UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

Rafael Soares Cavalcante

Síntese automática de controladores em ambientes de sensoriamento como serviço

Recife

2017

Rafael Soares Cavalcante

Síntese automática de controladores em ambientes de sensoriamento como serviço

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Glauco Estácio Gonçalves

Recife

2017

À ...

Minha família.

Agradecimentos

Agradeço à Universidade Federal Rural de Pernambuco por ter me dado a oportunidade de realizar este curso.

Agradeço ao meu professor Glauco Gonçalves por toda orientação, pela paciência, dedicação e ensinamentos que possibilitaram que eu realizasse este trabalho.

Agradeço de forma especial ao meu pai Israel Cavalcante da Silva Filho e à minha mãe Carmem Lúcia Soares Cavalcante, por não medirem esforços para que eu pudesse levar meus estudos adiante.

A minha namorada Laura Lobo De Farias que sem o seu carinho e apoio esse trabalho não seria possível.

A todos que direta ou indiretamente fizeram parte da minha formação, até mais e obrigado pelos peixes.

*“**NÃO ENTRE EM PÂNICO**”*

(Douglas Adams)

Resumo

O presente trabalho traz a proposta de uma solução de controle para ambientes inteligentes de sensoriamento como serviço através do uso da teoria o controle supervisório. A coleta de dados ambientais é essencial para diversas áreas tanto para pesquisadores para o desenvolvimento de projetos de pesquisas como para indústrias agrícolas para a otimização de suas culturas. O sensoriamento como serviço vem como uma solução para melhorar a forma atual de como esses dados são capturados, além de prover mecanismos para seu tratamento e armazenamento, e o posterior compartilhamento tanto desses dados como da própria infraestrutura de captura para quem tiver interesse. Para alcançar tal objetivo foi necessário modelar um sistema de eventos discreto e controlá-lo através da teoria do controle supervisório. Para aplicar a teoria do controle supervisório foi necessário um estudo sobre conceitos relacionados à modelagem de sistemas, sensoriamento como serviço, sistemas de eventos discretos, teoria dos autômatos, teoria da linguagem e a própria teoria do controle supervisório. Foi então feita a modelagem de estações de coleta de dados utilizando os conceitos estudados e mostrando através de testes nesse modelo que é possível realizar esse controle para a captura desses dados meteorológicos frente a diversos cenários de funcionamento.

Palavras chaves: Sensoriamento como serviço, Teoria do controle supervisório, Sistemas de eventos discretos.

Abstract

The present work proposes a control solution for intelligent sensing environments as a service through the use of supervisory control theory. Environmental data collection is essential for a number of areas, both for researchers to develop research projects and for agricultural industries to optimize their crops. Sensing as a service comes as a solution to improve the current way in which this data is captured, to provide mechanisms for its treatment and storage, and the subsequent sharing of both that data and the capture infrastructure itself for those who are interested. To achieve this goal, it was necessary to model a discrete event system and control it through supervisory control theory. In order to apply the theory of supervisory control, it was necessary to study concepts related to system modeling, sensing as a service, discrete event systems, automaton theory, language theory and the theory of supervisory control. It was then done the modeling of data collection stations using the concepts studied and showing through tests in this model that it is possible to perform this control for the capture of these meteorological data against different operating scenarios.

Key words: Sensing as a service, Supervisory control theory, Discrete event systems.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

O Brasil ainda não apresenta uma rede de estações meteorológica suficientemente grande para atender as necessidades em todo o país. A concentração de pontos de observação meteorológica está nas áreas mais desenvolvidas e pouquíssimas nas áreas remotas, como no estado do Amazonas. A coleta desses dados meteorológicos é essencial para que pesquisadores possam monitorar, analisar e compreender diversos processos e fenômenos naturais.

A coleta de dados ambientais é de extrema importância tanto para pesquisas, monitoramentos, e análises dos fenômenos naturais, onde, por exemplo, instituições de pesquisas, como universidades, podem se utilizar desses dados para auxiliar o desenvolvimento de pesquisas em múltiplas áreas.

Na agricultura esses dados são especialmente importantes, pois influenciam diretamente na produtividade da plantação, com esses dados é possível também mensurar diversos fatores que influenciam no gerenciamento das atividades na fazenda como a direção predominante dos ventos, a precipitação acumulada na safra, à temperatura média em um período, entre outros.

Dessa forma, monitorar essas variáveis é de suma importância para uma melhor tomada de decisão no campo e o registro em longo prazo dessas variáveis fornece suporte para um planejamento agrícola mais eficaz.

Com isso ainda é possível, por exemplo, fazer um manejo de irrigação, pois através dos dados registrados na estação é possível quantificar as principais formas de entrada e saída de água em determina cultura agrícola. Por esses motivos as informações geradas tornam-se fundamentais para o sucesso agrícola.

A coleta manual destes parâmetros apresenta problemas como a necessidade de trabalho de muitas pessoas, a baixa frequência das coletas e medidas imprecisas e algumas vezes insuficientes. A instrumentação do ambiente através de sensores e dispositivos de armazenamento dos dados coletados soluciona muitos desses problemas.

No entanto, ainda não é comum o uso de um serviço flexível que colete estes dados e os disponibilize para os usuários finais de maneira eficiente. A ausência de um mecanismo de monitoramento remoto também impede que sejam identificadas facilmente, falhas nos sistemas de coleta.

Uma das formas de criar esse mecanismo para a coleta esta no uso do Sensing as a Service (Sensoriamento como um serviço -S^2aaS) que cria esse ambiente em que

os dados são capturados de uma maneira eficiente , além de prover mecanismos para monitoramento, compartilhamento e reconfiguração da infraestrutura das estações.

Para a implementação de um controle preciso para esse sistema que permita a reconfiguração automática dessas estações de acordo com solicitações diversas dos usuários e que faça com que esse sistema funciona da melhor maneira possível mediante a qualquer tipo de cenário, como por exemplo, uma falha em algum dispositivo de captura é aplicado os conceitos da teoria do controle supervisório.

1.1 Sensoriamento como serviço

As empresas que tornam a internet das coisas possível geralmente operam em três modelos de negócios básicos: *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS) e *Software as a Service* (SaaS). Termos como *Database as a Service* (DBaaS), *Data as a Service* (DaaS), *Network as a Service* (NaaS) e *Sensing as a Service* (S^2aaS) são encontrados com menos frequência, mas estes modelos também são importantes.

Para se ter uma ideia sobre o tema, considere o seguinte cenário de uso do S^2aaS abordado nesse trabalho. Imagine vários sensores espalhados por uma grande região capturando dados meteorológicos e os enviando para a nuvem com o propósito de serem armazenados, processados e disponibilizados a quem tiver interesse, dados esses que podem ser utilizados para estudos de recorrência meteorológica e justificativas de operações influenciadas pelas condições atmosféricas.

Vale ressaltar uma característica importante do S^2aaS que faz esse método se destacar diante dos metodos de telemtiria por internet usados autalmente é que ele permite o compartilhamento da infraestrutura das estações de coleta entre múltiplos usuários que dinamicamente reconfiguram essa infraestrutura para atender seus interesses de coleta, como mostra a 1.

Portanto a organização desse processo de S^2aaS envolve pelo menos três partes interessadas:

1. Proprietários de sensores.
2. Organizações que publicam dados de sensores.
3. Terceiros interessados nos dados capturados como empresas, produtores agrícolas, governos, órgãos de pesquisa entre outros.

Essa é a forma de como esse conceito de S^2aaS estaria nesse trabalho, teríamos os proprietários das estações de coletas dos dados, essas estações estariam

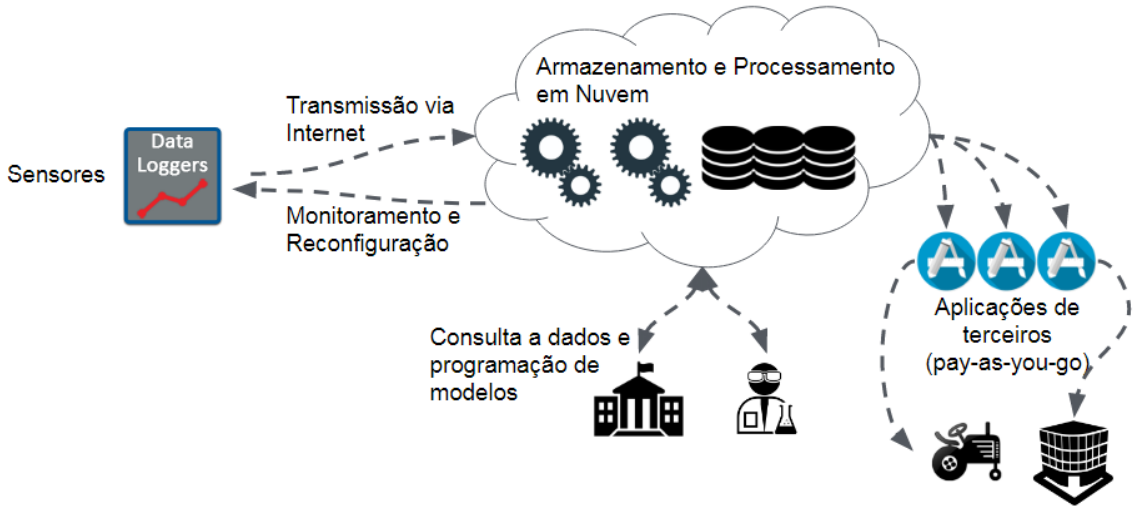

Figura 1 – Modelo de funcionamento do S^2aaS

periodicamente enviando seus dados para a Nuvem a fim de realizar seu armazenamento e processamento. Uma vez na Nuvem esses dados seriam disponibilizados para múltiplos usuários que estejam interessas em observar fenômenos diferentes a partir dos dados dos sensores além de poder se compartilhar a própria infraestrutura de sensores que realizam a captura dos dados.

Desta forma, tendo em vista a oferta de múltiplos sensores, e a capacidade de compartilhamento desses sensores, diferentes usuários podem fazer uso desse serviço, por exemplo, ao mesmo tempo em que um instituto de meteorologia pretende utilizar esse serviço na previsão do tempo em que estaria interessado nos dados dos sensores relacionados ao clima coletando a cada hora uma indústria agrícola poderia querer utiliza esses sensores para algum estudo em que precisariam reconfigura os sensores para coletar dados a cada sete horas.

Além desse compartilhamento da infraestrutura entre múltiplos usuários é possível também realizar o monitoramento da saúde do equipamento a fim de se prevenir de possíveis falhas nos dispositivos.

As principais caraterísticas desse tipo de serviço são:

1. Coleta mais eficiente dos dados.
2. Monitoramento de equipamentos, auxilia na prevenção de falhas.
3. Compartilhamento das estações de coleta entre vários usuários .
4. Captura, armazenamento e processamento dos dados.
5. Reconfiguração da infraestrutura de coleta para atender a interesses pessoais dos usuários.

Existem leituras interessantes que tratam o assunto com maior detalhamento, como por exemplo, (WEISS S. DELAERE, 2010) que foi um dos primeiros trabalhos acadêmicos a usar o termo S^2aaS e aborda uns dos aspectos principais desse serviço que é o compartilhamento de sensores entre clientes diferentes na forma de um serviço. Temos também um projeto da *Commonwealth Scientific and Industrial Research Organisation* (CSIRO) e que trabalha em uma solução IoT aplicada à agricultura (PERERA A. ZASLAVSKY, 2014) o projeto da CSIRO prevê a instalação de uma rede de sensores, implantada em cultivos experimentais, que coletam informação acerca do crescimento das plantações e condições climáticas locais que prevê também o compartilhamento de sua infraestrutura e dados capturados para diferentes atividades de pesquisa.

1.2 Motivações

O trabalho aqui desenvolvido foi influenciado pelos trabalhos *Discrete Control for the Internet of Things and Smart Environments*, de Mengxuan Zhao (ZHAO et al., 2013) transformado, posteriormente, em sua tese de doutorado denominada *Discrete Control in the Internet of things and Smart Environments through a Shared Infrastructure* (ZHAO, 2015).

Em seus trabalhos Zhao discute o controle de ambientes inteligentes através do uso da teoria do controle supervisório, nesse contexto o sistema é modelado como um autômato de modo que se permite observar os estados desse sistema e os eventos associados a cada um desses estados que causa suas transições. A teoria do controle supervisório oferece ferramentas para modelar o sistema e também para sintetizar seu controlador, de modo a atender certas politicas previamente estabelecidas no modelo.

Zhao utiliza a ideia de *smart build*, onde a partir dessa teoria ela pretende controlar os dispositivos que compõe o sistema como lâmpadas, ar condicionados, computadores através do uso da teoria do controle supervisório.

Zhao em seus trabalhados aborda como a teoria do controle supervisório pode ser empregada para modela e controlar ambientes inteligentes, a autora mostra como é possível modelar cada um dos dispositivos e as políticas de gerenciamento do ambiente por meio de sua respectiva máquina de estados. Em seguida determinam um conjunto de regras como objetivos de controle os quais servirão de base para sintetizar automaticamente o controlador.

O objetivo da autora para essa abordagem é criar um sistema em que ao mesmo tempo em que provê conforto personalizado aos moradores, como temperatura e iluminação adequada para cada tipo de cenário diferente, também evite altos valores no consumo de energia. Para se alcançar esse objetivo é necessário um ajuste preciso

dos dispositivos que compõem o espaço inteligente de modo a atender às políticas definidas no modelo.

Tendo como base os trabalhos de Zhao, esse trabalho se propõe em controlar ambientes inteligentes por meio da teoria do controle supervisório, nesse caso o controle automático de estações inteligentes de coleta de dados atmosféricos. Considere o seguinte exemplo para o melhor entendimento básico do problema de controle.

Imagine um prédio que possui um ar condicionado inteligente, ou seja, foi feito um código especifico para que esse aparelho seja controlado de maneira automática, por exemplo, caso esteja muito calor ligue na menor temperatura ou se estiver muito frio não ligue o ar condicionado.

Tendo em vista esse cenário construir um controlador para um único aparelho não parece uma tarefa tão difícil. Porém agora imagine que ao em vez de um de um aparelho temos vários aparelhos de ar condicionados nesse prédio e todos precisam ser controlados.

Quando se pensa em fazer um controlador específico para cada aparelho de ar condicionado presente no prédio, já se percebe que não é uma boa escolha, muitos dispositivos, diferentes cenários, interação entre esses aparelhos causam um grande problema para se gerar um controle.

Em ocasiões em que o sistema possui muitos dispositivos e cenários diferentes torna-se custoso a implementação de códigos específicos para o controle de cada um desses dispositivos, quando se adiciona ou retira dispositivos o custo em código para essas operações é muito alto, então qual seria a melhor opção para fazer esse controle?

Neste projeto pretende-se utilizar a teoria do controle supervisório como solução para controle de dispositivos. Pode-se enxergar a configuração de dispositivos como um problema de teoria de controle em que a planta (sistema a ser controlado) automaticamente modifica seus estados internos (configurações) seguindo os sinais de um controlador com base na referência informada pelos usuários do sistema.

A teoria do controle supervisório oferece uma solução para esse problema de controle, essa teoria nos da uma gama de recursos para a criação de controladores de maneira automática, além de permitir a adaptação da programação para o funcionamento em diferentes cenários. Apesar de dispositivos controláveis já existirem, a integração entre eles e a reação a eventos ainda é algo estudado por pesquisadores.

É esse o objetivo desse trabalho, a partir da teoria do controle supervisório sintetizar controladores de uma maneira automática, a fim de se controlar dispositivos em ambientes de S^2aaS.

