Pouco Urgente (amarela), Não Urgente prioritários (Azul) e Não Urgente (verde). A pulseira é utilizada para permitir a entrada do paciente nos ambientes do hospital, impossibilitando-o de entrar em salas de outras categorias. Entretanto, podem ocorrer situações em que deverá permitir pacientes de categoria diferente acessar uma sala de outra categoria, como a seguinte: está acontecendo uma nova epidemia viral, os pacientes são da categoria vermelha. No entanto, não há mais leitos disponíveis para estes pacientes, mas tem na ala amarela. Como o hospital não pode negligenciar o atendimento, é determinado que agora os pacientes da categoria vermelha podem acessar o ambiente que antes era apenas disponível para a categoria amarela. Então, é preciso alterar urgentemente a política dos acessos às áreas do hospital para que a partir de agora os pacientes possam acessar as novas áreas definidas e não fiquem sem atendimento médico.

3.2 Visão Geral da Arquitetura

Este trabalho apresenta uma solução que tem como objetivo auxiliar o gerenciamento de dispositivos IoT em hospitais inteligentes utilizando refinamento de política. A Figura 6 apresenta uma visão geral da proposta. A arquitetura é dividida nos módulos *Front-end*, Refinador de Políticas, Gerenciador de Dados, *Middleware* (MW) e Dispositivos IoT. Este trabalho focou-se nos módulos Refinador de Política, Gerenciador de Dados e *Middleware*. O módulo *Front-end* é responsável por apresentar a interface da plataforma para o usuário. Será onde ele poderá gerenciar (criar, editar e visualizar) as políticas. Nesta etapa, o usuário utilizará de uma linguagem visual para criar a política. Após criada ou editada, esta política será traduzida para um arquivo XML.

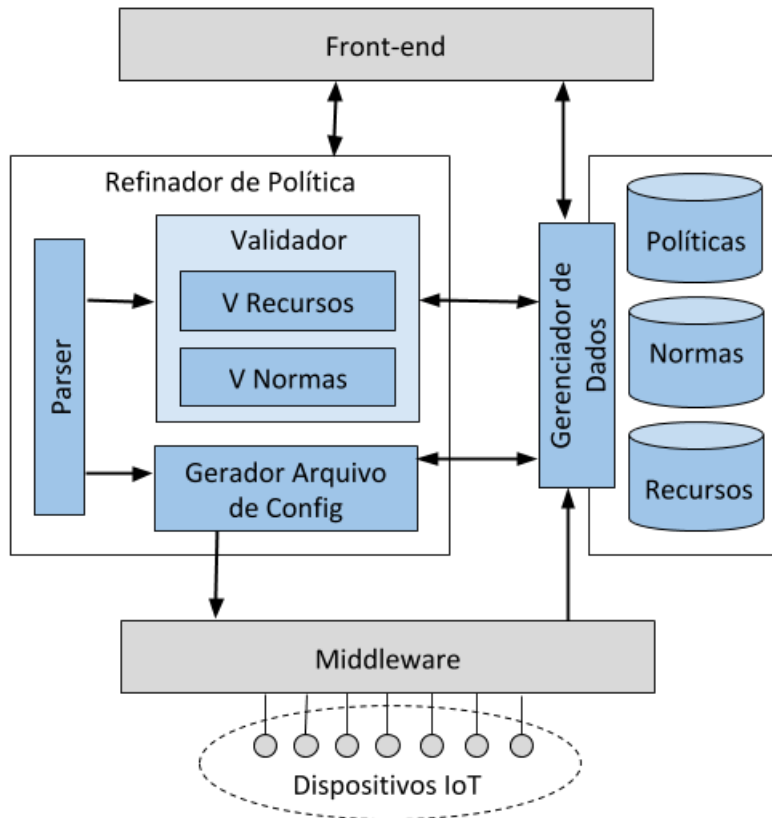


Figura 6 – Visão Geral da Arquitetura

O módulo de Refinamento de Política (RF) é responsável por realizar o processo de refinamento da política descrita em XML para um arquivo de configuração. Este, por sua vez, será utilizado pelo *Middleware* para customizar o comportamento dos dispositivos. Este módulo é crucial para o funcionamento da proposta, visto que contém as etapas críticas de tradução e validação da política submetida. Ele é composto pelos seguintes submódulos: *Parser*, Validador e Gerador de Arquivo de Configuração (CFILE). O *Parser* é responsável por receber o arquivo xml da política e realizar o *parse*, bem como criar a *parse tree* com todas as informações contidas no XML utilizando uma linguagem de programação orientada a objeto. Os elementos mapeados e identificados nesta etapa serão utilizados posteriormente pelo Validador e Gerador de Arquivo. O Validado é composto pelo Validador de Recursos (VR) e o Validador de Normas (VN). O VR é responsável por garantir que os recursos IoT, que estão sendo utilizados para realizar a captura do evento ou efetuar a ação resultante, existem e estão disponíveis para uso. Enquanto que o papel do VN é verificar se a política cumpre com todas as normas do hospital. Por fim, o Gerador de CFILE utiliza todas as informações fornecidas pelo *Parser* para criar um arquivo de configuração em uma linguagem que possa ser compreendida e executada pelo MW.

O Gerenciador de Dados desempenha o papel de gerenciar os dados que são inse-

ridos ou requisitados dos Bancos de Dados (BDs). Existem três BDs: Políticas, Normas e Recursos. O primeiro armazena as políticas que foram submetidas pelos usuários, validadas pelo VR e VN, aceitas e estão em vigência no sistema do hospital. O segundo mantém as normas do hospital. Por exemplo: qual a categoria de paciente pode entrar em qual ambiente do hospital e qual o intervalo de temperatura permitido nas salas do hospital. E, por último, o terceiro é utilizado para armazenar todas as informações sobre os recursos IoT. Como quais dispositivos estão disponíveis no hospital, a localização, quais serviços eles oferecem, entre outros. Além disso, o Gerenciado de Dados também é responsável por organizar e inserir dentro Banco de Dado (BD) os novos recursos descobertos pelo MW.

Por fim, tem-se o *Middleware* cuja função é coordenar os dispositivos IoT existentes no hospital. Sendo assim, ele acessa as políticas de configuração e quando um evento acontece, o MW interpreta a política correspondente e define o comportamento dos dispositivos definidos. Pode-se ter como base o MW apresentado no artigo (SINGH; BACON, 2014). E os dispositivos IoT são os atuadores e sensores que agem no hospital, como lâmpadas, sensores de sinais vitais, sensor de luminosidade, temperatura, entre outros.

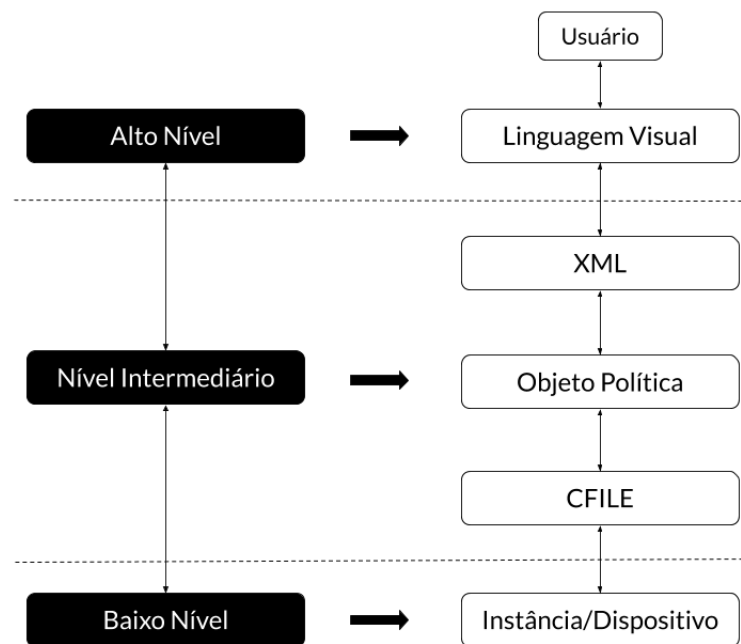


Figura 7 – Camadas de Abstração da Política

Assim, o refinamento de políticas consiste em todo o processo de transformação de política apresentado nesta seção. A Figura 7 apresenta as camadas de abstração da política. Começando com a política definida no *front-end* em uma linguagem visual, para a tradução no formato XML até o arquivo de configuração que será utilizado pelo MW para configurar os dispositivos. Desta forma, todo o processo de refinamento acontece nas

etapas que antecedem o *middleware* que tem o papel de utilizar das políticas já refinadas para determinar o comportamento dos dispositivos IoT.

3.3 Funcionamento da Arquitetura

Este trabalho utiliza refinamento de política para a partir de uma política de alto nível, gerar uma de baixo nível. Para isso, é necessário garantir que a política possa ser traduzida corretamente, assegurando todas as suas características iniciais. Além disso, durante o processo de refinamento é importante considerar as normas do hospital e os recursos existentes. Nesta seção será descrito o funcionamento do sistema que pode ser visualizado na Figura 8 e demonstra o processo, desde a submissão de uma política pelo usuário até a configuração do sistema IoT e seus dispositivos.