A ideia consiste em criar um ambiente de S^2aaS que é compostos por sensores e estações de coleta de dados onde se pretender controlar os dispositivos através do uso da teoria do controle supervisório.

O mais importante dessa nova abordagem é a presença de um mecanismo que identifica e resolve facilmente cenários não desejados, como por exemplo, uma falha na estação que impossibilite a coleta desses dados a partir dessa estação.

1.3 Objetivos do trabalho

Esse trabalho tem como objetivo principal Síntese automática de controladores em ambientes de sensoriamento como serviço, para que isso seja possível têm-se alguns objetivos secundários que precisam ser cumpridos.

Como objetivos secundários a fim de se alcançar a síntese automática de controladores têm-se:

- Modelagem dos sensores e das estações de coleta de dados como um sistema de eventos discretos.
- Estudo sobre o sistema a ser modelado para a criação dos diferentes tipos de cenários de funcionamento e implementação das regras de controle.
- Estudo sobre a aplicação da teoria do controle supervisório e conceitos relacionados.
- Sintese do controlador de acordo com as políticas criadas para o modelo.

Tendo em vista esses objetivos o trabalho foi desenvolvido da seguinte forma: Modelagem, síntese e testes, como mostra a figura 2.

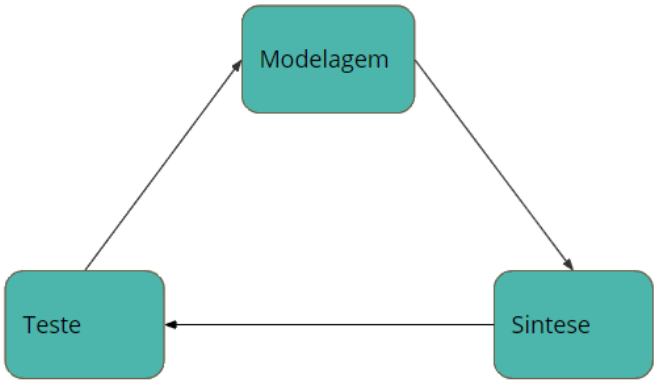

Figura 2 – Desenvolvimento do trabalho.

Na etapa da modelagem foram criados os autômatos para a representação dos subsistemas dos sensores e das estações e logo em seguida as regras de controle para esses subsistemas que irão definir o comportamento dos dispositivos e como eles vão interagir entre si.

Na etapa de síntese ocorre à composição paralela desses autômatos e a criação do controlador com base nas regras previamente estabelcidas para o sistema.

E por fim a etapa de teste do modelo que serve para observar se todos os componentes do sistema estação funcionado da maneira desejada, se o controlador esta garantindo o funcionamento de todas as regras, caso alguma coisa esteja errada volta-se para a etapa da modelagem.

A figura 3 a baixo mostra o que e necessário para que ocorra a sintese do controlador.

O modelo das coisas que são os modelos individuais dos subsistemas da estação e dos sensores, as regras de controle que vão definir como esses subsistemas funcionam e interagem entre si e o modelo do ambiente que é a estação de coleta em si, ou seja, é composta pelo modelo da estação mais o modelo dos sensores, tudo isso tem como objetivo de gerar automaticamente o controlador que vai ter a visão global do sistema e decidir como manter as regras modificando os estados de funcionamento. Com base nisso os componentes do modelo são:

1. Estações de coletas de dados meteorológicos.
2. Sensores de captura dados.
3. Usuários.
4. Controlador.

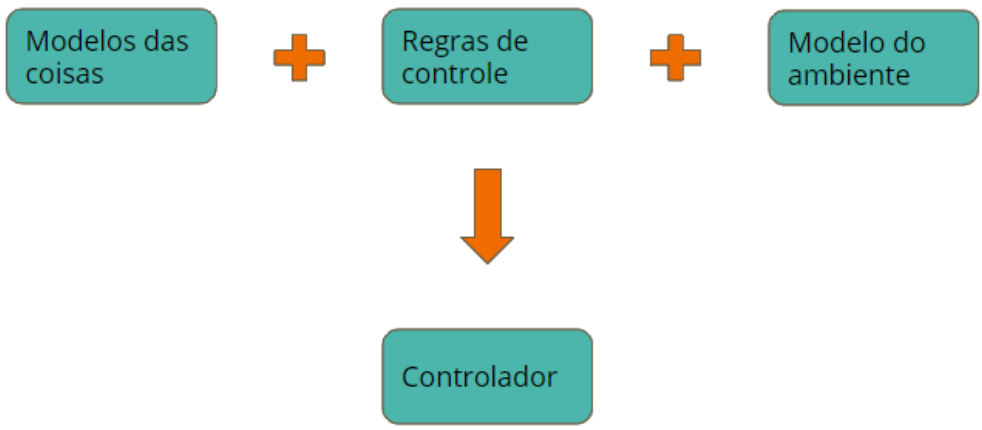

Figura 3 – Elementos para realizar a síntese do controlador.

2 Teoria do controle supervisório

2.1 Introdução

Antes de entrar nos detalhes do capítulo, é necessário descrever o que se entende por ”sistema”. Nesse capítulo pretende-se apresentar conceitos associados a sistema e modelagem com a finalidade do melhor entendimento da teoria do controle supervisório.

Comumente pesquisadores, cientistas tem seu foco em fenômenos naturais que são bem modelados, que possuem variáveis “continuas”, ou seja, são aquelas variáveis que evoluem com o decorrer do tempo, como por exemplo, a aceleração, a velocidade.

Esse tipo de sistema já possui uma série de recursos, como por exemplo, as equações diferencias da matemática, que já resolvem grande parte do problema de modelagem desse tipo de sistema.

Porém quando se pensa no mundo real, no dia a dia, nota-se que os problemas não são tão bem definidos, que as variáveis desses problemas não são de caráter contínuo, grande parte dos problemas do mundo real possuem variáveis “discretas” e têm como caraterísticas serem instantâneo, as quais as equações diferencias não se aplicam, os conceitos de variáveis “discretas” e “contínuas” serão aprofundadas durante o capítulo.

2.2 Noções básicas de controle do sistema

Nesta seção, vamos introduzir com cuidado os conceitos básicos de teoria dos sistemas. A seguir, vamos identificar os critérios fundamentais pelos quais sistemas se distinguem e classificá-los.

2.2.1 O conceito de sistema

Com relação ao conceito de sistema, na literatura, destacam-se três definições. São as seguintes:

1. Uma agregação ou conjunto de coisas de modo combinado pela natureza ou pelo homem para formar um todo integral ou complexo (*Encyclopedia Americana*).

2. Um grupo regular independente ou interativo que formam um todo unificado (*Websters Dictionary*).

3. Uma combinação de componentes que agem em conjunto para executar uma função que não é possível para as partes (*IEEE Standard Dictionary of Electrical and Electronic Terms*).

Quando se ler essas definições nota-se que existem duas características marcantes com relação aos sistemas. São elas:

1. Um sistema consiste de uma interação de ”componentes”

2. Um sistema está associado com uma ”função”que é previamente destinado a desempenhar.

Neste trabalho, o estudo dos sistemas tem por base representá-los através de modelos formais que permitam mostrar de forma satisfatória seu comportamento.

Para que o sistema desempenhe corretamente sua função, é necessário garantir que esse sistema ao qual se deseje modelar, funcione como desejado, ou seja, que seu comportamento não viole determinadas especificações, para isso é necessário implementar leis de controle que atuem sobre o sistema para que seu comportamento seja o desejado ou mais próximo disso, nesse contexto é onde se aplica a teoria do controle supervisório apresentada nesse trabalho.

2.2.2 O processo de modelagem Entrada e Saída

Como os cientistas, pesquisadores, engenheiros estão principalmente preocupados com o desenvolvimento de técnicas para a medição e controle dos sistemas, as definições dadas acima puramente qualitativas são inadequadas. Em vez disso, buscamos um modelo de um sistema real. Para ser mais preciso, precisamos desenvolver alguns meios matemáticos para descrever esse comportamento.

Para iniciar o processo de modelagem, começamos definindo um conjunto de variáveis mensuráveis associados a um determinado sistema. Por exemplo, as posições de partículas e velocidades, ou tensões e correntes em um circuito elétrico, que são todos os números reais.

Ao medir essas variáveis ao longo de um período de tempo $[t_0, t_f]$ podemos então coletar dados e sem seguida selecionar um subconjunto dessas variáveis e as-

sumir que se tem a capacidade de variar ao longo do tempo. Isso define um conjunto de funções que chamaremos as variáveis de entrada.

$$\{u_1(t), ..., u_p(t)\}t_0 \leq t \leq t_f \tag{2.1}$$

Em seguida, selecione outro conjunto de variáveis que supomos que podemos medir diretamente, enquanto varia-se $u_1(t), ..., u_p(t)$, e, assim, definir um conjunto de variáveis de saída.

$$\{y_1(t), ..., y_m(t)\}t0 \leq t \leq tf \tag{2.2}$$

Estas podem ser pensadas como a ”resposta”para o ”estímulo”fornecido pelas funções de entrada selecionadas. Note-se que pode haver algumas variáveis que não tenham sido associadas com a entrada ou a saída, estes são por vezes referidos como variáveis de saída suprimidas.

Para simplificar a notação, que representam as variáveis de entrada através de um vector coluna $u(t)$ e as variáveis de saída por meio de outro vetor coluna $y(t)$, para breve, nos referimos a eles como a entrada e saída, respectivamente. Assim, vamos escrever:

$$u(t) = [u_1(t), ..., u_p(t)]^T \tag{2.3}$$

Onde $[.]^T$ indica a transposição de um vector, e, de modo semelhante,

$$y(t) = [y_1(t), ..., y_m(t)]^T \tag{2.4}$$

Para completar um modelo, é razoável postular que existe alguma relação matemática entre a entrada e a saída. Assim, assumimos que podemos definir a função:

$$y_1(t) = g_1(u_1(t), ..., u_p(t)), ..., y_m(t) = g_m(u_1(t), ..., u_p(t)) \tag{2.5}$$

E obter o modelo de sistema na forma matemática:

$$y = g(u) = [g_1(u_1(t), ..., u_p(t)), ..., g_m(u_1(t), ..., u_p(t))]^T \tag{2.6}$$

Onde $g(\cdot)$ denota o vetor da coluna cujas entradas são as funções $g_1(\cdot), ..., g_m(\cdot)$. Este é o mais simples possível processo de modelagem, e é ilustrado na 4. Muitas vezes, o modelo apenas aproxima o verdadeiro comportamento do sistema. No entanto, uma vez que estamos convencidos de ter obtido um modelo ”bom”, esta distinção é normalmente descartada, e o modelo utilizado.

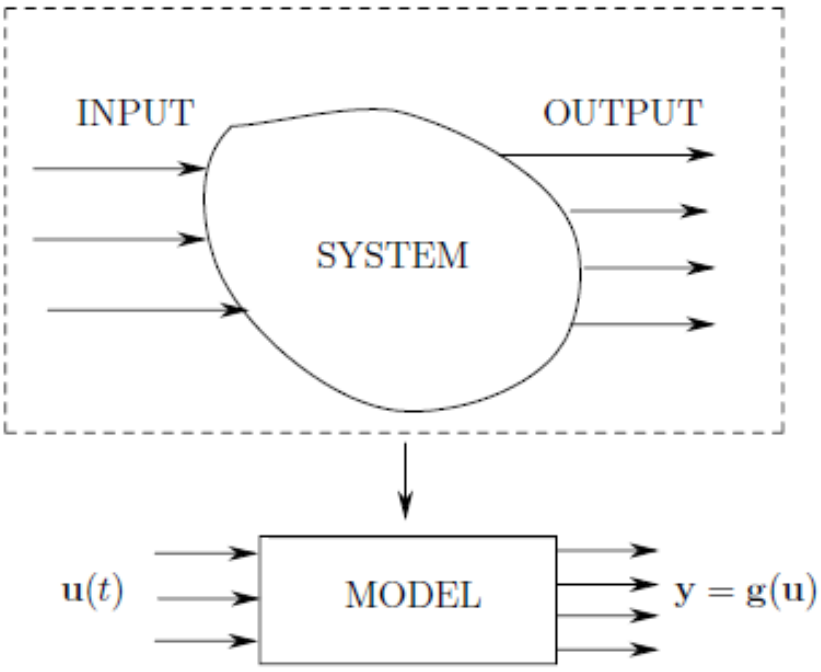

Figura 4 – Processo de modelagem simples. Fonte: (CASSANDRAS; LAFORTUNE, 2008)

É importante ressaltar uma observação em relação a modelos. Para qualquer dado sistema, em principio é sempre possível se obter um modelo, já o inverso não é verdade, pois equações matemáticas nem sempre produzem soluções reais.

É importante enfatizar a flexibilidade do processo de modelagem, uma vez que não existe nenhuma forma única para selecionar as variáveis de entrada e saída. Assim, é tarefa do modelador identificar estas variáveis.

2.2.3 O Conceito de Estado

De modo geral, o estado de um sistema em um determinado tempo t deve descrever o seu comportamento naquele instante, de alguma forma mensurável. Em teoria do sistema, o termo estado tem um significado muito mais preciso e constitui a pedra angular do processo de modelagem e de muitas técnicas analíticas.

Definição: O estado de um sistema no momento t_0 representa a informação necessária no t_0 tal que a saída $y(t)$, para todos $t \geq t0$, é determinado exclusivamente a partir desta informação e de $u(t)$, $t \geq t0$. (CASSANDRAS; LAFORTUNE, 2008)

Nos sistemas em que estamos interessados nesse trabalho, os sistema de eventos discretos, as variáveis de estado variam bruscamente em instantes determinados, e seus valores nos estados seguintes podem ser calculados diretamente a partir de valores prévios sem ter que considerar o tempo entre esses instantes.

Com relação aos estados possíveis que um sistema pode assumir tem-se o conceito de espaço de estado, que possui a seguinte definição:

Definição. O espaço de estado de um sistema, geralmente denotado por x, é o conjunto de todos os possíveis valores que o Estado pode tomar.(CASSANDRAS; LAFORTUNE, 2008)

Com relação às variáveis e seus valores, quando se fala em um sistema de eventos discreto, não existe nenhuma garantia que essas variáveis vão sempre assumir valores reais, o que seria facilmente resolvido com modelos baseados em equações diferencias.

Em oposição a esses valores temos valores de um determinado conjunto discreto como, por exemplo, {Ligado, Desligado}, {Alto, Médio, Baixo} ou {Verde, Vermelho, Azul}. O comportamento dinâmico desses sistemas de estados discretos é mais simples de visualizar porque suas transições são baseadas em declarações logicas como, por exemplo: ”se algo específico acontece e o estado atual é x_0, então o próximo estado se torna x_1”.

Porém todo o mecanismo por trás, para expressar formalmente as equações de estados e as resolver é por vezes muito mais complexos que os modelos contínuos que se reduz a análise de equações diferencias, onde já existe várias soluções e técnicas matemáticas para se resolver. Sistema de variáveis continuas e de variáveis discretas vão ser melhor explicados no decorrer do capítulo.

2.2.4 O Conceito de controle

A questa básica vista ate agora foi: Qual a saída de um sistema para uma dada entrada? Sistemas, no entanto, normalmente não existem no vácuo. A própria definição de sistema contém a ideia de realizar uma determinada função. Porém para que essa função seja realizada o sistema precisa ser controlado de forma que atinja algum ”comportamento desejado”. Assim, a entrada para um sistema é muitas vezes visto como um sinal de controle destinado a obter um comportamento desejado.

Associado a isso existe a ideia de feedback que é intuitivamente simples: Use qualquer informação disponível sobre o comportamento do sistema a fim de ajustar continuamente a entrada de controle. A propriedade chave de feedback é fazer correções, especialmente na presença de perturbações.