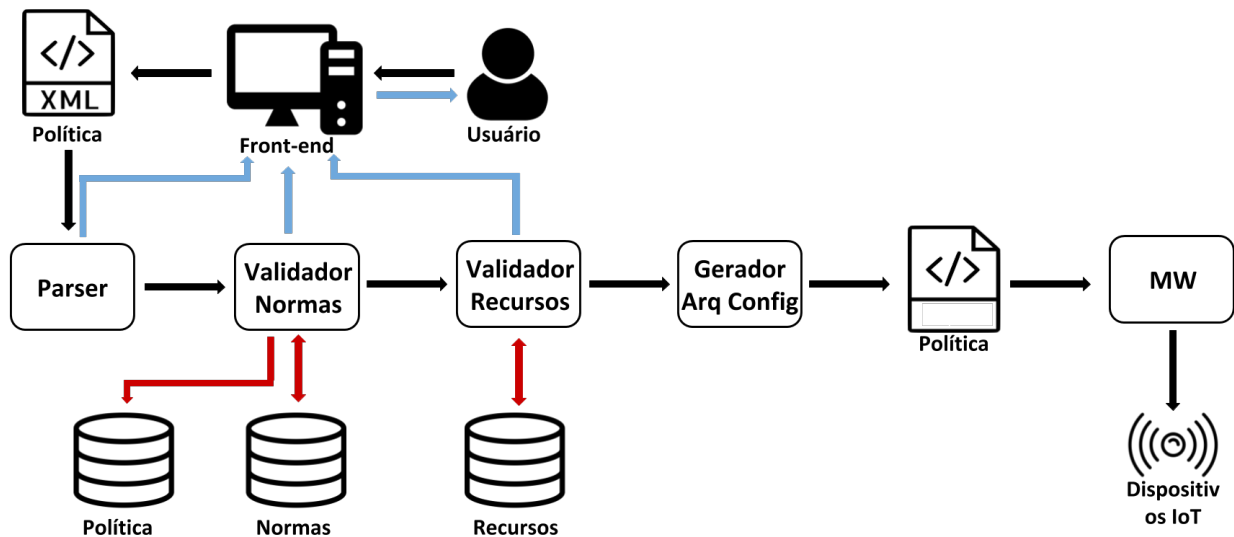


Figura 8 – Funcionamento da Solução

Quando um usuário cria uma nova política ou edita uma já existente, a política que está descrita em uma linguagem de programação visual é submetida. Após ela ser submetida na plataforma no *Front-end*, é traduzida para um arquivo XML. Esse arquivo contém todas as informações da política e também do usuário que a submeteu. Este arquivo é o artefato de entrada para o começo do processo de refinamento proposto neste trabalho. Então, o *Parser* realiza o *parse* da política em formato XML para um objeto de linguagem de programação e cria uma *parse tree*. Ela será utilizada no processo de validação e criação do arquivo de configuração. Se qualquer erro for encontrado durante o processo de *parse*, uma mensagem será retornada para o usuário alertando-o sobre o erro para que ele possa realizar as adequações necessárias e submeter novamente.

Quando o *parse* é concluído com sucesso, inicia-se o processo de validação. Nesta etapa, o Validador tem como entrada a política em um nível de abstração intermediária-

rio utilizando uma linguagem de programação e também a *parse tree*. Primeiramente, realiza-se a validação de acordo com as normas hospitalares. Para fazer isso, o Validador de Normas requisita ao Gerenciador de Dados as normas atuais do hospital. Então, é conferido se a política submetida está de acordo com as normas. Caso esteja, o processo continua. Caso não, a validação é interrompida e uma mensagem de erro é retornada ao usuário apontando a norma hospitalar que não está sendo obedecida. Após isso, é necessário verificar tanto os serviços e dispositivos utilizados para capturar os eventos, quanto se a condição e ação descritas na política podem ser realizadas pelos dispositivos existentes da forma que foi requerida. Sendo assim, o Validador de Recursos requisita ao Gerenciador de Dados as informações sobre os serviços e dispositivos disponíveis no BD de Recursos para que possam realizar o evento e a ação no local definido na política. Se não existir, uma mensagem de erro é retornada ao usuário. Uma vez que a política é validada, ela é salva no BD de Políticas. A política é salva no formato XML para que tanto a camada acima do usuário, quanto a abaixo do MW sejam independentes e possam mudar de linguagens caso necessário.

Após a validação, é necessário concluir o processo de tradução para uma linguagem que será mais fácil de ser interpretada pelo MW. Sendo assim, o Gerador de CFILE utiliza o objeto política e também a *parse tree* para construir o arquivo na linguagem de programação que será interpretada pelo MW para determinar o comportamento dos dispositivos. O arquivo contém as informações dos sensores e atuadores utilizados para executar a política, como também o evento, a condição e a ação descritos na política.

3.4 Modelo de Dados

Esta seção apresenta como os recursos IoT, políticas e normas hospitalares são armazenados nas estruturas de armazenamento. O modelo dos recursos está representado na Figura 9. Nele temos o componente Dispositivo, que é o recurso em si, contendo número de identificação, localização atual, os serviços que fornece e os eventos e ações que podem ser realizados a partir desses serviços. A seguir temos a descrição dos elementos presentes no modelo de recursos.

Dispositivo

- ❑ **id_dispositivo:** número de identificação do dispositivo;
- ❑ **nome_dispositivo:** o nome que o dispositivo utiliza;

Serviço

- ❑ **id_serviço:** número de identificação do serviço

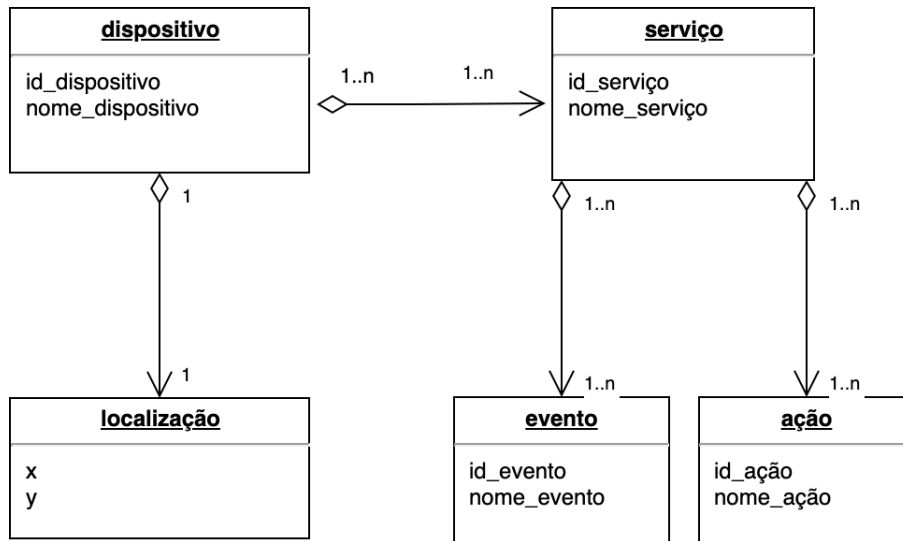


Figura 9 – Modelo de Dados dos Recursos

❑ **nome_serviço:** nome do serviço

Evento

❑ **id_evento:** número de identificação do evento

❑ **nome_evento:** nome do evento

Ação

❑ **id_ação:** número de identificação da ação

❑ **nome_ação:** nome da ação

Localização

❑ **x:** coordenada x do dispositivo

❑ **y:** coordenada y do dispositivo

A política é a orientadora do sistema IoT. É através dela que o sistema saberá qual ação tomar e realizar. Na Figura 10 é possível visualizar a estrutura de dados que representa uma política. Nela é apresentada as informações sobre a política e do funcionário que a criou e a regra que é formada pelo evento, condição e ação.

Política

❑ **id_pol:** número de identificação da política

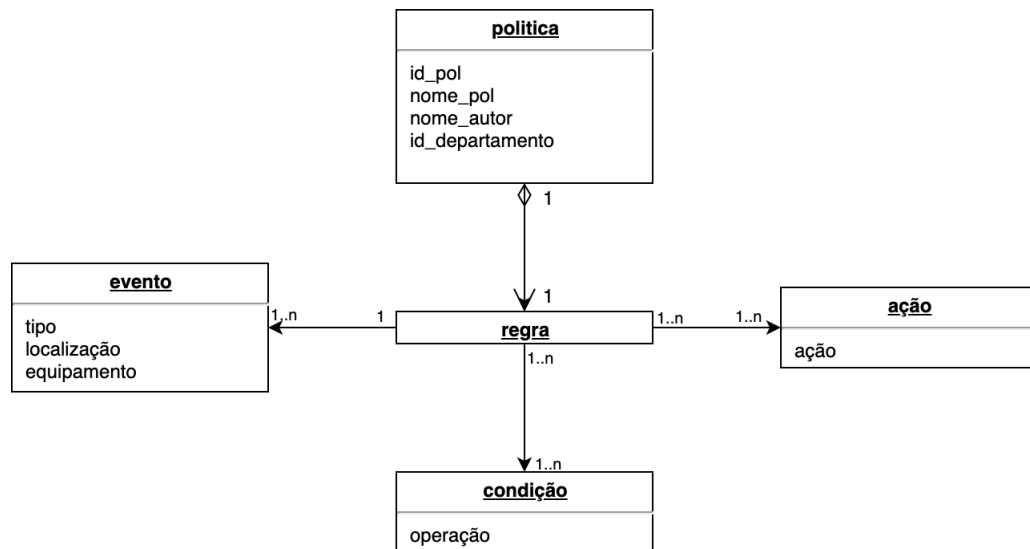


Figura 10 – Modelo de Dados das Políticas

- ❑ **nome_pol:** nome dado pelo autor para a política
- ❑ **id_autor:** número de identificação do autor da política
- ❑ **id_departamento:** número de identificação do departamento que a política foi criada

Regra

- ❑ **evento:** evento que desencadeará a utilização da regra
 - **tipo:** o nome do evento
 - **localização:** localização de onde o evento tem que ocorrer para valer a política
- ❑ **condição:** condição que desencadeará a ação
 - **operação:** a operação que é formada pelas condições que determina a realização da política
- ❑ **ação:** ação que o sistema deve realizar
 - **ação:** o nome da ação que deverá ser realizada

Por fim, há o modelo dos protocolos hospitalares, ilustrado na Figura 11. Eles são as regras que regem o hospital e consequentemente permitem ou não a implementação de uma política. O modelo contém o protocolo e sua identificação, podendo contar uma ou mais normas. Nas norma é apresentada a sua identificação e a regra que dita o que é permitido ou não no hospital. Ela funciona como uma política de autorização.