2.3 Sistemas de eventos discreto

Em (CASSANDRAS; LAFORTUNE, 2008) é realizada a classificação dos sistemas em algumas categorias. Essa classificação não é excludente, pois depende basi-

camente da perspectiva empregada para interpretar e compreender o sistema. Temos uma classificação que divide os sistemas nos chamados:

- Sistemas dinâmicos a variáveis contínuas: São um tipo de sistema que se caracteriza basicamente por dois fatores.

    1. O espaço de estados é contínuo, isto é, as variáveis do sistema podem assumir qualquer valor dentro de um determinado intervalo de variação contínuo.
    2. O comportamento das variáveis do sistema é regido pelo tempo.

De forma geral, o estudo desse tipo de sistema se baseia em equações diferenciais. O que não funciona para o outro tipo de sistema apresentado, que seria uma contrapartida aos sistemas de variáveis continua denominado de:

- Sistemas dinâmicos a eventos discretos: Esse tipo de sistema que se caracteriza pelos seguintes fatos:

    1. O espaço de estados é discreto, ou seja, as variáveis do sistema podem assumir valores pré-estabelecidos pertencentes a um conjunto discreto.
    2. O comportamento das variáveis independe do tempo e é dirigido por eventos.

Existem várias boas razões para se querer adotar essa abordagem discreta:

1. Qualquer computador digital que pode ser usado como um componente de um sistema que opera de modo em tempo discreto, isto é, ele é equipado com um relógio interno de tempo discreto. Seja qual for as variáveis o computador reconhece ou os controles são avaliados somente naquele instante de tempo correspondente a ”ticks de relógio”.
2. Muitas equações diferenciais de interesse em modelos de tempo contínuo só podem ser resolvidas numericamente através da utilização de um computador. Tais soluções geradas por computador são realmente versões de tempo discreto de funções de tempo contínuo. Portanto, é razoável começar com modelos de tempo discreto se as soluções definitivas vão ser nesta forma de qualquer maneira.
3. Técnicas de controle digital, que são baseados em modelos de tempo discreto, muitas vezes fornecem uma considerável flexibilidade, velocidade e baixo custo. Isto é por causa dos avanços em hardware digital e tecnologia dos computadores

4. Alguns sistemas são inerentemente sistemas de tempo discreto, tais como os modelos com base nos dados econômicos que é registrado apenas em intervalos discretos regulares

Esse tipo de sistema percebe as ocorrências no mundo externo através da recepção de estímulos, esse estímulos são o que chamamos de eventos. São exemplos de eventos o início e o término de uma tarefa (mas não sua execução), a chegada de um cliente a uma fila ou a recepção de uma mensagem em um sistema de comunicação, a ocorrência do evento causa uma mudança interna no sistema que pode ser ou não ser percebida a um observador externo.

Uma das principais características dessas mudanças é que elas são abruptas e instantâneas: ao perceber um evento, o sistema reage se adaptando instantaneamente a esse novo cenário e acomoda-se em tempo nulo numa nova situação, onde pode ficar por tempo indeterminado, que é uma das características desse tipo de sistema ele não depende do tempo para evoluir, ele irá permanecer nesse novo estado até que algum outro evento o retire de lá, ou seja, a simples passagem do tempo não é o suficiente para garantir que o sistema evolua, para tanto, é necessário que ocorram eventos.

Vale ressaltar também que esses eventos podem depender de fatores alheios ao próprio sistema, os que torna, em geral, impossível de se prever quando esse tipo de evento poderá ocorrer. Cabe ao sistema perceber a ocorrência desse cenário e tratar esse evento, de acordo com regras (vamos falar mais dessas regras no próximo capítulo) previamente estabelecidas, e acomodar o sistema em algum novo estado.

Um bom exemplo para visualizar esse tipo de cenário é quando ocorre um evento de falha de algum dispositivo, esse tipo de cenário é tratado nesse trabalho e é apresentado com uma profundidade maior no próximo capítulo. A falha é um evento que o sistema não tem como prever quando irá ocorrer, porém ele sabe o que fazer quando esse evento ocorrer, o sistema é capaz de se adaptar a esse cenário e tratar esse evento de acordo com regras previamente inseridas no sistema, alocando o sistema em outro estado, por exemplo, um estado de falha.

De acordo com o que foi mostrado, de maneira geral, pode-se de se identificar os seguintes pontos, como sendo características principais de um sistema de eventos discretos:

1. A ocorrência de eventos é assíncrona no tempo.
2. O estado do sistema permanece imutável até que ocorra um evento.
3. Para um dado estado do sistema, a ocorrência de um determinado evento não implica necessariamente a mudança de estado.

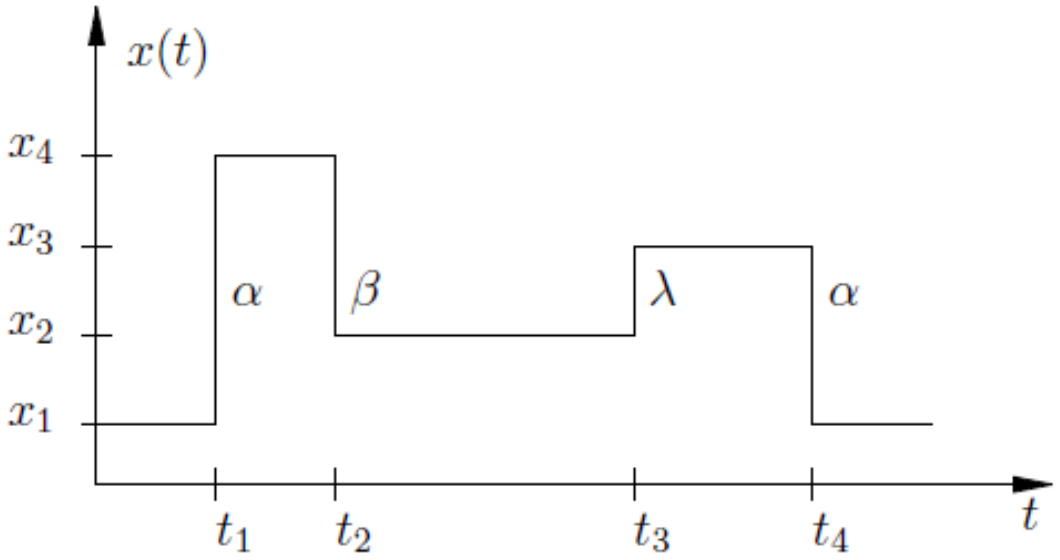

Figura 5 – Trajetória tipica de um sistema a eventos discretos
(CURY, 2001)

Definição: Sistema a eventos discretos (SED) é um sistema dinâmico que evolui de acordo com a ocorrência abrupta de eventos físicos, em intervalos de tempo em geral irregulares ou desconhecidos (CURY, 2001).

A ocorrência de um evento causa então uma transição ou mudança de estado no sistema, de forma que sua evolução no tempo pode ser representada pela trajetória percorrida no seu espaço de estados, conforme ilustrado na figura 14.

Nesta trajetória ocorrem eventos representados pelas letras α, β e λ, o sistema muda seu estado de acordo com a ocorrência desses eventos. Por exemplo, no tempo $t1$ α ocorre e faz com o que o sistema em questão mudeo do estado $x1$ para o estado $x4$

É importante notar nesse gráfico que a ocorrência de α pode levar o sistema a dois estados diferentes, o próximo estado ao qual o sistema vai depende do evento e do estado em que o sistema estava previamente a ocorrência desse evento, portanto existe casos em que um mesmo evento pode levar o sistema a diferentes estados. A trajetória pode continuar indefinidamente, porém, assume-se que o número total de eventos diferentes que podem ocorrer é finito.

Isso nos permite ver a tarefa de especificar o comportamento de um sistema a eventos discretos como sendo a de estabelecer sequencias ordenada de eventos que levem à realização de determinados objetivos. Como já foi visto, as equações diferencias que funcionam para sistemas contínuos não funcionam para esse tipo de sistema justamente por sua natureza discreta, tendo isso em vista e da importância do tema, foram desenvolvidos vários modelos para tratar SEDs, sem que nenhum tivesse

se afirmado como universal. Os principais modelos utilizados para sistemas a eventos discretos são os seguintes:

- Redes de Petri com e sem temporização.
- Redes de Petri Controladas com e sem temporização.
- Cadeias de Markov.
- Teoria das Filas.
- Lógica Temporal e Lógica Temporal de Tempo Real.
- Teoria de Linguagens e Autômatos (Teoria do controle supervisório).

Dentre os modelos citados acima, dois se destacam por apresentar uma característica em particular: São dotados de procedimentos de síntese de controladores. São eles:

- Os modelos de Redes de Petri.
- Teoria de Linguagens e Autômatos (Teoria do controle supervisório).

Vale ressaltar que também existe uma abordagem que permite considerar simultaneamente as características contínuas e discretas de seus componentes e de suas inter-relações. Segundo esta abordagem os sistemas podem ser classificados como “Sistemas Híbridos”.

2.3.1 O conceito de evento

Uma das características importante de um sistema de evento discreto, é que se for conhecido o seu estado inicial, o comportamento do sistema pode ser representado através de uma sequência de eventos. O sistema no qual estamos interessados em modelar nesse trabalho é um sistema de eventos discreto e possui essa característica, ou seja, é um sistema baseado em eventos, como já foi dito na seção anterior esse tipo de sistema percebe as ocorrências no mundo externo através da recepção de estímulos, esses estímulos são os nossos eventos, é importante enfatiza que esses eventos devem ser pensados como algo que ocorre instantaneamente e causa uma transição entre estados. Esses eventos podem ser divididos em dois tipos:

- Eventos controláveis: São aqueles eventos que podem ser habilitados, ou desabilitados, pelo controle supervisório.
- Eventos não controláveis: Todo o resto, ou seja, aqueles eventos que não são controláveis pelo controle supervisório

2.4 Autômatos e linguagens formais como modelos para sistemas de eventos discretos

A ideia dessa seção é mostrar como o comportamento lógico de um SED pode ser modelado a partir da teoria de linguagens formais e autômatos. Como ponto de partida para esse estudo podemos perceber o fato de que qualquer SED tem um conjunto E de eventos subjacente associado a ele. O conjunto E é pensado como o ”alfabeto”de uma linguagem e as sequências de eventos são pensadas como ”palavras”nessa língua. Para visualizar melhor essa ideia, imagine uma máquina que depois de ligada deve emitir um sinal para confirmar que está ligada, um sinal para informar seu status e outro sinal para concluir. Cada um desses sinais define um evento e todos os possíveis sinais que a máquina pode emitir, definem um alfabeto.

A combinação desses sinais pode ser pensado como a linguagem da máquina, dependendo dessa combinação o SED percebe se a máquina está funcionando corretamente ou requer alguma atenção especial, portanto o SED é responsável por reconhecer eventos e dar a interpretação apropriada a qualquer sequência particular recebida.

Com isso o comportamento de um SED pode ser visto através da sequência de eventos gerados, e se considerarmos que eventos representam elementos de um alfabeto, como foi exemplificado, e que sequências de eventos representam palavras sobre este alfabeto, pode-se descrever o comportamento do sistema através de uma determinada linguagem.

Definição: Uma linguagem definida sobre um conjunto de eventos E é um conjunto de strings de comprimento finito formado a partir de eventos em E. (CASSANDRAS; LAFORTUNE, 2008)

O uso da teoria da linguagem é bastante atrativa quando se quer apresentar os aspectos do modelo e discutir as propriedades de SEDs, entretendo, é importante ressaltar que ela não é adequada para se realizar, por exemplo, verificação das propriedades ou a síntese do controlador. É necessário que a linguagem empregada seja representa de uma forma conveniente, se a linguagem for finita é sempre possível listar todos os seus elementos, ou seja, todas as possíveis sequências de eventos.

Infelizmente isto raramente é factível. Para representar linguagens em SEDs utilizasse autômatos, pois a partir do uso da teoria dos autômatos é possível representar a linguagem de uma forma em que é possível destacar a estrutura do comportamento do sistema e ainda é cabível de se manipular quando se deseja realizar a verificação de propriedades e a síntese do controlador.

2.4.1 Teoria da linguagem

Os sistemas de eventos discretos são regidos por eventos que sinalizam o início ou fim de uma atividade. No desenvolvimento destas atividades ocorre uma série de eventos que contribuem para que o sistema alcance um novo estado. Neste sentido, a representação da alternância ou sequência de eventos através de linguagens tem o significado de descrever o comportamento dos sistemas a eventos discretos (BRANDIN; MALIK; MALIK, 2004). Para Cassandras e Lafortune (CASSANDRAS; LAFORTUNE, 2008), uma das maneiras formais de estudar o comportamento lógico de um SED é baseada na Teoria de Linguagens.

Dado um conjunto de eventos distintos Σ como o alfabeto de um SED, e assumindo que Σ seja finito. Entende-se por Σ^* o conjunto de todas as palavras possíveis e factíveis formadas pelos eventos constituintes deste alfabeto. Uma sequência de eventos sobre este alfabeto forma uma ”palavra”. Uma palavra que consiste apenas do evento ϵ é chamada de “palavra vazia”.

Uma linguagem definida sobre um alfabeto Σ é um conjunto de palavras de comprimento finito sobre este alfabeto.

Seja $\Sigma = \{a, b, g\}$ o conjunto de eventos, define-se a linguagem $L_1 = \{, a, abb\}$ que consiste de três palavras. Seja $s \in \Sigma^*$ sendo $s = tuv$ com $t, u, v \in \Sigma^*$, então:

- t é chamado de prefixo de s.
- u é chamado de subpalavra de s.
- v é chamado de sufixo de s.

Cassandras e Lafortune (CASSANDRAS; LAFORTUNE, 2008) também mostram que o processo de análise das linguagens se desenvolve de acordo com um conjunto de operações. Tais operações são em função do alfabeto de eventos característicos de cada sistema, em seu livro os autores definem três operações sobre linguagens que são:

- O Fechamento de Kleene de $L \cup \Sigma^*$ é definido como a união de todas as possíveis palavras, formadas pelos símbolos pertencentes a um alfabeto, incluindo a palavra vazia ϵ
- A concatenação de duas linguagens $L_a, L_b \cup \Sigma^*$, é definida como a justaposição de duas linguagens, dando origem a uma linguagem maior, que é composta dos símbolos pertencentes a primeira linguagem, imediatamente seguidos dos símbolos pertencentes a segunda linguagem

- prefixo − fechamento de uma linguagem $L \cup \Sigma^*$ é definido como o conjunto de todos os prefixos de uma dada linguagem incluindo a palavra vazia ϵ .

2.4.2 Autômatos como modelo para SED's

Como forma de representar linguagens através de transições de estados nos usamos os conceitos de autômatos, sendo possível ainda com isso, analisar comportamento, verificar propriedades e sintetizar controladores.

É possível também, através da composição de autômatos, representar o sistema através da composição dos vários subsistemas que o compõe. Com relação aos autômatos utilizados para modelar um SED, um ponto importante a se destacar é que nesse contexto não nos preocupamos se o sistema vai entrar em determinado estado, ou quanto tempo ele passa em cada estado, a preocupação é com a sequência de estados e os eventos associados que causam essas transições.

Tendo isso em vista para a compreensão da teoria do controle supervisório, destacam-se as teoria de linguagens formais, expressões regulares e autômatos. Tendo em vista que o objetivo do trabalho é o emprego da teoria do controle supervisório e a extensão destes assuntos, será apresentado apenas seus conceitos fundamentais.