Protocolo

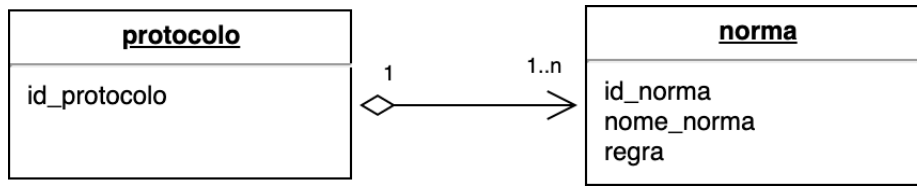


Figura 11 – Modelo de Dados das Normas

- ❑ **id_protocolo:** número de identificação do protocolo

Norma

- ❑ **id_norma:** número de identificação da norma
- ❑ **nome_norma:** nome dado para a norma
- ❑ **regra:** regra que determina o que pode ou não ser realizado no hospital

3.5 Política de Gerenciamento

Para realização da solução, foi necessário estabelecer um esqueleto para a política no formato do arquivo XML. O formato de política utilizado foi o ECA e o XML construído como apresentado na Figura 12. Nesta seção será explanado o que foi definido e utilizado neste trabalho.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <policy>
4      <info id=" " name=" ">
5          <id_author> </id_author>
6          <id_dep> </id_dep>
7      </info>
8
9      <event type=" ">
10         <at_dep> </at_dep>
11         <at> </at>
12         <equip> </equip>
13     </event>
14
15     <condition>
16         <operation op=" " </operation>
17     </condition>
18
19     <action>
20         <do> </do>
21     </action>
22 </policy>

```

Figura 12 – Modelo do XML

Evento

O elemento Evento, apresentado como <event> na Figura 12, apresenta qual o evento que deverá ser observado para a atribuição daquela política. Dentro do corpo deste elemento é contido o tipo <type> de evento para que a política deverá ser atribuída, como também para qual localização <at> no hospital ela é válida. O lugar pode ser uma sala, departamento, todo o hospital. Há também a possibilidade de aplicar a política a um equipamento <equip> específico.

Foram utilizados os seguintes eventos neste trabalho:

1. **Leitura de id (new_id)**: Evento de quando um sensor responsável por ler cartão ID identifica que tem um cartão encostado.
2. **Leitura de pulseira (new_band)**: Evento de quando um sensor identifica uma nova pulseira de identificação de paciente próxima.
3. **Monitoramento paciente (p_monitoring)**: Evento de monitoramento dos dados vitais do paciente.

Condição

O elemento de Condição, apresentado como <condition> na Figura 12, contém a operação que será utilizada para determinar se uma ação será ou não realizada. Os operadores <op> são utilizados para informar qual a operação que será realizada entre os dois valores contidos no elemento <op>. É possível visualizar todos os operadores e suas respectivas descrições na Tabela 2.

Tabela 2 – Operadores

Operador	Nome	Descrição
gt	maior que (greater than)	Compara dois valores e será verdade se o valor desejado for maior que o comparado
lt	menor que (less than)	Compara dois valores e será verdade se o valor desejado for menor que o comparado
equal	igual	Compara dois valores e será verdade se ambos tiverem o mesmo valor
and	e	É verdade se ambos os valores são verdadeiros
btw	entre (between)	É verdade se o valor estiver no intervalo fornecido
diff	diferente (different)	Compara dois valores e será verdade se ambos tiverem valores diferentes
or	ou	É verdade se um ou os dois valores forem verdadeiros

Ação

Quando uma nova política é submetida, torna-se necessário verificar quais são as ações que elas realizarão para garantir que não infrinjam as normas dos hospital. Dentro

do corpo do elemento Ação <action> é contida a chamada da execução da ação <do> que deverá ser realizada se a condição for satisfeita.

1. **Abrir porta (open_door):** Consiste em abrir a porta que identificou algum evento de tentativa de acesso. É preciso garantir que só entre na sala pessoas que tenham permissão para o mesmo. Tanto paciente quanto funcionários só podem estar em ambientes compatíveis com sua categoria.
2. **Mudar temperatura (change_temp):** Consiste em alterar a temperatura do objeto monitorado quando não está na temperatura desejada. É preciso garantir que a temperatura dos ambientes sempre estejam no intervalo estabelecido pelas normas.
3. **Ligar equipamento (equip_on):** Consiste em permitir a utilização de um equipamento para os funcionários que têm autorização. É preciso garantir que a categoria do funcionário que está tentando usar o equipamento do evento tem a permissão para utilizá-lo.
4. **Alarme para enfermeiro responsável (alarm_nurse):** Consiste em avisar, de acordo com as condições estabelecidas, o enfermeiro responsável pelo paciente monitorado.

Implementação e Avaliação

Como forma de verificar a proposta descrita no Capítulo 3, realizou-se testes por meio da implementação e execução da solução. Primeiramente, será descrito, em detalhes, como a proposta foi implementada, seguido pelo ambiente que foi utilizado para os testes. Por fim, será feita a apresentação e análise dos resultados encontrados.

4.1 Implementação

Com o intuito de provar que a solução proposta pode ser aplicada, foi implementado o processo de refinamento do arquivo XML até o arquivo de configuração e o seu envio para o MW, como ilustrado no modelo da Figura 6. Essa seção apresenta o processo de implementação, os testes e resultados obtidos a partir da execução e simulação da proposta. Para isso, utilizou-se a linguagem de programação Python, versão 3.x. A escolha dessa linguagem se deu devido a algumas vantagens que ela proporciona, como rápido desenvolvimento, fácil integração com novos módulos e, inclusive, outras linguagens. Por ser uma linguagem interpretada, possui portabilidade para diversos sistemas operacionais (Windows, Linux e MacOS). Além disso, o Python tem sido utilizado como linguagem de programação para projetos no contexto de IoT (PATCHAVA; KANDALA; BABU, 2015) (SHETE; AGRAWAL, 2016) (QUADRI; SATHISH, 2017).

A utilização do Python como linguagem de programação para o trabalho influenciou na forma de representar os dados na solução. Os BDs foram implementados usando arquivos CSV, pois utilizou-se da biblioteca Pandas. Ela foi criada para manipulação e tratamento de dados no Python e se apresenta como ferramenta bastante intuitiva para esse propósito, agilizando o processo de desenvolvimento da solução (MCKINNEY, 2012).

O primeiro passo foi a definição do arquivo de política em XML, considerando como e quais informações seriam dispostas. Anteriormente, no Capítulo 3 foi apresentado o esqueleto do arquivo XML que descreve a política. Na Figura 13 é possível visualizar uma política completa implementada e utilizada neste trabalho. Ela contém as informa-

ções sobre: o id da política (linha 3), sendo este valor dado pelo sistema na hora que se implementa uma nova política; o nome da política “Porta 100” (linha 3); o número de identificação do autor (linha 4); e o identificador do departamento do autor da política (linha 5). Esta é uma política definida para que quando o evento (linhas 7-11) de identificar uma pulseira de um paciente (linha 7) em uma porta da sala 100 (linha 9) ocorra, essa pulseira é lida e verifica-se a categoria daquele paciente. Dentro do elemento `<condition>` define-se a condição da política (linhas 12-17). Caso seja da categoria laranja ou vermelha (linhas 13-16), a porta abrirá pra ele. Esta ação de abertura da porta está definida entre as linhas 18 e 20.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <policy>
3      <info id="1" name="Porta 100">
4          <id_author>15</id_author>
5          <id_dep>1</id_dep>
6      </info>
7      <event type="new_band">
8          <at_dep>1</at_dep>
9          <at>100</at>
10         <equip>None</equip>
11     </event>
12     <condition>
13         <operation op="or">
14             <operation op="equal">band;orange</operation>
15             <operation op="equal">band;red</operation>
16         </operation>
17     </condition>
18     <action>
19         <do>open_door</do>
20     </action>
21 </policy>

```

Figura 13 – Política de gerenciamento de porta automática

Com o modelo xml das políticas pronto, foi implementado o *parser*. O *parser* é responsável por encontrar e mapear todos os dados contidos no XML para sua variável correspondente na classe política. Além disso, há também a criação da *parser tree*. Nessa etapa as operações contidas na condição (linhas 12-16) são identificadas e a *parser tree* é construída. Para isso, a biblioteca `cElementTree` foi utilizada para leitura do arquivo XML. Ela cria uma árvore com todos os elementos do documento XML. Após isso, foi necessário acessar os nós da árvore para ter acesso a todas as informações existentes. As informações foram inseridas em um objeto Política na linguagem python. Os dados contidos nos elementos informação `<info>`, evento `<event>` e ação `<action>` são inseridos da mesma forma que está descrito no XML, enquanto que condição `<condition>` é analisada e trabalhada para a criação da *parse tree*. Optou-se pela criação de uma árvore binária,

como pode ser visto na Figura 14 em que temos os operadores centralizados nos nós e os valores que serão utilizados para inferir se a condição é verdadeira ou falsa, como os filhos, identificador à esquerda e valor da variável à direita. Pode-se ter infinitas operações definidas numa política, aumentando o tamanho da árvore e duração do processamento.

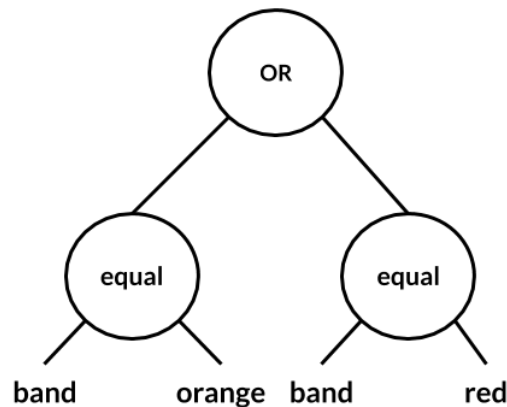


Figura 14 – Exemplo de *Parse Tree*

Uma vez realizado o *parse* do arquivo, faz-se necessário validar a política. Antes de conferir as normas e recursos, é importante checar as informações contidas no elemento <info> da política. Sendo assim, é conferida a existência do local em que o usuário deseja implementá-la. Em seguida é verificado se o funcionário pode criar uma política para aquele local e, ainda, se a ação desejada na política é válida para os serviços disponíveis no hospital. Em caso afirmativo, é feita a validação das normas. Para realização da validação é verificado qual o tipo de ação que será realizada pela política caso a condição seja cumprida e confere se há alguma norma no hospital sobre aquela ação. Caso esteja, esta etapa é concluída com sucesso e avança para a validação de recurso. Caso desobedeça, será necessário interromper o processo de refinamento da política e retornar uma mensagem para o usuário informando o erro identificado.

Um exemplo de norma hospitalar, utilizada neste trabalho, estabelece que a temperatura da geladeira de medicamentos deve permanecer entre 2 °C e 8 °C. Sendo assim, caso uma política seja criada para esse equipamento, o validador verifica se a norma é respeitada. Ou seja, se o usuário tentar implementar uma temperatura fora desse intervalo para esse equipamento, a política não será validada e uma mensagem de erro será apresentada. As checagens de informação e de normas são sequenciais, então, caso a política não seja validada em qualquer uma delas, o processo é interrompido e uma mensagem de erro, descrevendo o motivo da falha de validação, é enviada ao usuário, facilitando no processo de entendimento da causa da falha.

O próximo passo é garantir que os recursos utilizados existem no local solicitado, o que é realizado em duas etapas. A primeira consiste em verificar se o evento definido na política pode ser realizado no local desejado, enquanto a segunda é se a ação pode

ser executada. Para tal, é preciso verificar quais serviços são necessários para realização do evento e da ação definida. Tendo a informação do serviço, é realizada uma busca nos recursos para saber se há algum dispositivo que realize tais eventos no local. Se for encontrado, então segue o processo para o gerenciador de dados, mas se não, é retornada uma mensagem para o usuário informando qual foi o erro. Por exemplo, a política da Figura 13 tem o evento que identifica a presença de uma nova pulseira, então, é preciso ter um sensor na sala 100 do departamento 1 que possa ler as pulseiras dos pacientes na localidade descrita no documento. E, para a realização da ação definida no exemplo supracitado, é preciso que exista uma porta automática que possa realizar a abertura da mesma caso a condição seja satisfeita.

Após conferir que a política não infringe nenhuma norma e que todos os recursos utilizados existem, é preciso gerar o arquivo de configuração, como o da Figura 15. Nela são apresentadas as informações a respeito do número identificador do sensor (linha 1) e atuador (linha 2) que são utilizados para realização da política. A linha 4 apresenta o evento que ativará a verificação desta política. A linha 5 e 6 apresentam a condição e ação que será executada caso a condição seja positiva. Essas duas últimas linhas estão escritas na linguagem Python e podem ser facilmente executadas. O objeto Política é utilizado para escrever as informações do sensor e do atuador.

```
1  SENSOR_ID: 3
2  ACTUATOR_ID: 2
3
4  when new_band
5  if (( band == 'orange' ) or ( band == 'red' )):
6  |   answer = 'open_door'
```

Figura 15 – Exemplo de Arquivo de Configuração de uma Política

Quando o arquivo de configuração é gerado, é concluído o processo de tradução e refinamento pelo módulo Refinador de Política (ver Figura 6). Entretanto, para uma melhor análise da proposta, foi implementado um *middleware* que recebe o arquivo de configuração e realiza uma organização das políticas existentes que ele deverá considerar para ditar o comportamento dos dispositivos do sistema IoT. Sendo assim, quando o *middleware* tem conhecimento de um novo arquivo de configuração, ele irá primeiro armazenar este arquivo em seu repositório de políticas já no formato CFILE. Essas políticas foram armazenadas pelo seu número de identificação (id), pois caso já haja uma política com aquele número, é sabido que foi uma edição de política e ela será substituída pela versão mais atual.

4.2 Ambiente de Teste

Esta seção apresentará o ambiente que foi construído e utilizado para a realização dos testes da proposta, como as políticas, normas e recursos que foram definidas e implementadas. Para a definição das políticas utilizadas, foram considerados os cenários descritos na Seção 3.1 e também foram definidas algumas características do ambiente hospitalar imaginado para os testes. Algumas das políticas implementadas serão descritas a seguir:

- ❑ **Política 1: QUANDO** identificado nova pessoa na porta 100. **SE** categoria for do tipo laranja, **ENTÃO** a porta abre.
- ❑ **Política 1 modificada: QUANDO** identificado nova pessoa na porta 100. **SE** categoria for do tipo laranja ou vermelha, **ENTÃO** a porta abre.
- ❑ **Política 2: QUANDO** requisitado iniciar desfibrilador. **SE** enfermeiro ou médico, **ENTÃO** ligar desfibrilador.
- ❑ **Política 2 modificada: QUANDO** requisitado iniciar desfibrilador. **SE** enfermeiro ou médico ou técnico de enfermagem ou auxiliar de enfermagem, **ENTÃO** ligar desfibrilador.
- ❑ **Política 3: QUANDO** verificação de sinais vitais. **SE** se ritmo respiratório maior que 20/minuto e frequência cardíaca maior que 100BPM/minuto, **ENTÃO** notificar enfermeiro responsável.
- ❑ **Política 4: QUANDO** verificado temperatura de todas as salas. **SE** se temperatura menor que 21 ou maior que 25, **ENTÃO** mudar temperatura para 22.

Tomando por base as políticas escolhidas e também os documentos de protocolos hospitalares e suas normas, algumas normas foram utilizadas para realização dos testes, como: a autorização do acesso de pacientes à áreas de acordo com sua classificação de gravidade, as normas que ditam o intervalo de temperatura permitido em uma sala hospitalar, o intervalo de temperatura permitido em uma geladeira de medicamentos, intervalo de iluminação de um ambiente hospitalar e profissionais do hospital que podem utilizar os equipamentos hospitalares.

Então, foi necessário definir os recursos que seriam utilizados para os testes e inseri-los na base de dados. Para isso, foi realizado um levantamento dos recursos que poderiam ser utilizados pelas políticas que seriam implementadas para os testes. Então, foram definidos os serviços ofertados por dispositivos implantados no hospital, como: identificação de presença, leitor de pulseira, temperatura, movimento, luminosidade, umidade, leitor de cartão, frequência cardíaca, frequência respiratória, entre outros. E os tipos

dispositivos que podem oferecer um ou mais desses serviços, como: fechadura inteligente (leitor de pulseira e destrava automaticamente), monitor de sinais vitais (frequência cardíaca, frequência respiratória, etc), sensor de temperatura (temperatura, umidade), entre outros. Assim como os dispositivos de cada tipo, junto com a sua localização no hospital.

Além disso, foi necessário definir algumas características gerais do hospital. O hospital foi dividido em departamentos e salas. Cada sala pertence a um departamento específico. Por exemplo: as salas de número 100, 101, 102 pertencem ao departamento de número 1, enquanto as salas 205, 208, 209 ao departamento de número 2 e assim por diante. Definiu-se que os funcionários do hospital seriam médicos, enfermeiros, técnicos de enfermagem e auxiliar de enfermagem. E como exemplos de equipamentos médicos foram utilizados os Eletrocardiógrafo ECG, Monitor de PA/FC/ECG/Saturação, Desfibrilador, Ventilador Mecânico, Bisturi Elétrico, Máquinas de Radiografia, Tomografia e Ressonância.