A Teoria de Controle Supervisório representa o comportamento livre de um SED através de um autômato na forma $(Q, \Sigma, \delta, q_0, Q_m)$, onde:

- Q – representa o conjunto de estados utilizados para descrever o comportamento do sistema segundo a abstração empregada.
- Σ − representa o conjunto de eventos relevantes, usualmente denominado alfabeto de eventos.
- q_0 – representa o estado inicial do sistema, sendo que $q_0 \in Q$.
- $\delta : Q \times \Sigma \rightarrow Q$ − representa a função de transição de estados, sendo que esta função pode ser definida para apenas alguns pares ordenados em Q.
- Q_m – representa um conjunto de estados marcados, sendo que $Q_m \subseteq Q$.

Emprega-se a notação $\delta(q, \sigma)$ para especificar que a função é definida para o par ordenado (q, σ). A notação $\neg\delta(q, \sigma)$ é utilizada em caso contrário.(VIEIRA, 2007).

O autômato aqui ilustrado possui uma ideia diferente da ideia tradicional de autômato finito determinístico, aqui o autômato possui uma função de transição parcial. Em contraposição aos autômatos tradicionais, esses autômatos têm a função de transição definida apenas para um subconjunto de eventos em cada estado. Em seds,

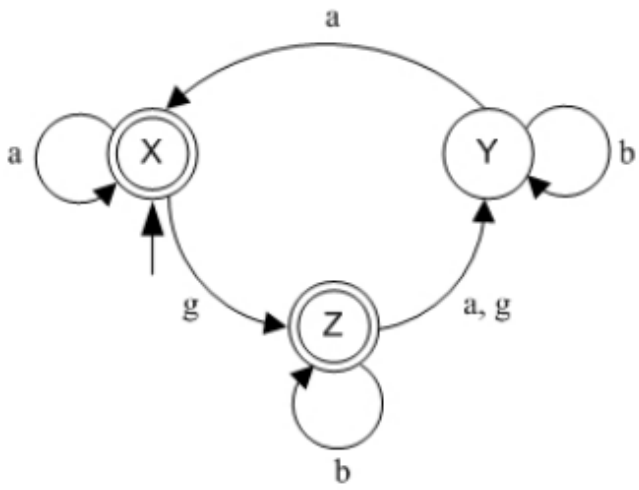

Figura 6 – Exemplo de autômato deterministico, Fonte: (CASSANDRAS; LAFORTUNE, 2008)

com já foi citado, é bastante utilizado a teoria dos autômatos para representar linguagens. Quando o número de estados do autômato é finito é possível realizar sua representação através de uma maquina de estados que aceita um conjunto particular de palavras sobre um determinado alfabeto Σ. Dado o alfabeto de entrada Σ, para um determinado autômato finito G, todos os símbolos sucessivos em uma palavra causará uma mudança de estado. Ha somente um estado associado para cada transição em cada combinação de símbolos $e \in \Sigma$, e um estado $q \in Q$.

Considere o grafo da figura 6 retirado de Cassandra e Lafortune (CASSANDRAS; LAFORTUNE, 2008) onde os nós representam os estados e os arcos representam as transições.

O conjunto de nós que representa os estados do sistema, $Q = \{x, y, z\}$ e as transições são os eventos (alfabeto) do autômato, $\Sigma = \{a, b, g\}$. Os arcos no grafo dão uma representação gráfica da função de transição, que se denota como $\delta : Q \times \Sigma \rightarrow Q$:

- $\delta(x, a) = x$; Se o autômato estiver no estado x e acontecer a continue em x.
- $\delta(x, g) = z$; Se o autômato estiver no estado x e acontecer g faça uma transição para z.
- $\delta(y, a) = x$; Se o autômato estiver no estado y e acontecer a faça uma transição para x.
- $\delta(y, b) = y$; Se o autômato estiver no estado y e acontecer b continue em y.
- $\delta(z, b) = z$; Se o autômato estiver no estado z e acontecer b continue em z.

- $\delta(z, a) = \delta(z, g) = y$; Se o autômato estiver no estado z e acontecer a ou g faça uma transição para y.

É importante ressaltar alguns pontos sobre esse exemplo:

1. Um evento pode ocorrer sem mudança de estado, $\delta(x, a) = x$.
2. Dois eventos distintos podem ocorrer em um estado levando a mesma transição, $\delta(z, a) = \delta(z, g) = y$.
3. A função δ é uma função parcial sobre seu domínio $Q \times \Sigma$, isto é, não é necessário uma transição ser definida para cada evento em Σ para cada estado de Q , por exemplo, $\delta(x, b)$ e $\delta(y, g)$ não são definidas.

Importante também destacar a ideia de estado marcado, que são estados em que existe uma conclusão de tarefas. O estado inicial q_0 e um subconjunto de Q, Q_m representar esses estados marcados, no caso do nosso exemplo, os estados x e z representam esses estados marcados.

Em todo estado q_0 e Q_m é preciso definir uma função $\Gamma : Q \to 2^{\Sigma}$, que seria uma função de eventos ativos, $\Gamma(q)$ é o conjunto de todos os eventos e para quais $\delta(q, e)$ é definida, é chamada de conjuntos de eventos ativos.

Uma palavra de comprimento nulo, ou seja, uma sequência de eventos vazia é usualmente designada por ε. O conjunto Σ^* representa a linguagem composta por todas as possíveis sequências de eventos de comprimento finito formadas por eventos pertencentes ao alfabeto Σ e a palavra de comprimento nulo. Todo subconjunto de Σ^* constitui uma linguagem sobre Σ, em particular a linguagem vazia (representada por $\emptyset$), ε e Σ^* são linguagens em Σ.

2.4.3 Operações sobre autômatos

De modo a estudar e modelar um sistema de eventos discretos é importante que se conheça algumas operações que se podem ser feitas sobre os autômatos, que permitem, por exemplo, combinar ou compor dois ou mais autômatos. Nessa seção vamos introduzir os conceitos básicos de forma resumida algumas das operações mais comuns.

2.4.3.1 Acessibilidade

Um estado é dito acessivel, segundo Cassandras e Lafortune (CASSANDRAS; LAFORTUNE, 2008) se este pode ser alcançado através de qualquer cadeia s $\in L(G)$ tendo como origem o estado $q_0 \in Q$. Das definições de $L(G)$ e $Lm(G)$, observa-se

que é possível apagar de G todos os estados que não são acessíveis ou alcançáveis a partir de q_0 através de alguma palavra de $L(G)$, sem afetar as linguagens geradas e marcadas por G. Quando se fala em autômato, pode-se dizer que ele é acessível quando todos os estados podem ser alcançados a partida de q_0.

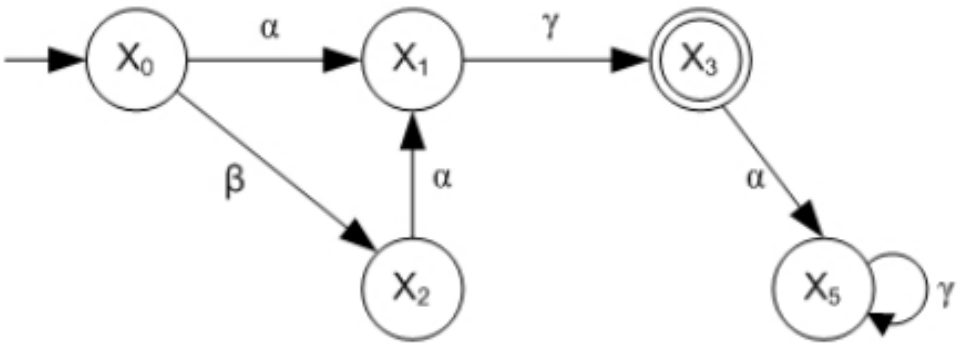

Figura 7 – Autômato acessível. Fonte: (SILVA, 2007)

2.4.3.2 Co-acessibilidade

Um autômato é dito co-acessivel quando todas as cadeias são prefixo de cadeias marcadas. $L(G) = Lm(G)$, ou seja, a partir de qualquer estado pode-se chegar a um estado final (marcado). É possível obter uma co-acessivel, eliminando os estados alcançados por cadeias que não podem ser completadas em tarefas.

2.4.3.3 Operação trim

Um autômato é dito Trim quando ele é acessível e co-acessivel ao mesmo tempo. Ele representa a ausência de bloqueios no sistema, isto é, a partir do estado inicial $q0$ ou de qualquer outro estado, sempre existirá um caminho que conduz a um estado marcado.

2.4.3.4 Composição Paralela

Cassandras e Lafortune (CASSANDRAS; LAFORTUNE, 2008) mostram que a composição paralela é uma operação que representa o comportamento sincronizado entre dois autômatos. Nesta operação um evento comum aos dois alfabetos de eventos, é permitido ocorrer se e somente se ele ocorre em ambos os autômatos simultaneamente.

No caso de um evento pertencer a somente um dos alfabetos, ele sempre estará habilitado a ocorrer no autômato resultante quando se realizar a composição. Desta

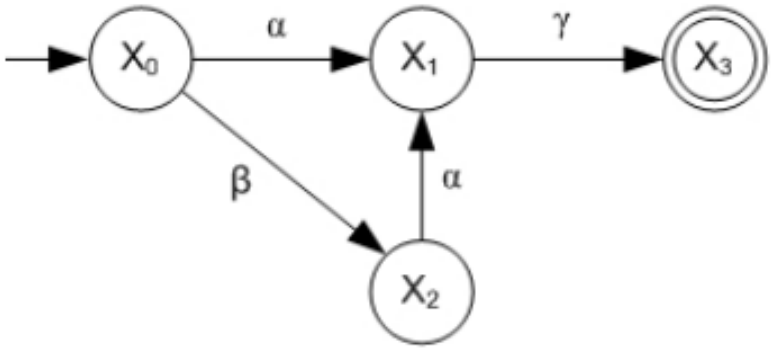

Figura 8 – Autômato Trim. Fonte: (SILVA, 2007)

forma o sincronismo se observa somente quando analisamos eventos comuns a ambos os alfabetos.

Um estado do autômato composto será marcado se ele resultar de estados marcados em ambos os geradores. Considere esse exemplo retirado de (SILVA, 2007), para um melhor entendimento sobre os conceitos de composição:

A composição paralela de dois autômatos G_1 e G_2, e formalmente definida como: $G_1||G_2 := Ac(Q_1 \times Q_2, \sigma_1 \cup \ \sigma_2, \delta, (q_0 1, q_0 2), Q_m 1 \times Q_m 2)$, Onde:

$$f((x_1, x_2), e) := \begin{cases} (f_1(x_1, e), f_2(x_2, e)), & \text{se } e \in \Gamma_1(x_1) \cap \Gamma_2(x_2) \\ (f_1(x_1, e), x_2), & \text{se } e \in \Gamma_1(x_1) \backslash \Sigma_2 \\ (x_1, f_2(x_2, e)), & \text{se } e \in \Gamma_2(x_2) \backslash \Sigma_1 \\ \text{indefinida para qualquer outro caso} & \end{cases} \tag{2.7}$$

Sejam os autômatos G_1 e G_2, onde estado inicial zero significando o estado de repouso e o estado um, o estado representativo do sistema em operação. Os eventos $(\alpha_1 e \alpha_2)$ representam início de operação, e os eventos (β_1, β_2) significam final de operação.

Figura 9 – Autômatos G_1 e G_2. Fonte: (SILVA, 2007)

A composição paralela desses autômatos, resulta no seguinte autômato:

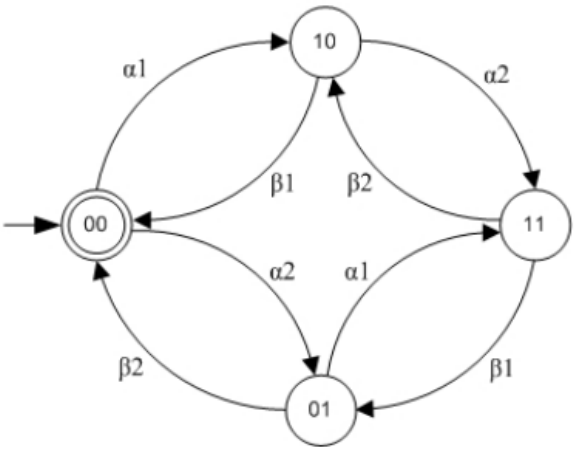

Figura 10 – Autômato composto de G_1 e G_2. Fonte: (SILVA, 2007)

Observa-se na figura 10 que em função dos autômatos G_1 e G_2 serem assíncronos, ou seja, não possuem eventos em comum, a evolução do autômato resultante determina que todos os eventos definidos em ambos os autômatos também estarão definidos em $(Q_1 \times Q_2)$. Por exemplo, o estado (01) representa a sincronização do estado 0(zero) de G_1, com o estado 1(um) de G_2. Isto significa que os eventos α_1 e β_2 deverão estar definidos.

Agora vamos mostrar, através de outro exemplo retirado de (LIMA, 2008), como seria essa composição para autômatos síncronos onde um evento pertencente a ambos os autômatos somente poderá ser executado se os dois autômatos os executarem simultaneamente. Ao realizar essa operação dois autômatos são sincronizados em seus eventos em comuns, os outros eventos poderão ser executados sempre que possível sem restrições. Considere os dois autômatos a seguir:

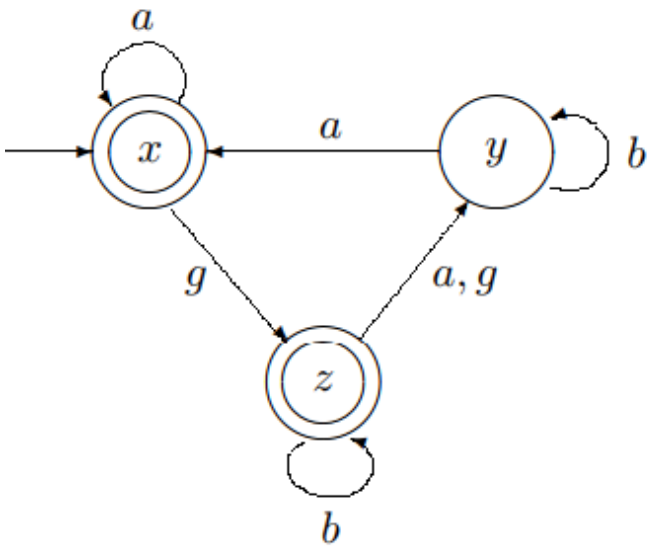

Figura 11 – Autômato G_1. Fonte: (LIMA, 2008)

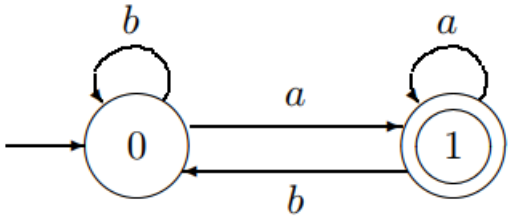

Figura 12 – Autômato G_2. Fonte: (LIMA, 2008)

A composição paralela desses dois autômatos resulta no autômato da figura 13. O conjunto de eventos comuns é a,b, e G_1 é o único que apresenta eventos particulares, nesse caso o evento g. Os estados da composição são formados por pares, do autômato G_1 e do autômato G_2.

$G_1||G_2$ são formados por pares. No estado inicial (x,0), o evento em comum a é o único evento possível de ocorrer e leva o sistema para (x,1), que por sua vez é um estado marcado, pois x é marcado em G_1 e 1 é marcado em G_2. Em contraste com $G_1 \times G_2$, outra transição pode ocorrer em (x,0): G_1 pode executar o evento g, sem a participação de G_2 e levar o autômato composto para o novo estado (z,0). Apos essa ocorrência, G_1 está no estado z e G_2 permanece no estado 0. O processo é repetido, encontrando-se todas as possíveis transições em (x,1) e (z,0), e em todos os novos estados gerados.