4.3 Experimentos e Resultados

Este trabalho tem como objetivo principal fornecer uma solução para que os funcionários de hospitais inteligentes sejam capazes de gerenciar o comportamento dos dispositivos existentes no sistema IoT, possibilitando o funcionamento correto e eficiente dos serviços ofertados pelo hospital para seus pacientes e funcionários. Esta seção apresenta os testes realizados e resultados obtidos a partir da implementação da proposta. Ela é composta por três partes: teste de execução do processo de refinamento, simulação do ambiente e testes de desempenho em relação ao tempo de resposta, baseado nos trabalhos de (PONTE; GOMES, 2018) e (CHEMINOD et al., 2019).

Testes de Execução

Para verificar que a solução implementada cumpre os três requisitos propostos para o refinamento de políticas em hospitais inteligentes deste trabalho, foram realizados testes de execução e, em seguida, a análise dos resultados obtidos. Os requisitos são Tradução, Validação dos Recursos e Validação das Normas (definidos na Seção 2.3.1). Primeiramente, selecionou-se nove políticas. Elas foram descritas no modelo de arquivo XML, como no exemplo da Figura 13 e utilizadas como artefato de entrada na execução da proposta.

O primeiro requisito conferido foi o de Tradução, ou seja, se a política descrita em XML foi corretamente traduzida para o arquivo de configuração. Para isso, as políticas em arquivo XML foram submetidas ao refinador. Este retorna como saída um arquivo CFILE que permite sua análise e verificação para garantir a corretude do processo. A tradução é realizada de forma automática, sem a necessidade de intervenção humana em

nenhuma etapa de execução. Verificou-se que ela é capaz de garantir a consistência e característica da política durante todo o processo, traduzindo-a corretamente. Além das entradas e saídas do refinador, também foram analisadas as versões intermediárias da *parse tree* e do objeto Python. Esta verificação foi necessária para garantir que todas as etapas do processo foram executadas corretamente. As Figuras 13 e 15, mostradas anteriormente, apresentam respectivamente a entrada e saída de uma política.

Em seguida, os requisitos de validação de normas foram testados e analisados. Essa etapa foi necessária para certificar que os recursos e normas desempenham corretamente seus respectivos papéis e que a validação da política em relação às normas do hospital está correta. De forma a testar ambas possibilidades, foram submetidas aos testes tanto políticas que estavam em consonância com as normas hospitalares como também as que desobedeciam às mesmas. Como resultado dos testes a saída esperada deve ser a mensagem de “política correta”, indicando que a política está em consonância com as normas e permitindo a continuação do refinamento para a próxima etapa: validação dos recursos. No caso da detecção de políticas incoerentes com as normas o processo de validação é interrompido e uma mensagem de erro é retornada para o usuário. Por exemplo: uma norma do hospital é que apenas enfermeiros e médicos podem ligar e utilizar desfibriladores, então uma política que permite que assistentes de enfermagem possam ligar e utilizar um desfibrilador é submetida. Esta política deve ser invalidada, pois não está respeitando a norma supracitada. Desta forma, durante a realização dos testes, esta política foi submetida e conferiu-se que a solução pode corretamente invalidar a política e retornou a mensagem exibida na Figura 16, junto com a informação das categorias que podem manusear o equipamento descrito na política.

```

erro de autenticação: categoria não permitida

      EQUIPAMENTO  DEPARTAMENTO  CATEGORIAS PERMITIDAS
1      DISFIBRILADOR           2      MÉDICO, ENFERMEIRO

```

Figura 16 – Saída do Validador de Normas

O próximo passo foi a verificação do validador de recursos que foi realizada de forma semelhante ao validador de normas. Para isso, os testes utilizaram tanto políticas que continham recursos existentes nos dispositivos definidos para o ambiente de testes, quanto políticas que fazem uso de recursos inexistentes. A análise dos resultados demonstrou coerência com o que foi projetado e as saídas obtidas nos testes foi correta para todas as situações simuladas. A Figura 17 apresenta um exemplo de saída quando uma política é invalidada. Neste caso, o evento de captar a temperatura (`read_temperature`) não pode ser realizado no local definido na política, pois não há nenhum recurso que o ofereça.

erro de validação: Evento read_temperature não disponível no local.

Figura 17 – Saída do Validador de Recursos

O resultado obtido de cada requisito pode ser visualizado na Tabela 3. O trabalho cumpriu com os três requisitos propostos: tradução, validação de requisitos e validação de normas. Sendo estes realizados sem necessitar de intervenção humana. Além destes, verificou-se que o trabalho está obtendo sucesso na cobertura escrita. Isto é, a forma como a política foi definida inicialmente em XML foi fielmente traduzida para o CFILE. A verificação de conflitos não faz parte da proposta e também não foi implementada.

Tabela 3 – Resultado dos Requisitos

Requisito	Solução
(1) Tradução	Sim, automática
(2) Recursos	Sim, automática
(3) Cobertura	Sim, manual
(4) Conflitos	Não
(5) Normas	Sim, automática

Simulação

Uma simulação foi realizada para verificar se as políticas estavam sendo atualizadas e enviadas de forma correta para o *middleware*, após serem refinadas e o arquivo de configuração ser criado. Para isso, criou-se uma lista de sinais composta pelo ID do sensor e o valor captado e implementou-se o *middleware* que, ao receber um sinal de um dispositivo da sua rede, irá verificar se existe alguma política implementada. Caso exista, a política é interpretada e uma resposta é definida. A Figura 18 apresenta um trecho da saída desse teste. Ela tem cinco verificações de sinais. Cada verificação é composta por três elementos principais: Requisição, *Middleware* e Resposta. O elemento requisição diz respeito às informações do sensor (sensor id) e o que ele detectou (sinal sensor). O *Middleware* repete o id do sensor e separa o sinal recebido no serviço do recurso que o detectou (sinal serviço) e o valor coletado (sinal valor). Por fim, o elemento Resposta apresenta qual é a ação que deverá ser tomada (ação) e por qual recurso (atuador id).

Na Figura 18 temos a primeira requisição, linhas 1 a 3. Esta é a requisição de um paciente da categoria laranja tentando acessar uma sala (linha 1). O *middleware* verifica se há alguma política para esta situação (linha 2). Como existe, a política é executada e é retornado que a porta poderá ser aberta. Então, o *middleware* ordena que o dispositivo de id 2 realize a ação de abrir a porta (linha 3). Em seguida, um paciente da categoria vermelha tentou acessar a mesma porta (linha 4), porém a política implementada até então não permitia o seu acesso e a resposta foi "sinal inválido para o serviço"(linha 6),

```

1 REQUISICÃO - sensor id: 3 | sinal sensor: BAND READER;LARANJA
2 MIDDLEWARE - sensor id: 3 | sinal serviço: BAND READER | sinal valor: LARANJA
3 RESPOSTA - atuador id: 2 | ação: open_door

4 REQUISICÃO - sensor id: 3 | sinal sensor: BAND READER;VERMELHO
5 MIDDLEWARE - sensor id: 3 | sinal serviço: BAND READER | sinal valor: VERMELHO
6 RESPOSTA - sinal inválido para o serviço

7 REQUISICÃO - sensor id: 6 | sinal sensor: CARD READER;ENFERMEIRO
8 MIDDLEWARE - sensor id: 6 | sinal serviço: CARD READER | sinal valor: ENFERMEIRO
9 RESPOSTA - atuador id: 6 | ação: turnon_equipment

10 REQUEST - sensor id: 13 | sinal sensor: CARD READER;ASSISTENTE
11 MIDDLEWARE - sensor id: 13 | sinal serviço: CARD READER | sinal valor: ASSISTENTE
12 RESPOSTA - sem política implementada para o sensor

13 REQUISICÃO - sensor id: 3 | sinal sensor: BAND READER;VERMELHO
14 MIDDLEWARE - sensor id: 3 | sinal serviço: BAND READER | sinal valor: VERMELHO
15 RESPOSTA - atuador id: 2 | ação: open_door

```

Figura 18 – Simulação

significando que o *middleware* não ordena nenhuma ação. As linhas 7, 8 e 9 representam uma requisição de um enfermeiro para ligar um dispositivo médico. Como a política permite, é retornada a ação de ligar o equipamento. Entre as linhas 10 e 12 é apresentado um exemplo de quando não existe uma política implementada para o sensor. Nestas situações, pode existir uma ação *default* que seja realizada. E a linha 13 contém a mesma requisição da linha 4, só que desta vez a política havia sido alterada para permitir que pacientes da categoria vermelha possam entrar na sala. Por isso, na linha 15, o *middleware* ordena que o atuador de id 2 realize a ação de abrir a porta.

Testes de Desempenho

Além disso, também foi verificado o tempo necessário para a realização do processo completo de refinamento de política, assim como de cada etapa separada. O teste refere-se ao refinamento de 1000, 2000, 3000, 4000 e 5000 políticas para aferir a escalabilidade da solução. Além disso, o teste foi executado 30 vezes e foi obtida a média destas 30 execuções. O intervalo de confiança utilizado foi de 90%.

A Figura 19 apresenta o tempo de execução do parser, que compreende o tempo para abertura e parse do arquivo XML, a criação do objeto Python e a criação da *Parse Tree*. O tempo médio de execução por política fica aproximadamente 0,00007s, que é um tempo extremamente curto, comprovando a eficiência do parser. Além disso, é possível observar que o tempo de execução do parser se mantém linear à medida que o número de políticas aumenta, o que assegura a sua escalabilidade.

A Figura 20 apresenta o gráfico do tempo de execução do Validador de Normas. Neste procedimento inclui-se a verificação das informações do usuário e da política (identificador, nome, local, autor) e também se a política proposta (evento, ação e condição)

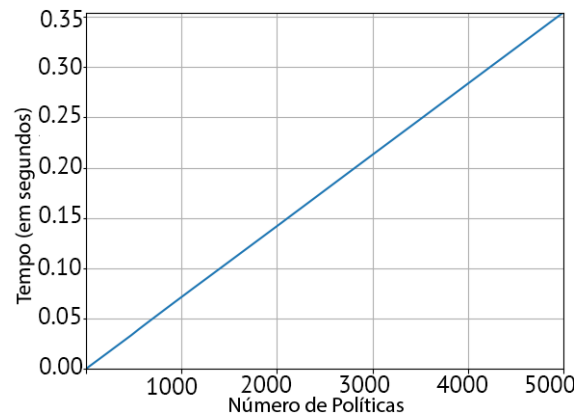


Figura 19 – Gráfico de Parser

estão condizentes com as normas do hospital. O tempo total para execução das 5000 políticas foi de 29,322s, ou seja, 0,0059s por política.

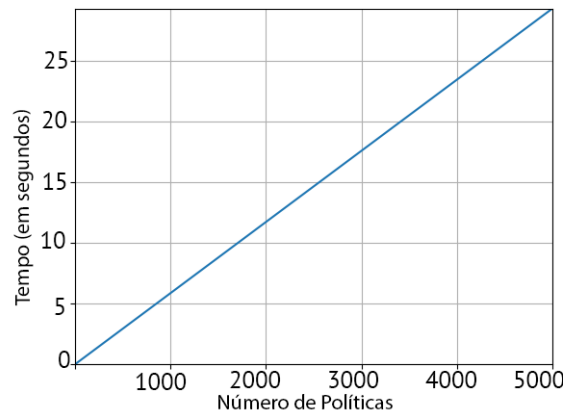


Figura 20 – Gráfico de Validação de Normas

Em seguida, o tempo de execução do Validador de Recursos. O gráfico do resultado pode ser visto na Figura 21. Ele apresentou um tempo total de 30,659s, o que resulta em 0,00613 por política. Este foi o tempo necessário para verificar os recursos que são utilizados na política, conferir o BD dos recursos e verificar se eles existem no hospital e também no local indicado. Desta forma, o processo completo de validação se mostrou eficiente com o tempo 0,012 por política. A linearidade dos gráficos exibidos nas Figuras 20 e 21 mostra a capacidade de escalabilidade da proposta.

Por fim, o gráfico do tempo de execução do arquivo de configuração pode ser visualizado na Figura 22. O tempo total para as 5000 políticas foi de 15,536s, ou seja, 0,003s por política, o que é um tempo ínfimo. Este é o tempo necessário para escrever o arquivo com as informações do Objeto política e da *Parse Tree*, como também de salvá-lo na pasta dos arquivos de configuração e enviar a informação para o *middleware* sobre a nova política ou atualização de uma política existente.

A Figura 23 apresenta o tempo utilizado para executar o refinamento das políticas da entrada do XML até a saída do arquivo de configuração. O tempo total para 5000

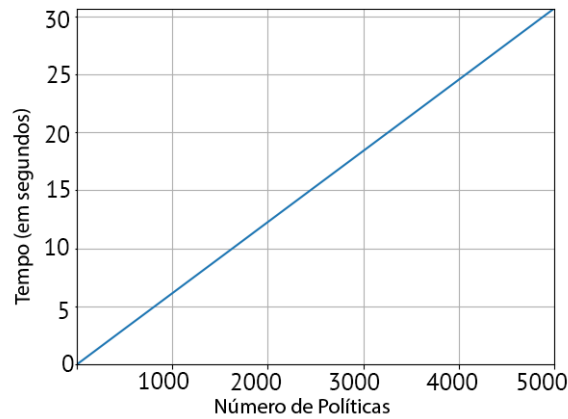


Figura 21 – Gráfico de Validação de Recursos

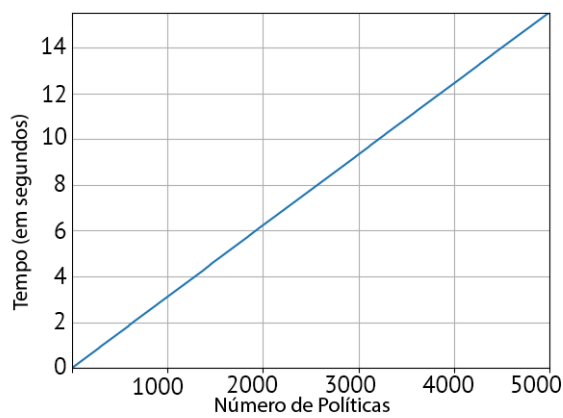


Figura 22 – Gráfico de Gerador de Arquivo de Configuração

políticas foi de 79,15 segundos, ou seja, o processo completo de refinamento de uma política dura em média 0,0158s por política. É possível analisar que o tempo utilizado pelo parser é só de 0,44% do processo todo, enquanto a validação das normas e de recursos gasta 37,04% e 38,77%, respectivamente. Isso acontece devido ao acesso aos dados das normas e dos recursos. O de geração de arquivo consome 19,63% do processo, pois também é preciso realizar acesso aos dados, escrevendo e lendo o arquivo de configuração.

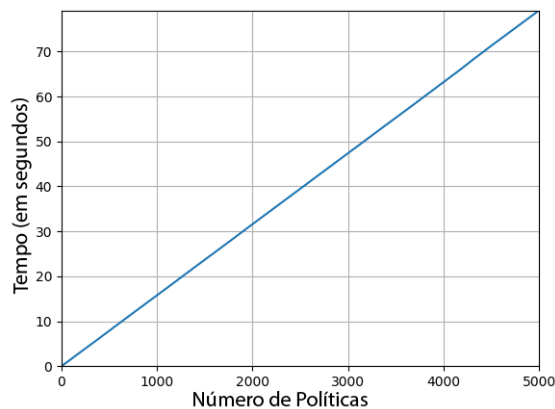


Figura 23 – Gráfico de Processo Completo de Refinamento de Política

Com esses valores de 0,0158s por política para realização de todo processo de refinamento, podemos notar que o tempo de retorno de sucesso ou erro ao usuário não será demorado e as políticas poderão ser modificadas quando houver necessidade, tornando viável a implementação em um ambiente crítico, como é o caso do hospital. Entretanto, é necessário a implementação do requisito de conflito para que a solução possa garantir que o hospital esteja sempre funcionando corretamente, sem maiores problemas. Junto ao tempo, temos a garantia do refinamento com sucesso, permitindo que o usuário possa realizar a criação da política a um nível mais abstrato, como em uma linguagem visual. E com isso certificando que os usuários sejam capazes de criar políticas que serão utilizadas para reconfigurar os dispositivos IoT no hospital, permitindo que o hospital esteja sempre em seu ótimo funcionamento e que o sistema não fique defasado.

Com os resultados descritos, podemos concluir que a solução proposta traz uma contribuição com a arquitetura definida que pode ser utilizada por trabalhos futuros para utilizar o refinador de políticas proposto neste trabalho, como também para usar de base para criação de refinadores para atuar em outras áreas de ambientes inteligentes, como também de informática. Desta forma, mostrando que é possível utilizar refinamento de políticas para sistemas IoT em ambientes inteligente e apresentando também um novo requisito que é o de normas hospitalares. Além disso, as políticas foram modeladas em XML e o seu modelo é apresentado neste trabalho, como também um XSD que pode ser visualizado na Figura 24. Além do modelo XML, também foi criado e apresentado o arquivo CFILE que pode ser utilizado pelo MW de sinais para definir o comportamento dos dispositivos.

Conclusão

O uso de Internet das Coisas tornou factível a implementação de diversos ambientes inteligentes. Com isso, tornou-se importante fornecer maneiras mais fáceis de gerenciar os objetos inteligentes dispostos nestes locais. Por isso, este trabalho propõe uma solução para o gerenciamento de dispositivo IoT em hospitais inteligentes, utilizando refinamento de política. A política é um conjunto de regras que define o comportamento desejado do sistema e seus componentes. Neste trabalho ela foi implementada no formato ECA, onde, a partir de um evento, uma condição é checada e, se satisfeita, desencadeia uma ação. O uso de política para gerenciar o comportamento dos dispositivos IoT permite que os sistemas IoT instalados nos hospitais possam ser mais flexíveis e que os usuários possam alterá-los caso necessário, como também que um mesmo sistema possa ser instalado em diferentes hospitais.