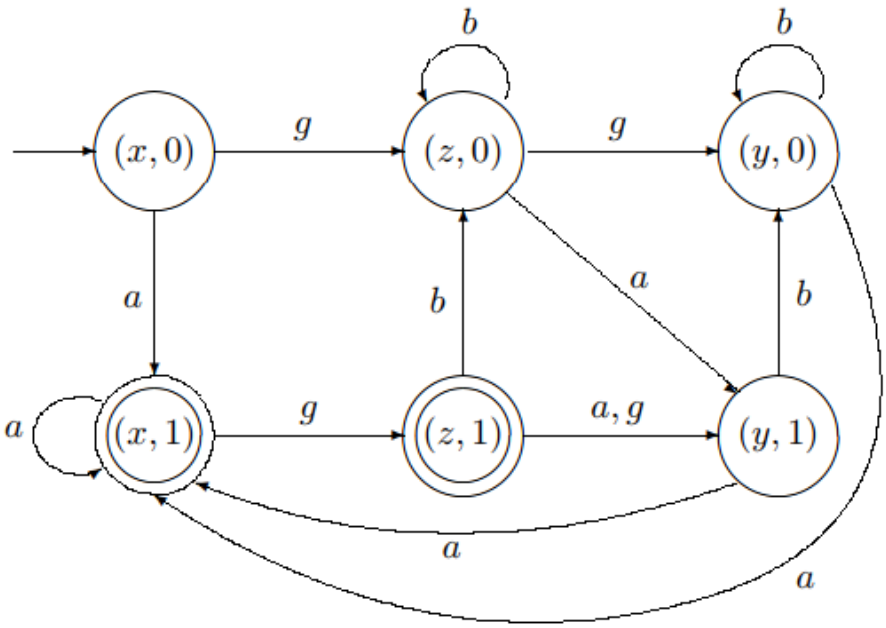

Figura 13 – Autômato resultante da composição paralela entra G_1 e G_2. Fonte: (LIMA, 2008)

2.4.4 Composição de autômatos para modelagem de SEDs

Uma característica importante da modelagem de SED é que ele pode ser visto como uma junção de todos os subsistemas que representam a totalidade do sistema, ou seja, o sistema como um todo pode ser visto como uma composição de todos os subsistemas que o compõe.

Quando se pensa na modelagem de um SED utilizando-se dos conceitos de autômatos e linguagens formais, pode se abordar a modelagem desse sistema de duas formas: Uma abordagem local e uma abordagem global e uma abordagem local como mostra (CURY, 2001). Na abordagem global o sistema é analisado como um todo e procura-se por um ADEF que represente todas as sequências possíveis.

Quando se ler esse tipo de abordagem já se pode perceber que não seria o jeito ideal de ser realizar a modelagem de um SED. Isso é facilmente constatado quando se pensa em um modelo de algum sistema grande, objetivamente quando se pensa em realizar alguma mudança na lógica de controle desse sistema, ou implantação de algum novo dispositivo teria que reestruturar todo o modelo, modificar toda a logica de controle para que isso seja possível.

Tendo isso em vista e a ideia de que, como já foi citado, um SED pode ser visto como a composição de subsistemas sobre a gerência de um controlador aplicando as devidas restrições de controle previamente estabelecidas, temos a conceito de uma abordagem local. A abordagem local de modelagem parte do princípio de que se pode

construir modelos locais para cada subsistema e impor restrição de coordenação, e que se pode compor os mesmos para obter um modelo do sistema global.

Ao contrário de uma abordagem global, essa abordagem local sugere uma maior facilidade para alterar modelos de sistema de grande porte, pois basta quando for necessária alguma alteração, modificar somente o modelo de subsistema correspondente e não o modelo como um todo. Com relação mais especificamente a modelagem realizada nesse trabalho foi utilizada a abordagem local.

Foram desenvolvidos autômatos separados para os dispositivos, cada um com suas respectivas regras, no nosso foi modelado dois subsistema, uma são os sensores de capturar de dados atmosféricos e outro a estação de coleta, cada subsistema desses terá seu comportamento independente representado através de um autômato esses modelos vão ser mais bem explicados no próximo capítulo, ficando a cargo do controlador gerenciar todos esses subsistemas como um único sistema.

2.4.5 Linguagem Gerada e linguagem marcada

Para a melhor compreensão desse trabalho é importante que se destaque dois conceitos relacionados à modelagem de um SED. Tem-se a ideia de linguagem gerada e linguagem marcada.

O conjunto de todas as sequencias de eventos possível de serem executadas a partir do estado inicial forma a linguagem gerada por um autômato, $L(G)$. Sendo formalmente definida de acordo com Ramadge e Wonham (Ramadge, P.J.G.; Wonham, W.M., 1989) como:

$$L(G) := s \in \Sigma^* : \delta(q_0, s) definida \tag{2.8}$$

O conjunto de sequencia pertencente a linguagem gerada que leva o sistema a um estado marcado constitui a linguagem marcada por um autômato, $L_m(G)$. Formalmente definida de acordo com Ramadge e Wonham (Ramadge, P.J.G.; Wonham, W.M., 1989) como:

$$L_m(G) := s \in L(G) : \delta(q_0, s) \in Q_m \tag{2.9}$$

Estas duas linguagens são executadas sobre o alfabeto Σ e fazem parte do conjunto Σ^* . Diz-se que um autômato é não - bloqueante quando, $L_m(G) = L(G)$

Para se entender melhor essa definição considere o seguinte exemplo: Suponha $E = \{a, b\}$ como um conjunto de eventos.Considere a linguagem: $L = \{a, aa, ba, aaa, aba, baa, bba, ...\}$ consistindo de todas as sequencias formadas pelos eventos a ou b sempre terminados pelo evento a. Essa linguagem é marcada pelo autômato

de estado finitos: $G = (X, E, f, \Lambda, X_0, X_m)$, representado pelo diagrama de transição da figura 14, em que $X = \{0, 1\}, X_0 = 0, X_m = \{1\}$ e f é definida como se segue: $f(0, a) = 1, f(0, b) = 0, f(1, a) = 1, f(1, b) = 0$.

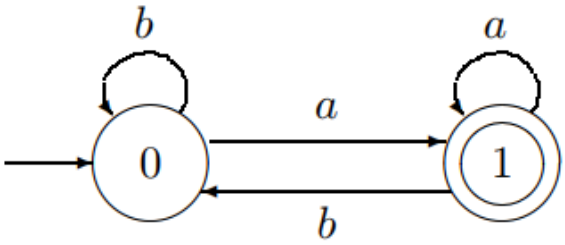

Figura 14 – Trajetória tipica de um sistema a eventos discretos. Fonte: (CURY, 2001)

Assim, estando no estado inicial 0, a única maneira de se alcançar o estado marcado 1 e pela ocorrência do evento a, em algum momento. Alcançando esse estado só ira retornar ao estado 0 apos a ocorrência do evento b. Pode-se concluir que $L_m(G) = L$. Note que f e uma função total em seu domínio, e portanto, a linguem gerada por G é $L(G) = E^*$.

2.5 Teoria do controle supervisório

A teoria do controle supervisório, das abordagens citadas para o controle de sistemas de eventos discretos, foi à escolhida para o desenvolvido desse trabalho. Essa teoria nos permite baseado na modelagem do sistema e no conjunto de restrições, regras estabelecidas para o funcionamento correto do sistema, realizar a síntese de um controlador, que vai ter a visão geral do sistema e controla-lo de modo que todas as regras sejam obedecidas e o sistema funcione da melhor maneira possível.

Durante o capítulo foi apresentado à ideia que podemos representar por uso de linguagens formais e autômatos todas as possíveis sequencias de eventos que pode ocorrer no sistema. Porem algumas dessas sequencia de eventos podem descrever um comportamento inadequado, indesejado do sistema.

Segundo (Ramadge, P.J.G.; Wonham, W.M., 1989) a teoria do controle supervisório propõe que uma estrutura denominada de controlador observe a sequência de eventos gerados no sistema a ser controlado, denominado de planta e que, logo após, instantaneamente, determine o subconjunto de eventos concatenados a essa sequência preservam o sistema dentro do comportamento desejado.

Este subconjunto de eventos define uma entrada de controle e especifica todos os eventos que estão habilitados a ocorrer e os que não estão habilitados ficam desa-

bilitados. Após um novo evento a entrada de controle deve ser atualizada e um novo subconjunto de eventos deve ser habilitado.

Objetivamente falando a teoria do controle supervisório busca coordenar todos os subsistemas que compõem o modelo de modo a diminuir o tempo de execução das operações globais, garantir a ausência de bloqueios e não violar as especificações de segurança (QUEIROZ, 2002).

Como mostra (BRANDIN B.A; CHARBONNIER, 1994) existem duas estratégias principais para o propósito de um sistema de controle supervisório:

1. Supervisão centralizada, onde toda a tarefa do sistema supervisório é executada por um único supervisor que incorpora o comportamento em malha fechada da planta. Além disto, tal supervisor é provido de informações completas sobre a ocorrência de eventos na planta.

2. Supervisão Modular, a qual implica na divisão de toda a tarefa do sistema supervisório em dois ou mais supervisores que possuem uma visão parcial da planta, atuando concorrentemente.

2.5.1 Supervisão centralizada ou monolítico

Esse trabalho abordar a ideia de uma supervisão centralizada, ou seja, objetivo do controle supervisório monolítico é projetar um único controlador cuja função é habilitar ou desabilitar eventos controláveis, conforme a sequência de eventos observados na planta, de forma que o sistema em malha fechada obedeça a algumas regras operacionais especificadas (QUEIROZ, 2002).

Com a finalidade de molear um SED para aplicação dos conceitos da teoria do controle supervisório Ramadge e Wonham (Ramadge, P.J.G.; Wonham, W.M., 1989) particionam o sistema em dois conjuntos de eventos: eventos controláveis (Σ_c) e eventos não-controláveis (Σ_u), de modo que $\Sigma = \Sigma_c \cup \Sigma_u$. Os eventos Σ_c podem ser desabilitados em qualquer momento pelo controlador, ao passo que os eventos Σ_u não são afetados pela ação do controlador.

Um subconjunto de eventos γ que estão aptos a ocorrer é defino através da determinação das entradas de controle. Uma entrada de controle para um gerado G consiste do subconjunto $\gamma \subseteq \Sigma$ que satisfaça $\Sigma_u \subseteq \gamma$, de acordo com essa condição, qualquer evento não controlável esta permito ocorrer, o conjunto de entradas de controle e denotado por $\Gamma \subseteq 2^{\Sigma}$ (Ramadge, P.J.G.; Wonham, W.M., 1989).

Um SED quando esta sobre ação de um supervisor f obedece a restrições adicionais. Formalmente um supervisor f corresponde a seguinte função: $f : L(G) \rightarrow \Gamma$

que associa a cadeia $s \in L(G)$ gerada pela planta, a uma entrada de controle $\gamma = f(s)$ que restringe seu comportamento.

Após a geração de uma palavra w pela planta, o próximo evento deve ser um elemento de $f(w) \cap \Sigma(\delta(w, q_0))$. Define-se $\Sigma(q)$ como conjunto de eventos factíveis no estado q com, $\Sigma(q) \subset \Sigma$, de modo que para cada evento $\sigma \in \Sigma(q), \delta(\sigma, q)$ é definida. Portanto o evento a ser gerado após a palavra w deverá ter simultaneamente a sua entrada de controle habilitada pelo supervisor e a função de transição definida na planta satisfazendo $f(w) \cap \Sigma(\delta(w, q_0))$ (Ramadge, P.J.G.; Wonham, W.M., 1989).

A linguagem marcada $L_m(f/G)$ e que representa o comportamento marcado do sistema sob supervisão é definida como: $L_m(f/G) = L_m(G) \cap L(f/G)$. Ou seja, ela representa o conjunto de tarefas completas realizadas pela planta e que sobrevivem à ação de controle do supervisor.

O supervisor ou controlador que afeta o comportamento da planta ao habilitar e desabilitar eventos controláveis pode ser descrito, ou representado na prática, por um autômato $T : T = (X, \Sigma, \delta, X_m, x_0)$, onde: Onde X é o conjunto de estados, Σ é o alfabeto utilizado por G, δ a função de transição, X_m é o conjunto de estados marcados e x_0 é o estado inicial. (Ramadge, P.J.G.; Wonham, W.M., 1989).

De um modo prático, o supervisor f será representado graficamente pelo autômato T, onde as ações de controle de f sobre a planta G estarão definidas na estrutura de transição de T. Os eventos não habilitados pelo supervisor f não aparecerão na estrutura de transição de estados do autômato T, assim, na operação de composição síncrona, como os alfabetos são comuns, um evento só poderá ocorrer, se ele ocorrer simultaneamente em ambos os autômatos de T e G.

Desta forma, dado o autômato T que representa f, se uma dada cadeia $s \in L(f/G)$, então, $s \in L(T)$ e $s_\sigma \in L(T)$ somente se $\sigma \in f(s)$. Ainda assim, se $s \in L(f/G)$, $s_\sigma \in L(G)$, e $\sigma \in f(s)$, então $s_\sigma \in L(T)$. A primeira afirmação assegura que as transições desabilitadas por f , não aparecem na estrutura de transição de T , ao passo que a segunda, assegura que as transições habilitadas por f, e que são factíveis em G, aparecerão na estrutura de transição de T.

A figura 15 ilustra o funcionamento de um supervisor monolítico, observa-se que a planta gera os eventos controláveis e não-controláveis, cabendo ao supervisor a ação de desabilitar somente os eventos controláveis (SILVA, 2007).

Assume-se que os autômatos T (Supervisor) e G (planta), são executados de forma concorrente, ou seja, estando os autômatos em um determinado estado $(x, q) \rightarrow (x', q')$, a ocorrência de um evento σ determina que o autômato T evolui de $x \rightarrow x'$, e o autômato G de $q \rightarrow q'$. (SILVA, 2007)

Figura 15 – Arquitetura do controle monolítico. Fonte: (Ramadge, P.J.G.; Wonham, W.M., 1989)

3 Modelagem do sistema de eventos discretos

3.1 Introdução

Nesse capítulo pretende-se mostrar de fato como foi feita a modelagem do sistema de eventos discretos e a aplicação das técnicas de controle da teoria do controle supervisório. Além disso, pretende-se da uma breve explicação sobre o que é o sistema a ser modelado e do que se tratam as estações de coleta de dados.

Sistemas de eventos discretos são caraterizados por sua natureza instantânea, o que lhes confere um caráter discreto no tempo, como por exemplo, uma mudança de estado em um sensor. A natureza discreta dos SEDs faz com que modelos matemáticos convencionais, baseados em equações diferenciais, não sejam adequados para tratá-los (Ramadge, P.J.G.; Wonham, W.M., 1989).

Por outro lado, a ampla gama de aplicações discretas faz com que seja altamente desejável encontrar soluções para problemas relacionados ao seu controle. Uma das Soluções para o controle de SEDs é por meio da teoria do controle supervisório abordada neste TCC.

Será mostrado a aplicação da teoria do controle supervisório em um problema real, nesse caso, para o controle de estações de coleta de dados atmosférico, essas estações e seus sensores serão modelados como um SED e controlados a partir da teoria do controle supervisório.

3.2 Definição e características de estações de coletas dados meteorológicos

Para se alcançar o objetivo proposto no trabalho se pensou em um cenário onde o sistema de eventos discretos a ser modelado seria uma estação de coleta de dados atmosféricos.