Além disso, uma política pode ter diferentes níveis de abstração, possibilitando que alternativas mais acessíveis de gerenciamento dos dispositivos IoT sejam oferecidas a usuários em diferentes níveis de conhecimento tecnológico. Uma dessas alternativas consiste no uso de linguagens de programação visual mais próximas da linguagem natural. E, para poder traduzir as políticas descritas em mais alto nível de abstração, é preciso utilizar de refinamento de política. O refinamento de política implementado neste trabalho não lida somente com a tradução, mas também com a validação da política em relação aos recursos IoT e da obediência das normas hospitalares em vigor, para que desta forma possa garantir que a política criada pelo usuário será traduzida e validada corretamente.

A proposta foi implementada e para isso foram definidos modelos de política em XML, como também todos os detalhes dos elementos que a compõem (informação, evento, condição e ação), e o modelo do arquivo de configuração. Para garantir que os requisitos do refinamento estavam sendo realizados na proposta, foram realizados testes de execução em diversos cenários. Os testes mostraram que o refinamento está funcionando de forma correta, garantindo a tradução da política desde o arquivo XML até o arquivo de configuração de forma correta e que está mantendo todas as características originais.

Além disso, a validação em relação os recursos IoT e às normas também se mostraram corretas. Ademais, testes de desempenho foram realizados. Eles mostraram que o tempo de resposta do processo de refinamento de uma política é de apenas 0,015s por política, o que evidencia a viabilidade da implementação em um ambiente crítico como o caso do hospital.

Desta dissertação, teve-se como contribuição a publicação do artigo *Using policy refinement to assist IoT device management in Smart Hospitals* na International Conference on Computational Science and Its Applications, como pode-se ver na citação (CASTRO et al., 2019).

Espera-se que os funcionários do hospital sintam-se mais confiantes e que tenham mais facilidade para adotar e gerenciar a tecnologia IoT. Em vista disso, pretende-se desenvolver, em trabalhos futuros, uma aplicação *front-end* e uma linguagem visual que permita os usuários gerenciarem (criando, editando, visualizando) suas próprias políticas. Desta forma, permitindo que testes de usabilidade sejam realizados com funcionários de hospitais. Assim como implementar um cenário hospitalar em que testes com dispositivos reais possam ser realizados. Com relação ao refinamento de política, pretende-se implementar o requisito responsável pela verificação e tratamento dos conflitos de políticas.

Referências

- ABNT. **Iluminância de interiores**. 1991. Disponível em: <<http://ftp.demec.ufpr.br/disciplinas/TM802/NBR5413.pdf>>.
- AHMED, E. et al. Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. **IEEE Wireless Communications**, IEEE, v. 23, n. 5, p. 10–16, 2016.
- Anvisa. **RESOLUÇÃO-RE Nº 9**. 2003. Disponível em: <portal.anvisa.gov.br/documents/33880/2568070/RE_09_2003.pdf>.
- ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. **Computer networks**, Elsevier, v. 54, n. 15, p. 2787–2805, 2010.
- BAKER, S.; XIANG, W.; ATKINSON, I. Internet of Things for Smart Healthcare: Technologies, Challenges, and Opportunities. **IEEE Access**, v. 5, p. 1–1, 2017. ISSN 2169-3536. Disponível em: <<http://ieeexplore.ieee.org/document/8124196/>>.
- BORGIA, E. The internet of things vision: Key features, applications and open issues. **Computer Communications**, Elsevier B.V., v. 54, p. 1–31, 2014. ISSN 01403664. Disponível em: <<http://dx.doi.org/10.1016/j.comcom.2014.09.008>>.
- BOUTABA, R.; AIB, I. Policy-based management: A historical perspective. **Journal of Network and Systems Management**, v. 15, n. 4, p. 447–480, 2007. ISSN 10647570.
- CARNAZ, G. J. F.; NOGUEIRA, V. An Overview of IoT and Healthcare. p. 1–12, 2016. Disponível em: <<http://hdl.handle.net/10174/19998>>.
- CARVALHO, T. C. M. et al. Towards sustainable networks-energy efficiency policy from business to device instance levels. In: **ICEIS (3)**. [S.l.: s.n.], 2012. p. 238–243.
- CASTRO, J. M. de et al. Using policy refinement to assist iot device management in smart hospitals. In: SPRINGER. **International Conference on Computational Science and Its Applications**. [S.l.], 2019. p. 645–657.
- CATARINUCCI, L. et al. An IoT-Aware Architecture for Smart Healthcare Systems. **IEEE Internet of Things Journal**, v. 2, n. 6, p. 515–526, 2015. ISSN 23274662.

- CHEMINOD, M. et al. A comprehensive approach to the automatic refinement and verification of access control policies. **Computers and Security**, Elsevier Ltd, v. 80, p. 186–199, 2019. ISSN 01674048. Disponível em: <<https://doi.org/10.1016/j.cose.2018.09.013>>.
- CHEN, X.; THOMAS, N.; HARRISON, M. Performance evaluation of scheduling policies in a smart hospital environment. In: IEEE. **Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on**. [S.l.], 2011. p. 585–592.
- CHEN, X. et al. Patient Flow Scheduling and Capacity Planning in a Smart Hospital Environment. **IEEE Access**, v. 4, p. 135–148, 2016. ISSN 21693536.
- DHARIWAL, K.; MEHTA, A. Architecture and Plan of Smart hospital based on Internet of Things(IOT). **International Research Journal of Engineering and Technology**, p. 2395–56, 2017. ISSN 2395-0072.
- Ebserh. **CONTROLE DA TEMPERATURA DA GELADEIRA DE MEDICAMENTOS**. 2017. Disponível em: <http://www2.ebserh.gov.br/documents/147715/0/ROP_controle_temp_geladeira.pdf>.
- EDGCOMB, A. D.; VAHID, F. MNFL: the monitoring and notification flow language for assistive monitoring. **Proceedings of the 2nd ACM SIGHIT symposium on International health informatics - IHI '12**, p. 191–200, 2012. Disponível em: <<http://dl.acm.org/citation.cfm?id=2110363.2110387>>.
- HA, M.; LINDH, T. Distributed performance management of Internet of Things as a service for caregivers. **2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)**, p. 1–6, 2017. Disponível em: <<http://ieeexplore.ieee.org/document/8210765>>.
- HUMBLE, J. et al. “playing with the bits” user-configuration of ubiquitous domestic environments. In: SPRINGER. **International Conference on Ubiquitous Computing**. [S.l.], 2003. p. 256–263.
- JAHNKE, J. H.; D’ENTREMONT, M.; STIER, J. Facilitating the programming of the smart home. **IEEE Wireless Communications**, v. 9, n. 6, p. 70–76, 2002. ISSN 15361284.
- KAFLE, V. P.; FUKUSHIMA, Y.; HARAI, H. Internet of things standardization in itu and prospective networking technologies. **IEEE Communications Magazine**, IEEE, v. 54, n. 9, p. 43–49, 2016.
- KOCH, T.; KRELL, C.; KRAMER, B. Policy definition language for automated management of distributed systems. In: IEEE. **Proceedings of IEEE International Workshop on System Management**. [S.l.], 1996. p. 55–64.
- MCKINNEY, W. **Python for Data Analysis: Data Wrangling with Pandas, Numpy, and Ipython**. 1st. ed. [S.l.]: O’Reilly, 2012.
- MIORANDI, D. et al. Internet of things: Vision, applications and research challenges. **Ad Hoc Networks**, Elsevier B.V., v. 10, n. 7, p. 1497–1516, 2012. ISSN 15708705. Disponível em: <<http://dx.doi.org/10.1016/j.adhoc.2012.02.016>>.

- MOHANTY, S. P.; CHOPPALI, U.; KOUGIANOS, E. Everything you wanted to know about smart cities: The Internet of things is the backbone. **IEEE Consumer Electronics Magazine**, v. 5, n. 3, p. 60–70, 2016. ISSN 2162-2248. Disponível em: <<http://eric.ed.gov/ERICWebPortal/recordDetail?accno=EJ548517{\%}5Cnhttp://ieeexplore.ieee.org/document/75392>>.
- MORAIS, I. D. et al. Proposing an IoT-Based Healthcare Platform to Integrate Patients, Physicians and Ambulance Services. v. 10409, p. 188–202, 2017. Disponível em: <<http://link.springer.com/10.1007/978-3-319-62407-5>>.
- PANG, Z. **Technologies and Architectures of the Internet-of-Things (IoT) for Health and Well-being**. Tese (Doutorado) — KTH Royal Institute of Technology, 2013.
- PATCHAVA, V.; KANDALA, H. B.; BABU, P. R. A smart home automation technique with raspberry pi using iot. In: **International Conference on Smart Sensors and Systems**. [S.l.]: IEEE, 2015.
- PERERA, C. et al. Semantic-driven configuration of internet of things middleware. **Proceedings - 2013 9th International Conference on Semantics, Knowledge and Grids, SKG 2013**, n. October, p. 66–73, 2013.
- PONTE, F. R. P.; GOMES, R. L. IoT Device Programmable Language Customization for Home Automation. **2018 IEEE Symposium on Computers and Communications (ISCC)**, IEEE, p. 168–173, 2018.
- QUADRI, S. A. I.; SATHISH, P. Iot based home automation and surveillance system. In: **International Conference on Intelligent Computing and Control Systems**. [S.l.]: IEEE, 2017.
- REISINGER, M. R.; SCHRAMMEL, J.; FRÖHLICH, P. Visual Languages for Smart Spaces : End-User Programming between Data-Flow and Form-Filling. n. 695965, p. 165–169, 2017.
- RIEKSTIN, A. C. et al. A survey of policy refinement methods as a support for sustainable networks. **IEEE Communications Surveys and Tutorials**, v. 18, n. 1, p. 222–235, 2016. ISSN 1553877X.
- SHETE, R.; AGRAWAL, S. Iot based urban climate monitoring using raspberry pi. In: **International Conference on Communication and Signal Processing**. [S.l.]: IEEE, 2016.
- SICARI, S. et al. A policy enforcement framework for Internet of Things applications in the smart health. **Smart Health**, Elsevier, v. 3-4, n. May, p. 39–74, 2017. ISSN 23526483. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S2352648316300435>>.
- SINGH, J.; BACON, J. M. On middleware for emerging health services. **Journal of Internet Services and Applications**, v. 5, n. 1, p. 1–19, 2014. ISSN 18690238.
- SLOMAN, M. Policy driven management for distributed systems. **Journal of network and Systems Management**, Springer, v. 2, n. 4, p. 333–360, 1994.