Porém, primeiramente é necessário se entender o que é uma estação de coleta de dados atmosférico, como ela funciona e para que serve. Segundo o instituto nacional de meteorologia (INMET):

“Uma estação meteorológica automática (EMA) coleta, de minuto em minuto, as informações meteorológicas (temperatura, umidade, pressão atmosférica, precipitação, direção e velocidade dos ventos, radiação solar) representativas da área em que

está localizada. A cada hora, estes dados são integralizados e disponibilizados para serem transmitidos, via satélite ou telefonia celular, para a sede do INMET, em Brasília. O conjunto dos dados recebidos é validado, através de um controle de qualidade e armazenado em um banco de dado para a elaboração de previsão do tempo e dos produtos meteorológicos diversos de interesse de usuários setoriais e do público em geral e para uma vasta gama de aplicações em pesquisa em meteorologia, hidrologia e oceanografia." (INMET, 2011)

Ainda segundo o INMET uma estação meteorológica é composta pelos seguintes elementos:

1. Subsistema de coleta de dados;
2. Subsistema de controle e armazenamento;
3. Subsistema de energia (painel solar e bateria);
4. Subsistema de comunicação.

O subsistema de coleta de dados é quem de fato realiza a coleta dos dados atmosféricos. Essa coleta é feita através de sensores para medição dos parâmetros meteorológicos a serem observados. As medidas são tomadas, em intervalos de minuto a minuto, e integralizadas no período de uma hora, para poder ser transmitidas.

O subsistema de armazenamento é composto por um processador central de baixo consumo de energia (*datalogger*),que faz o registro dos valores observados em uma unidade de memória que contém as instruções programadas para aquela unidade.

Os dados são armazenados em uma memória não volátil que mantém os dados medidos por um período especificado. O sub-sistema de energia torna a estação independente de energia elétrica externa e não requer nenhum equipamento ou sala para sua operação diária.

Por fim o subsistema de comunicação faz a transmissão dos dados coletados que estão armazenados na memória, são esses os principais elementos que compõem uma estação meteorológica.

A figura 16 mostra uma estação meteorológica automática para a coleta de dados atmosféricos.

Figura 16 – típica Estação Meteorológica Automática composta por sensores, mastro com caixa data-logger, painel solar, pára-raios, cercado. Fonte: (INMET, 2011)

O sistema de evento discreto modelado nesse trabalho representa estações de coleta de dados atmosféricos. As estações aqui modeladas serão bem mais simples que a apresentada na descrição do INMET, contando apenas com o que seria o subsistema de coleta de dados, os outros elementos foram retirado do modelo.

Portanto, nosso sistema possuirá duas estações de coleta de dados, onde cada estação é composta por dois sensores responsáveis pela coleta de dados, um sensor responsável pela captura dos dados relacionados à temperatura e um sensor responsável por capturar os dados relacionados à umidade.

Outro elemento importante desse sistema é o usuário, todo sistema funciona com base no que o usuário determina, ou seja, o sistema vai tentar atender a solicitação do usuário da melhor maneira possível.

Para que esse requisito seja atendo existe no modelo a figura do controlador, que tem como objetivo fazer com que o sistema atue de forma coordenada e da maneira esperada, ele observa os eventos ocorridos e escolhe que eventos, dentre os fisicamente possíveis de ocorrer no estado atual, que são permitidos que ocorram a seguir.

Para isso o sistema precisa ser controlado de alguma forma de modo a conseguir esse ”comportamento desejado”. Assim, a entrada do sistema é muitas vezes

visto como um sinal de controle destinado a obter esse comportamento desejado.

Diferente da descrição do INMET onde os sensores capturam os dados em intervalos de minuto a minuto, no sistema a ser modelado o usuário é que determina qual o intervalo em que ele quer que essa captura seja realizada, dentre as três opções existentes, captura com frequência alta, média e baixa, sendo a alta a opção onde a captura de dados ocorrerá com o menor intervalo de tempo.

Um fato importante sobre o sistema modelado que vale ser destacado é que foi preciso discretizar a frequência associada aos sensores. Normalmente uma frequência é vista como algo aumenta gradativamente aqui foi necessário realizar essa discretização da frequência fazendo com que ela passe abruptamente de uma frequência baixa para uma frequência média ou alta, por exemplo. Isso foi necessário para que o sistema se encaixe com as características de um SED.

O controlador vai garantir que o usuário consiga medir o dado solicitado com a frequência desejada não importando qual a estação ou sensor vai precisar usar.

Por exemplo, considere o cenário básico onde o usuário faz uma requisição para a captura a temperatura numa frequência alta. Resumindo o que ocorre no sistema, nesse cenário específico, o usuário faz uma requisição ao sistema, nesse caso, ele solicita dados referentes à temperatura numa frequência alta, a partir dessa solicitação o controlador faz uma verificação para decidir se é possível entregar esses dados ao usuário. Uma das verificações possíveis é, por exemplo, ver se as estações não estão em falha, o que impossibilitaria essa estação especifica de entregar os dados e o controlador teria que solicitar esses dados de outra estação. Esse passo a passo de funcionamento do modelo vai ser mais bem explicado no capitulo quatro.

Para que o sistema funciona corretamente é necessario que ele seja controlado, para isso existe a figura do controlador que é quem vai ter a visão geral do sistema, ele vai ser responsável por atender todas as regras previamente estabelecidas no modelo e fazer com que o sistema funcione corretamente sem nenhum conflito ou cenários indesejados. Essas regras do modelo vão ser detalhadas mais a frente.

Tendo em vista esse exemplo, se consegue identificar o que seria uma sequência de passos para o funcionamento geral do sistema, nesse caso específico.

1. O usuário informa ao sistema qual o tipo de dado que ele quer coletar e em qual frequência
2. O sistema verifica se é possível atender a requisição em alguma das estações de coleta
3. Se a estação ou sensor escolhido primeiro para a realização da coleta estiver com algum problema o sistema verificará se é possível atender a requisição do

usuário em outra estação

4. Caso seja possível atender o pedido do usuário, o sistema fará a captura e enviará os dados

5. Caso não seja possível o sistema retornará que não foi possível realizar a coleta de dados

3.3 Composição do sistema a ser modelado

O sistema a ser modelado apresenta duas estações de coleta de dados. Cada estação conta com dois sensores para a realização dessa coleta, um sensor para a captura de dados relativos à temperatura e um sensor para a captura de dados relativos à umidade.

A estação de captura de dados pode se apresentar nos estados ligado, desligado ou em um estado de falha. Durante a modelagem, pensou-se em alguns cenários que levariam a eventuais falhas do sistema. Dentre os cenários possíveis estão:

1. Uma falha em algum dos sensores de uma estação, o sensor pode falhar por algum eventual problema de hardware ou energia.

2. Uma falha da própria estação, a estação pode falhar por exemplo por uma falta de energia.

3. Um falha tanto da estação quanto dos sensores.

Para cada um desses cenários de falha o sistema se comporta de uma maneira especifica a fim de corrigir o problema e entregar o que o usuário solicitou.

Como o sistema é composto por mais de uma estação, tem-se algumas estratégias para lidar com essas falhas, caso tudo esteja funcionando perfeitamente o sistema tem a liberdade pra escolher de qual estação ele vai usar os sensores para realizar a captura dos dados. Caso uma ou outra estação falhe o controlador tem total liberdade de migrar entre as estações pra realizar a coleta desses dados, o usuário não tem a opção de escolher de qual estação essa captura vai ser realizada, isso fica a cargo do controlador.

Têm-se então alguns cenários de falha e espera-se que o sistema funcione de uma maneira que seja possivel entregar os dados ao usuário, esses cenários e o funcionamento geral do sistema vão ser mais bem abordados no próximo capítulo.

3.4 Modelagem da estação de captura de dados atmosféricos

A estação de coleta de dados modelada nesse trabalho é composta por dois sensores um que capturar dados referentes à temperatura e outra para capturar dados referentes à umidade. Tanto os sensores como as estações são modeladas como autômatos que tomam decisões baseadas em regras previamente estabelecidas e mudam de estado de acordo com eventos que são disparado, eventos, como por exemplo, uma requisição de usuário para capturar algum dado em determinada frequência, que é um evento controlável e vai fazer o sistema mude seus estados a fim de atender essa solicitação.

A modelagem dos dispositivos é feita por meio de estados de funcionamento. Por exemplo, a estação de coleta possui o estado ligado, desligado ou em falha, sendo assim é possível pensar em cada estação de coleta do sistema como sendo representada por um autômato de estados finitos, tanto a estação quanto os sensores são vistos como subsistemas que após a composição formam o sistema de uma maneira geral cabendo ao controlador a gerência desse sistema.

É importante notar aqui que o controlador não enxerga esses subsistemas separados, ele só passa a controlar e aplicar as técnicas de controle supervisório após a composição desses subsistemas em um único sistema

Para o modelo em estudo, foram desenvolvidos três autômatos: um para representar as estações de coleta, outro para representar sensores e por fim a representação do estado de falha por um autômato a parte.

3.4.1 Autômato da estação de coleta de dados

O autômato da estação possui dois estados L (ligado) e D (desligado), além disso, o autômato retorna uma variável de estado booleana e (de estação), que retorna *true* para o caso do autômato se encontrar no estado ligado e *false* para o estado desligado, de acordo com esse retorno o controlador consegue identificar o status da estação, conforme mostra a figura 17.

O autômato alterna entre seus estados de acordo com o evento controlável recebido, nesse caso CE (controle da estação), esse evento vai definir se a estação permanece ligada ou muda de estado para desligado.

3.4.2 Autômato dos sensores e das frêquencias

A seguir temos o autômato relacionado aos sensores e suas frequências, o mesmo possui quatro estados um para quando o sensor não estiver sendo usado, o

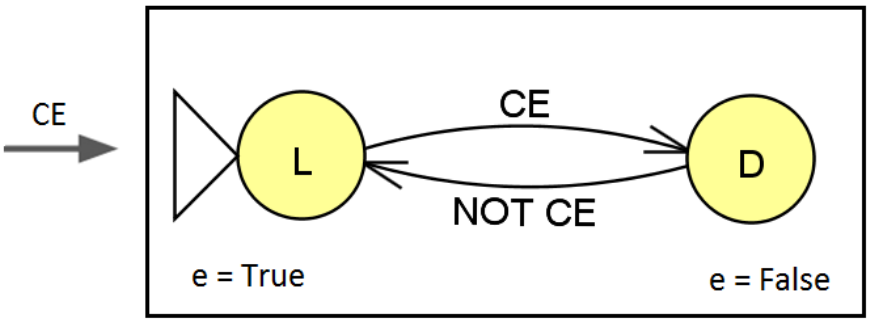

Figura 17 – Autômato relacionado a estação de coleta de dados atmosféricos

estado NC (não conectado) e outros três para cada uma das frequências de captura: $L1$ (captura lenta), $L2$ (captura média) e $L3$ (captura alta).

Sendo possível alternar entre os estados de acordo com o evento que for disparado, conforme mostra a figura 18. Os possíveis eventos de ocorrer nesse autômato são os seguintes:

1. Os eventos controláveis $U1$ e $D1$ caso ocorram, significa que o autômato vai saltar um nível ou regredir um nível no autômato, levando em consideração seu estado atual.
2. Os eventos controláveis $U2$ e $D2$ caso ocorram, significa que o autômato vai saltar dois níveis ou regredir dois níveis no autômato, levando em consideração seu estado atual.
3. Os eventos controláveis $U3$ e $D3$ caso ocorram, significa que o autômato vai saltar três níveis ou regredir três níveis no autômato, levando em consideração seu estado atual.

Além disso, é importante notar que associado a cada um desses estados do autômato existe algumas variáveis de estados como mostra a figura 18 essas variáveis indicam ao controlador o status do autômato, ou seja, em que frequência ele se encontra operando naquele momento e qual o estado atual do sensor ligado ou desligado. Por exemplo, no estado $L1$ têm-se as seguintes variáveis de estado e seu significado:

1. Variável de estado relacionado ao status do sensor, S. Que indica ao controlador se o sensor esta ligado $S = true$ ou desligado $S = false$.
2. Variáveis de estado relacionado com a frequência do sensor, $l1$, $l2$, $l3$. Essas variáveis indicam ao controlador se o sensor se encontra no nível $L1$, $L2$ ou $L3$, ou seja, se o autômato se encontra operando na frequência baixa, média ou alta respectivamente.

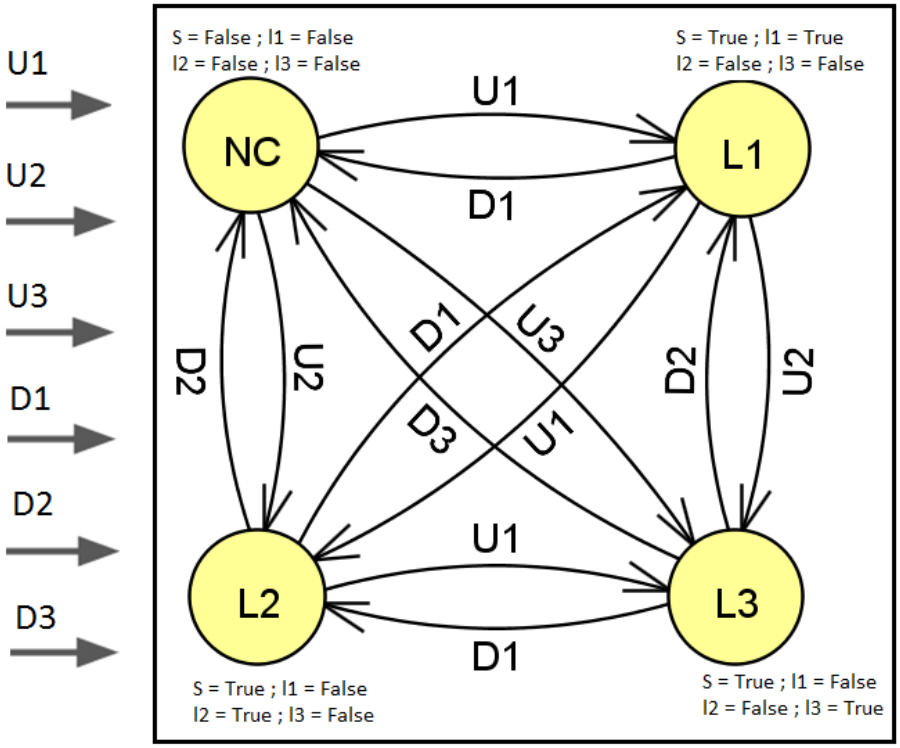

Figura 18 – Autômato relacionado ao sensor de captura de dados atmosféricos

3.4.3 Autômato de falha do sensor e da estação

O terceiro autômato modelado é o autômato relacionado às falhas. Deve-se destacar duas estratégias diferentes para adicionar o aspecto das falhas a um modelo: o modo direto e o modo por composição.

Na estratégia direta deve-se modelar diretamente o estado de falha no autômato desejado. Em nosso caso, isto seria feito pela simples adição de um novo estado aos autômatos do sensor e da estação e pela adição das transições correspondentes. As vantagens desta estratégia são diminuir o número de estados existentes no modelo final após as composições e simplificar a geração do controlador, ganhando assim desempenho.

Por outro lado, esta estratégia além de tornar os autômatos mais complicados, pois seria preciso criar novas transições partindo de todos os estados já existentes no autômato para este novo estado, dificultando assim futuras mudanças, essa abordagem ainda da uma margem maior ao erro, pois essas transições de estado seriam todas feitas de forma manual.