ULLAH, M.; FIEDLER, M.; WAC, K. On the ambiguity of Quality of Service and Quality of Experience requirements for eHealth services. **2012 6th International Symposium on Medical Information and Communication Technology (ISMICT)**, p. 1–4, 2012. Disponível em: <<http://www.scopus.com/inward/record.url?eid=2-s2.0-84862900161&partnerID=tZOtx>>.

VERMA, D. C. **Policy-based networking: architecture and algorithms**. [S.l.]: New Riders Publishing, 2000.

YU, L.; LU, Y.; ZHU, X. Smart Hospital based on Internet of Things. **Journal of Networks**, v. 7, n. 10, p. 1654–1661, 2012. ISSN 1796-2056. Disponível em: <<http://ojs.academypublisher.com/index.php/jnw/article/view/8492>>.

ZHANG, H. et al. Connecting Intelligent Things in Smart Hospitals using NB-IoT. **IEEE Internet of Things Journal**, v. 4662, n. c, p. 1–11, 2018. ISSN 23274662.

Apêndices

APÊNDICE A

Políticas

A.1 Schema XML da Política

```

1  <?xml version = "1.0" encoding = "UTF-8" ?> <xs:schema xmlns:xs = "http://w3.org/2001/XMLSchema">
2
3  <xs:element name="policy">
4  <xs:complexType>
5    <xs:sequence>
6      <xs:element name="info">
7      <xs:complexType>
8        <xs:sequence>
9          <xs:element name="id_author" type = "xs:integer"/>
10         <xs:element name="id_dep" type="xs:integer"/>
11       </xs:sequence>
12       <xs:attribute name="id" type="xs:integer"/>
13     </xs:complexType>
14   </xs:element>
15   <xs:element name="event">
16   <xs:complexType>
17     <xs:sequence>
18       <xs:element name="at_dep" type="xs:string" />
19       <xs:element name="at" type="xs:string"/>
20       <xs:element name="equip" type="xs:string"/>
21     </xs:sequence>
22     <xs:attribute name="type" type="xs:string"/>
23   </xs:complexType>
24 </xs:element>
25 <xs:element name="condition">
26 <xs:complexType>
27   <xs:sequence>
28     <xs:element name="operation">
29     <xs:complexType>
30       <xs:simpleContent>
31         <xs:extension base="xs:string">
32           <xs:attribute name="op" type="xs:string"/>
33         </xs:extension>
34       </xs:simpleContent>
35     </xs:complexType>
36   </xs:element>
37   </xs:sequence>
38 </xs:complexType>
39 </xs:element>
40 <xs:element name="action">
41 <xs:complexType>
42   <xs:sequence>
43     <xs:element name="do" type="xs:string"/>
44   </xs:sequence>
45 </xs:complexType>
46 </xs:element>
47 </xs:sequence>
48 </xs:complexType>
49 </xs:element>
50 </xs:schema>

```

Figura 24 – XSD da Política

A.2 Políticas em XML

```
<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="1" name="Porta 100">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="new_band">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>None</equip>
  </event>
  <condition>
    <operation op="equal">pulseira;red</operation>
  </condition>
  <action>
    <do>open_door</do>
  </action>
</policy>
```

Figura 25 – Políticas 01 em XML

```
<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="1" name="Porta 100">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="new_band">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>None</equip>
  </event>
  <condition>
    <operation op="or">
      <operation op="equal">band;orange</operation>
      <operation op="equal">band;red</operation>
    </operation>
  </condition>
  <action>
    <do>open_door</do>
  </action>
</policy>
```

Figura 26 – Políticas 02 em XML

```
<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="3" name="Desfibrilador 1">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="new_card">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>defibrillator</equip>
  </event>
  <condition>
    <operation op="equal">id;doctor</operation>
  </condition>
  <action>
    <do>turnon_equipment</do>
  </action>
</policy>
```

Figura 27 – Políticas 03 em XML

```

<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="4" name="Geladeira 100">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="temperature">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>refrigerator</equip>
  </event>
  <condition>
    <operation op="lt">temperatura;2</operation>
  </condition>
  <action>
    <do>change_temperature 2</do>
  </action>
</policy>

```

Figura 28 – Políticas 04 em XML

```

<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="5" name="Taquipnéia e Taquicardia">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="monitoring_vs">
    <at_dep>all</at_dep>
    <at>all</at>
    <equip>None</equip>
  </event>
  <condition>
    <operation op="and">
      <operation op="gt">freqcard;100</operation><operation op="gt">ritmoresp;20</operation>
    </operation>
  </condition>
  <action>
    <do>alarm_rnurse</do>
  </action>
</policy>

```

Figura 29 – Políticas 05 em XML

A.3 Políticas em CFILE

```

<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="6" name="Taquipnéia e Taquicardia e Febre e Idade">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="monitoring_vs">
    <at_dep>all</at_dep>
    <at>all</at>
    <equip>None</equip>
  </event>
  <condition>
    <operation op="or">
      <operation op="and">
        <operation op="gt">freqcard;100</operation>
        <operation op="gt">ritmoresp;20</operation>
      </operation>
      <operation op="and">
        <operation op="gt">temp;38</operation>
        <operation op="lt">idade;2</operation>
      </operation>
    </operation>
  </condition>
  <action>
    <do>alarm_rnurse</do>
  </action>
</policy>

```

Figura 30 – Políticas 06 em XML

```

<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="1" name="Porta 100">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="new_band">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>None</equip>
  </event>
  <condition>
    <operation op="diff">pulseira;red</operation>
  </condition>
  <action>
    <do>open_door</do>
  </action>
</policy>

```

Figura 31 – Políticas 07 em XML

```

<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="4" name="Geladeira">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="temperature">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>geladeira</equip>
  </event>
  <condition>
    <operation op="and">
      <operation op="lt">temperatura;3</operation>
      <operation op="gt">temperatura;5</operation>
    </operation>
  </condition>
  <action>
    <do>change_temp 4</do>
  </action>
</policy>

```

Figura 32 – Políticas 08 em XML

```
<?xml version="1.0" encoding="UTF-8"?>

<policy>
  <info id="1" name="Porta 100">
    <id_author>15</id_author>
    <id_dep>1</id_dep>
  </info>
  <event type="new_band">
    <at_dep>1</at_dep>
    <at>100</at>
    <equip>monitor cardiaco</equip>
  </event>
  <condition>
    <operation op="equal">pulseira;doctor</operation>
  </condition>
  <action>
    <do>equip_on</do>
  </action>
</policy>
```

Figura 33 – Políticas 09 em XML

```
SENSOR_ID: 3
ACTUATOR_ID: 2

when new_band
if (band == 'red'):
    answer = 'open_door'
```

Figura 34 – Políticas 01 em CFILE

```
SENSOR_ID: 3
ACTUATOR_ID: 2

when new_band
if (band == 'red' or band == 'orange'):
    answer = 'open_door'
```

Figura 35 – Políticas 02 em CFILE

```
SENSOR_ID: 6
ACTUATOR_ID: 6

when new_card
if (id == 'doctor'):
    answer = 'turnon_equipment'
```

Figura 36 – Políticas 03 em CFILE

```
SENSOR_ID: 7
ACTUATOR_ID: 8

when temperature
if (temperatura < '2'):
    answer = 'change_temperature (2)'
```

Figura 37 – Políticas 04 em CFILE

```
SENSOR_ID: 7
ACTUATOR_ID: 8

when temperature
if (freqcard > '100' and ritmoresp > '20'):
    answer = 'alarm_nurse'
```

Figura 38 – Políticas 05 em CFILE

```
SENSOR_ID: 7
ACTUATOR_ID: 8

when temperature
if ((freqcard > '100' and ritmoresp > '20') or (temp > '38' and idade < '2')):
|   answer = 'alarm_rnurse'
```

Figura 39 – Políticas 06 em CFILE

```
SENSOR_ID: 12
ACTUATOR_ID: 9

when temperature
if (freqcard > '100' and ritmoresp > '20'):
|   answer = 'alarm_rnurse'
```

Figura 40 – Políticas 07 em CFILE

```
SENSOR_ID: 12
ACTUATOR_ID: 10

when temperature
if ((freqcard > '100' and ritmoresp > '20') or (temp > '38' and idade < '2')):
|   answer = 'alarm_rnurse'
```

Figura 41 – Políticas 08 em CFILE