Na estratégia por composição, o modelo de falhas é criado em separado e é posteriormente combinado ao modelo desejado por meio da operação de composição paralela, conceito abordado no capítulo dois. O autômato de falha possui dois estados W (funcionando) e F (falha), e alterna entre seus estados conforme a ocorrência dos

eventos não controláveis FC e RC, como mostra a figura 19. Além disso, possui uma variável de estado f para indicar ao controlador se o modelo está em falha (f = true) ou funcionando (f = false).

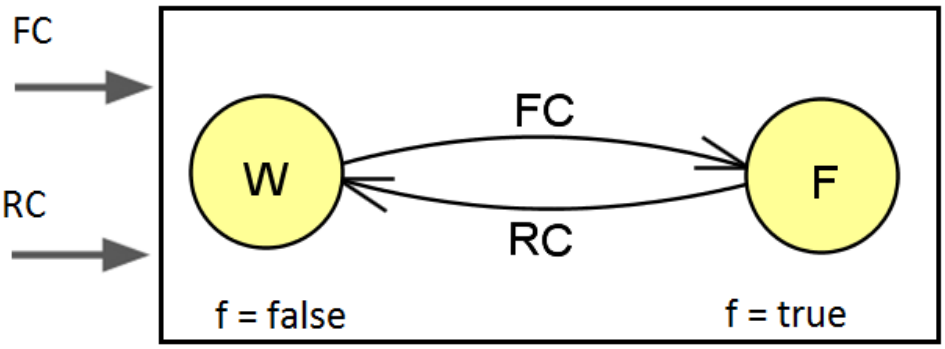

Figura 19 – Autômato relacionado ao estado de falha do sensor e da estação

3.5 Regras de controle do modelo

Além dos autômatos, a descrição do modelo é complementada pela adição de regras que deverão ser observadas pelo controlador.

São essas regras que irão definir o comportamento dos dispositivos e como eles vão interagir entre si. As regras são definidas a partir dos sinais que são passados para o sistema. São os seguintes sinas possíveis de ocorrer:

1. O usuário solicita medir temperatura numa frequência baixa.
2. O usuário solicita medir temperatura numa frequência média.
3. O usuário solicita medir temperatura numa frequência alta.
4. O usuário solicita medir umidade numa frequência baixa.
5. O usuário solicita medir umidade numa frequência média.
6. O usuário solicita medir umidade numa frequência alta.

De acordo com o sinal recebido e o cenário atual do sistema o controlador, baseado no conjunto de regras previamente escritas, vai fazer as transições de estados adequadas, habilitar ou desabilitar que eventos ocorram, a fim de atender a solicitação do usuário.

Nosso modelo é controlado por dezenove regras descritas abaixo em linguagem natural. Essas regras vão ser abordadas de uma maneira mais formal no próximo capítulo.

As regras do modelo são divididas da seguinte forma:

1. Regras relacionadas as estações.
   Essas regras são importantes, pois controlarão a estação tanto quando não se tem sensor em uso, desligado a estação e evitando gasto de energia desnecessário quanto para cenários de falhas em que a estação deve desligar quando os sensores não então funcionando. São elas:

   - Estação um deve estar desligada quando os sensores estão sem uso
   - Estação um deve estar ligada quando pelo menos um sensor esta em uso
   - Estação dois deve estar desligada quando os sensores estão sem uso
   - Estação dois deve estar ligada quando pelo menos um sensor esta em uso

2. Regras relacionadas ao uso dos sensores e frequências.
   Essas são as regras mais importantes do modelo, pois são as regras que vão gerencia qual sensor usar, de que estação e em qual frequência de acordo com a requisição do usuário e a situação atual do sistema, são as regras base do modelo. São as principais regras que controlador tem que fazer com que sejam cumpridas para o bom funcionamento do sistema. São elas:

   - Se usuários precisam de temperatura em algum nível, então um e somente um sensor de temperatura do sistema deve estar ligado.
   - Se usuários precisam de temperatura com captura intensa, então algum sensor de temperatura do sistema deve estar ligado e com captura intensa.
   - Se usuários precisam de temperatura com captura moderada, então algum sensor de temperatura do sistema deve estar ligado e com captura moderada.
   - Se usuários precisam de temperatura com captura baixa, então algum sensor de temperatura do sistema deve estar ligado e com captura baixa.
   - Se usuários não precisam de temperatura, então os sensores de temperatura das estações de coleta devem estar desligados.
   - Se usuários precisam de umidade em algum nível, então um e somente um sensor de umidade do sistema deve estar ligado.
   - Se usuários precisam de umidade com captura intensa, então algum sensor de umidade do sistema deve estar ligado e com captura intensa.
   - Se usuários precisam de umidade com captura moderada, então algum sensor de umidade do sistema deve estar ligado e com captura moderada.
   - Se usuários precisam de umidade com captura baixa, então algum sensor de umidade do sistema deve estar ligado e com captura baixa.

- Se usuários não precisam de umidade, então os sensores de umidade das estações de coleta devem estar desligados.

3. Regras de apoio para o controle de falhas.
   Essas são as regras que garantem os cenários de falhas. Com essa essas regras os sensores em falha não sejam usado e que as estações sejam desligadas caso todos seus sensores estejam em falha. São elas:

   - Se o sensor de temperatura da estação um estiver em falha, então desligue esse sensor.
   - Se o sensor de temperatura da estação dois estiver em falha, então desligue esse sensor.
   - Se o sensor de umidade da estação um estiver em falha, então desligue e esse sensor.
   - Se o sensor de umidade da estação dois estiver em falha, então desligue esse sensor.
   - Se estação um estiver em falha, então não desligue essa estação.
   - Se estação dois estiver em falha, então não desligue essa estação.

4 Análise da modelagem do sistema de eventos discretos

Nesse capítulo vamos apresentar uma descrição das partes mais importantes da modelagem e realizar testes para verificação das regras de controle.

4.1 Ferramentas utilizadas

A fim de se alcançar o objetivo proposto nesse trabalho foi necessário à utilização de algumas ferramentas, como o foco do trabalho não está nas ferramentas utilizadas, mas no resultado de seu uso, essa seção vai abordar o básico sobre essas ferramentas.

O ambiente de testes foi configurado em uma instância do Cloud9, que é um ambiente de desenvolvimento integrado on-line. Ele suporta várias linguagens de programação, incluindo C, C ++, PHP, Ruby, Perl, Python, JavaScript com Node.js e Go. Além de permitir aos desenvolvedores começar a codificar imediatamente com espaços de trabalho pré-configurados, ainda permite a colaboração compartilhada do código, testes de compatibilidade de visualização e visualização ao vivo.

Os autômatos foram feitos na linguagem BZR e compilados pelo Heptagon e a síntese do controlador foi feita pelo Sigali, levou um bom tempo para aprender o uso básico dessas ferramentas necessárias para realizar a modelagem, porque além da pouca documentação também não existe muito material sobre elas disponíveis na internet como em sites e fóruns, por exemplo.

De uma forma bastante resumida o Heptagon/BZR, segundo seu manual, é um idioma de fluxo de dados síncrono, com uma sintaxe que permite a expressão de estruturas de controle. Um programa típico do Heptagon levará como entrada uma sequência de valores e produzirá uma sequência de valores. Então, variáveis (entradas, saídas ou locais), bem como constantes são realmente fluxos variáveis ou constantes. Os operadores usuais (por exemplo, aritmética ou operadores booleanos) são aplicados de forma pontual nessas sequências de valores.

Considere o exemplo retirado do manual do Heptagon, o programa Heptagon abaixo é composto por um nó soma, executando a soma pontual de suas duas entradas inteiras:

```
[h] node plus(x:int;y:int) returns (z:int)
let
z = x + y;
tel
```

x e y são as entradas do nó e z é a saída x, y e z são do tipo int, z é definido pela equação $z = x + y$. Uma execução do nó soma pode então ser:

Tabela 1 – Resultado da soma do nodo plus

x	1	2	3	4
y	2	4	5	8
plus(x,y)	3	6	8	12

Já o sigali foi usado para fazer a síntese do controlador, também de acordo com seu manual, é uma ferramenta de verificação de modelos que manipula os Sistemas Dinâmicos Polinomiais (PDS) (que podem ser vistos como uma representação implícita de um autômato) como modelos intermediários para sistemas de eventos discretos. Oferece funcionalidades para verificação de sistemas reativos e síntese de controlador discreto.

As técnicas utilizadas consistem em manipular o sistema de equações em vez dos conjuntos de soluções, o que evita a enumeração do espaço de estados. Cada conjunto de estados é caracteristicamente caracterizado por um polinômio e as operações em conjuntos podem ser realizadas de forma equivalente nos polinômios associados. Portanto, um amplo espectro de propriedades, como vivacidade, invariância, acessibilidade e atratividade podem ser verificados. Muitos algoritmos para computar estados de predicados também estão disponíveis.

Para mais informações sobre o Heptagon e Sigali recomendo a leitura dos seus manuais, Heptagon (HEPTAGON/BZR…, 2017) e Sigali (SIGALI…, 2004).

4.2 Descrição dos autômatos do modelo e suas regras

No capítulo anterior foi mostrado cada autômatos que compõe o sistema e as regras de controle, aqui pretende-se mostrar, formalmente, como ficou os principais pontos da modelagem dos autômatos e das regras que compõem o modelo, em termos de código, bem como as mudanças que sofreram em cada versão implementada, no presente trabalho foram feitas três versões para a modelagem do sistema.

Inicialmente vamos mostrar a modelagem da estação de captura de dados meteorológicos, o modelo mostrando na figura 20 tem como objetivo representar uma estação de coleta de dados, a ideia desse modelo é representar uma estação onde dependendo do acontecimento ou não do evento controlável CE, o controlador altera

o estado da estação para desligado caso o evento CE ocorra e ligado caso CE não ocorra.

```
node station(ce : bool) returns (e:bool)
let
    automaton
      state Ligado do
        e = true
      unless ce then Desligado
      state Desligado do
        e = false;
      unless not ce then Ligado
    end
tel
```

Figura 20 – Modelo de uma estação de coleta de dados

O modelo da estação foi um dos poucos modelos que não teve mudanças em relação a sua versão final. Diante desse modelo podemos observar que o evento controlável CE é que vai determinar o estado em que o autômato vai estar, e dependendo do estado em que ele estiver a variável de estado e vai retornar *true* para informar que a estação esta no estado ligado e *false* para o estado desligado como mostra a figura 20.

O estado em que autômato da estação estiver vai influenciar os estados de outros autômatos, por exemplo, o estado da estação vai influenciar no estado dos sensores, caso a estação esteja desligada o controlador vai desabilitar o evento que usa os sensores dessa estação para captura dos dados requisitados, e habilitar o evento de captura na outra estação, tomando as medidas necessárias de acordo com esse necessário, sempre obedecendo às regras que foram previamente impostas a ele, para garantir o funcionamento desejado do sistema.

A seguir temos dois modelos que sofreram muitas mudanças entre sua primeira versão e sua ultima versão. Vamos mostrar agora como era e como ficou os autômatos do sensor e da frequência do sensor.

Inicialmente no modelo tínhamos dois autômatos separados como mostra as figuras 21 e 22. Na figura 21 temos o autômato que representa o subsistema do sensor do nosso modelo, que funciona da mesma forma que o autômato da estação, alternando entre ligado e desligado, de acordo com o acontecimento ou não de determinado evento controlável, representado por CT no modelo do autômato.

A ideia é a mesma da estação de coleta, a figura 21 mostra como ficou modelagem inicial do sensor. Temos uma variável de estado S para indicar se o sensor se encontra ligado ou desligado a depender do acontecimento ou não do evento controlável CT, caso o evento ocorra s retorna *true* para indicar que o sensor se econtra no

estado ligado e caso o evento não ocorra s recebe *false* para indiciar que o sensor está no estado desligado.

```
node sensor(ct : bool) returns (s:bool)
let
     automaton
      state Ligado do
        s = true;
      unless ct then Desligado
      state Desligado do
        s = false;
      unless not ct then Ligado
    end
tel
```

Figura 21 – Modelo do sensor de captura de dados

Já o autômato que foi modelado inicialmente para representa as frequências, figura 22, é um pouco mais complexo, pois possui mais estados e transições. O autômato possui três leveis, $l1$, $l2$, $l3$ que representam as frequências de captura baixa, média e alta consecutivamente. Como eventos controláveis ele recebe *ups*, $u1$, $u2$, $u3$ do controlador para avançar nessas frequências o seja ir para a frequência baixa, média ou alta e *donws*, $d1$, $d2$, $d3$ para regredir nessas mesmas frequências.

```
node frequency(u1, u2, u3, d1, d2, d3: bool) returns (l1,l2,l3:bool)
let
    automaton
      state NC do
        l1 = false; l2 = false; l3 = false;
      unless u1 then L1 |
             u2 then L2 |
             u3 then L3
      state L1 do
        l1 = true; l2 = false; l3 = false;
      unless u1 then L2 |
             u2 then L3 |
             d1 then NC
      state L2 do
        l1 = false; l2 = true; l3 = false;
      unless u1 then L3 |
             d1 then L1 |
             d2 then NC
      state L3 do
        l1 = false; l2 = false; l3 = true;
      unless d1 then L2 |
             d2 then L1 |
             d3 then NC
    end
tel
```

Figura 22 – Modelo da frequência relacionado ao sensor de captura

Antes de mostrar como ficou a modelagem final do sensor e de suas frequências, é importante falar um pouco sobre o funcionamento desses modelos e porque das mudanças.

No decorrer da modelagem observou-se que se poderia fazer a junção desses dois autômatos em um único autômato para representar o sensor, juntando os autômatos de frequência e o do sensor, isto reduziu o número de estados, pois na composição paralela desses autômatos temos oito estados, enquanto que desta forma temos quatro estados apenas, melhorando o desempenho para a compilação.

Durante a modelagem percebeu-se que o sensor só teria alguma frequência associada a ele se o mesmo estivesse ligado, pensando nisso notou-se que se poderia fazer a junção em um único autômato que associaria o estado do sensor e sua frequência, para isso foi acrescentando uma variável de estado para representar o status do sensor em cada frequência, essa variável retorna *true* para o sensor ligado e *false* para o sensor desligado, com mostra a figura 23 a seguir:

```
node frequency(u1, u2, u3, d1, d2, d3: bool) returns (s,l1,l2,l3:bool)
 let
    automaton
      state NC do
        s = false; l1 = false; l2 = false; l3 = false;
      unless u1 then L1 |
             u2 then L2 |
             u3 then L3
      state L1 do
        s = true; l1 = true; l2 = false; l3 = false;
      unless u1 then L2 |
             u2 then L3 |
             d1 then NC
      state L2 do
         s = true; l1 = false; l2 = true; l3 = false;
      unless u1 then L3 |
             d1 then L1 |
             d2 then NC
      state L3 do
         s = true; l1 = false; l2 = false; l3 = true;
      unless d1 then L2 |
             d2 then L1 |
             d3 then NC
    end
tel
```

Figura 23 – Autômato para representar o modelo do sensor e sua frequência

A figura 23 nos mostra o que não existe mais dois autômatos separados para o sensor e sua frequência, tem-se um único autômato com quatro estados NC que é o estado em que o sensor se encontra desligado por consequência não possui nenhum nivel de frequência associada a esse estado, pode-se observar isso olhando para as variáveis de estado s relacionada ao status do sensor e $l1$, $l2$, $l3$ para as frequências baixa, média e altas todas retornando *false* no estado NC.

O estado $L1$, para o sensor ligado em frequência baixa, nesse caso diferente do estado NC, a variável de estado s retorna *true* e a variável de estado $l1$ retorna *true*, $l2$ e $l3$ retornam *false*. A mesma logica se aplica para os estados $L2$ em que o sensor se encontra ligado capturando em frequência média e $L3$ para o sensor ligado capturando em frequência alta frequência alta.

Após a modelagem desses subsistemas o próximo passo foi introduzir o cenário de falha no sistema, que não estava presente nas primeiras versões da modelagem. A modelagem do autômato que representa a falha, tanto falha do sensor como falha da estação, segue a logica dos autômatos do sensor e da estação já modelados.

Apresenta dois estados em falha e funcionando, tem uma variável de estado f, para representar a falha, *true* para dispositivo em falha e *false* para dispositivo funcionando. O autômato tem associado a eles dois eventos não controláveis que é o evento de controle de falha de dispositivo representando por FC e de controle de recuperação do dispositivo representando RC como mostra a figura 24.

Vale lembrar que como o evento é não controlável o controlador não tem como sabe quando o dispositivo vai falhar, porém, caso ocorra esse cenário o controlador sabe o que fazer, ele sabe quais eventos habilitar e desabilitar, tendo por base as regras estabelecidas no modelo, que vai garantir, mediante a esse cenário, que o controlador atue da maneira desejada.

```
node fail(fc,rc: bool) returns (f:bool)
let
    automaton
      state Funcionando do
        f = false;
      unless fc then Falha
      state Falha do
        f = true;
      unless rc then Funcionando
    end
tel
```

Figura 24 – Autômato para representara falha

Tendo todos esses modelos dos dispositivos, o controlador vai fazer a composição desses autômatos, um controle monolítico, como foi explicado no capítulo dois.

Prosseguindo com a modelagem se observou que era possível encapsular um conjunto de autômatos e fazer sua composição de forma manual, melhorando o desempenho do sistema. Pois além de todos os benefícios de usar encapsulamento a composição manual desses autômatos reduz o número de composições que ficam a cargo do controlador realizar aumentando seu desempenho. A figura 23 a seguir mostra como ficou o encapsulamento desses autômatos.

No encapsulamento mostrado na figura 25 seis autômatos são instanciados em um único *node* como mostra a figura, cada *inlined* é um autômato sendo estanciado, sendo dois autômatos de frequência, que é composto pelo sensor e sua frequência, um para temperatura e uma para umidade. Um autômato de estação e três autômatos de falha, uma falha para a estação e duas falhas para cada sensor estanciado e algumas variáveis auxiliares.

```
node stationsystem(c_e_t_u1,c_e_t_u2,c_e_t_u3,c_e_t_d1,c_e_t_d2,c_e_t_d3,
                   c_e_h_u1,c_e_h_u2,c_e_h_u3,c_e_h_d1,c_e_h_d2,c_e_h_d3,
                   c_e,e_t_fc,e_t_rc,e_h_fc,e_h_rc,e_fc,e_rc:bool)
returns (e_s,e_t_s,e_h_s,e_t_l1,e_t_l2, e_t_l3,
         e_h_l1, e_h_l2, e_h_l3, e_t_f, e_h_f, e_f:bool)
var
    aux1, aux2, aux3, aux4: bool;
let
    (e_t_s,e_t_l1,e_t_l2,e_t_l3) = inlined frequency(c_e_t_u1,c_e_t_u2,c_e_t_u3,
                                                     c_e_t_d1,c_e_t_d2,c_e_t_d3);
    (e_h_s,e_h_l1,e_h_l2, e_h_l3) = inlined frequency(c_e_h_u1,c_e_h_u2,c_e_h_u3,
                                                      c_e_h_d1,c_e_h_d2,c_e_h_d3);
    e_s = inlined station(c_e);
    e_f = inlined fail(e_fc,e_rc);
    aux1 = e_t_fc or e_f;
    aux2 = e_h_fc or e_f;
    aux3 = e_t_rc or e_rc;
    aux4 = e_h_rc or e_rc;
    e_t_f = inlined fail(aux1,aux3);
    e_h_f = inlined fail(aux2,aux4);
tel
```

Figura 25 – Encapsulamento dos autômatos da estação, frequência e falha

Aqui vale a pena uma descrição maior para o melhor entendimento do que foi feito. O *node stationsystem* recebe os determinados eventos relacionados as frequências, estações e falhas como segue:

- $stationsystem(c\_e\_t\_u1, c\_e\_t\_u2, c\_e\_t\_u3, c\_e\_t\_d1, c\_e\_t\_d2, c\_e\_t\_d3, c\_e\_h\_u1, c\_e\_h\_u2, c\_e\_h\_u3, c\_e\_h\_d1, c\_e\_h\_d2, c\_e\_h\_d3, c\_e, e\_t\_fc, e\_t\_rc, e\_h\_fc, e\_h\_rc, e\_fc, e\_rc : bool)$

Descrevendo cada um desses eventos tem-se:

1. c_e, e_fc,e_rc = Eventos relacionados ao status estação, sua falha e recuperação.
2. e_t_fc,e_t_rc = Eventos relacionados ao status do sensor de temperatura, sua falha e recuperação.
3. c_e_t_u1,c_e_t_u2,c_e_t_u3 = Eventos controláveis relacionado a temperatura em que a frequência aumenta, ou seja, recebe um *up*.
4. c_e_t_d1,c_e_t_d2,c_e_t_d3 = Eventos controláveis relacionado a temperatura em que a frequência diminui, ou seja, recebe um *donw*.
5. e_h_fc,e_h_rc = Eventos relacionados ao status do sensor de umidade, sua falha e recuperação.
6. c_e_h_u1,c_e_h_u2,c_e_h_u3 = Eventos controláveis relacionado a umidade em que a frequência aumenta, ou seja, recebe um *up*.

7. c_e_h_d1,c_e_h_d2,c_e_h_d3 = Eventos controláveis relacionado a umidade em que a frequência diminui, ou seja, recebe um *donw*.

Esses são os eventos que esse *node* recebe, cabe ao controlador baseado nas regras decidir o que fazer de acordo com cada evento que foi disparado. Como retorno do *stationsystem* têm-se:

- $returns(e\_s, e\_t\_s, e\_h\_s, e\_t\_l1, e\_t\_l2, e\_t\_l3, e\_h\_l1, e\_h\_l2, e\_h\_l3, e\_t\_f, e\_h\_f, e\_f : bool)$

Esses retornos vão representar:

1. e_s = vai retornar *true* ou *false* para representar o status da estação, ligada ou desligada.
2. e_t_s = vai retornar *true* ou *false* para representar o status do sensor de temperatura, ligada ou desligado.
3. e_h_s = vai retornar *true* ou *false* para representar o status do sensor de umidade, ligado ou desligado.
4. e_t_l1,e_t_l2, e_t_l3 = vai retornar *true* ou *false* para representar a frequência em que o sensor de temperatura está operando, baixo, médio ou alto.
5. e_h_l1, e_h_l2, e_h_l3 = vai retornar *true* ou *false* para representar a frequência em que o sensor de umidade esta operando, baixo, médio ou alto.
6. e_t_f, e_h_f, e_f = vai retornar *true* ou *false* para representar se algum dispositivo esta em falha, sensor de temperatura, umidade ou estação.

Com isso, a composição e encapsulamento desses dispositivos é feita, cabendo ao controlador só o gerenciamento desas composição com base nas regras previamente estabelecidas e eventos disparados.

Por fim temos a parte mais importante do modelo, as regras. Essas regras são escritas seguindo um padrão, tem-se um trecho referente à condição atual e após isso um trecho dizendo o que fazer diante aquela situação.

Como já foi citado o Heptagon utiliza a lógica proposicional básica, essa ferramente não possui elementos como o XOR e o se, ou seja, todo o conjunto de regras é montado à partir de proposições, das quais se pode dizer que são enunciados com valores de verdadeiro ou falso.

As proposições podem ser do tipo simples, com apenas uma proposição, ou composta, que é constituída por duas ou mais proposições simples articuladas entre si. Desta forma, o código deve ser escrito mapeando as proposições não atendidas.

Como um exemplo do uso dessa logico tem-se:

- *not* sensor de temperatura estação um *and not* sensor de umidade estação um $\Rightarrow$ *not* estação um

Essa ideia de regra nos informa que caso o sensor de temperatura e o sensor de umidade da estação um estiverem desligados então desligue a estação um.

Tendo isso em mente vamos agora mostrar como ficou a implementação de fato de cada regra dentro do modelo.

Inicialmente temos as regras referente ao status da estação um, são elas:

1. $\neg(\neg e1\_t\_s \wedge \neg e1\_h\_s) \vee (\neg e1\_s)$.
2. $\neg((e1\_t\_s \vee e1\_h\_s) \wedge (\neg e1\_f)) \vee (e1\_s)$.

Essas regras servem para informar ao controlador que a estação um deve ficar desligada quando os sensores estão sem uso e deve ser ligada quando pelo menos um sensor está em uso.

$\neg(\neg e1\_t\_s \wedge \neg e1\_h\_s)$ esse trecho serve para informar que se o sensor de temperatura da estação um estiver desligado e o sensor de umidade da estação um também estiver desligado faça algo, o próximo trecho do código diz o que fazer caso essa condição seja verdade, $\vee(\neg e1\_s)$ o trecho diz para então desligar a estação um. Têm-se as mesmas regras, escritas da mesma forma para estação dois.

Após esse primeiro conjunto de regras referente ao status da estação têm-se regras para o uso dos sensores de temperatura em ambas as estação:

1. $\neg((uth \vee utm \vee utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_s \wedge \neg e2\_t\_s) \vee (\neg e1\_t\_s \wedge e2\_t\_s))$.
2. $\neg((\neg uth \wedge \neg utm \wedge \neg utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee (\neg e1\_t\_s \wedge \neg e2\_t\_s)$.
3. $\neg((uth \wedge \neg utm \wedge \neg utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l3 \vee e2\_t\_l3))$.
4. $\neg((\neg uth \wedge utm \wedge \neg utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l2 \vee e2\_t\_l2))$.

5. $\neg((\neg uth \wedge \neg utm \wedge utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l1 \vee e2\_t\_l1))$.

Aqui temos regras para o controle do sensor e sua frequência, essas regras nos informam, por exemplo, que caso não exista uma solicitação de usuário para capturar temperatura sem se importar com a frequência, $\neg((uth \vee utm \vee utl)$, e caso os sensores de temperatura não estiverem em falha, $(\neg(e1\_t\_f \wedge e2\_t\_f)$, e as estações também não estiverem em falha, $neg(e1\_t\_f \wedge e2\_t\_f))$, então a regra diz para o controlador executar a captura em qualquer estação com qualquer frequência $((e1\_t\_s \wedge \neg e2\_t\_s) \vee (\neg e1\_t\_s \wedge e2\_t\_s))$, nesse caso se a coleta for realizada na estação um não utilizar a estação dois e vise versa.

As regras seguintes seguem a mesma ideia, uma regra para caso não exista solicitação de usuário e os sensores e estações não se encontrarem em estado falha, $\neg((\neg uth \wedge \neg utm \wedge \neg utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f)))$ então desligue os sensores de temperatura de ambas as estações, $(\neg e1\_t\_s \wedge \neg e2\_t\_s)$.

As últimas três regras dessa sequência nos informam que caso o usuário deseje capturar a temperatura em alguma frequência especifica, no caso alta, $\neg((uth \wedge \neg utm \wedge \neg utl)$, média, $\neg((\neg uth \wedge utm \wedge \neg utl)$ ou baixa, $\neg((\neg uth \wedge \neg utm \wedge utl)$, e os sensores e estação não estiverem em falha como mostrado nas regras anteriores então realizar a captura na frequência requisita pelo usuário, ou seja, na frequência alta $((e1\_t\_l3 \vee e2\_t\_l3))$, media $((e1\_t\_l2 \vee e2\_t\_l2))$ ou baixa $((e1\_t\_l1 \vee e2\_t\_l1))$ na estação um (e1) ou na estação dois (e2).

Existem ainda no modelo mais cinco regras para o controle da frequência e do sensor referentes à captura de dados da umidade que seguem a mesma ideia das regras mostrada para o dado de temperatura.

Por fim o modelo também conta com algumas regras para o controle de falhas, são elas:

- Controle de falha para os sensores de temperatura e umidade

  1. $\neg(e1\_t\_f) or (\neg e1\_t\_s)$.
  2. $\neg(e2\_t\_f) or (\neg e2\_t\_s)$.
  3. $\neg(e1\_h\_f) or (\neg e1\_h\_s)$.
  4. $\neg(e2\_h\_f) or (\neg e2\_h\_s)$.

- Controle de falha para as estações de coleta de dados.

  1. $\neg(e1\_f) or (\neg e1\_s)$.
  2. $\neg(e2\_f) or (\neg e2\_s)$.

Essas regras são mais fáceis de compreender que as anteriores, elas nos informam que caso algum componente entre em estado de falha, no caso se o sensor de umidade da estação um entra em falha, $\neg(e1\_h\_f)$, então não use esse sensor de umidade, $(note1\_h\_s)$, a mesma ideia vale para os outros sensores e para as estações de coleta de dados.

4.3 Testes do modelo

Nessa seção vamos demostrar como de fato funcionam os subsistemas que compõem o modelo, ou seja, como se comporta cada autômato mostrado no capítulo três de acordo com uma entrada especifica, mostrar passo a passo de como se comporta cada autômato e a saída esperada para determinado cenário.

4.3.1 Cenário inicial

Antes de mostrar como o modelo funciona frente a alguns cenários com e sem falhas, é importante mostrar o que seria a configuração inicial do sistema, ou seja, qual seria um cenário possível dos estados dos autômatos assim que o sistema é iniciado, pode ser pensando como uma configuração inicial do sistema, que é representado na figura 26.

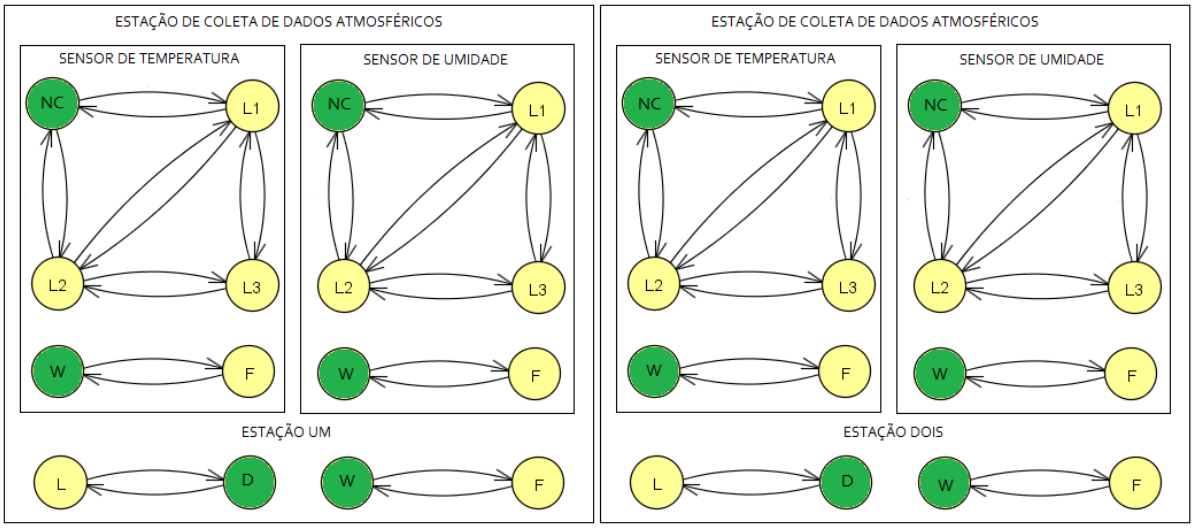

Figura 26 – Autômatos do modelo

A figura 26 representa o que seria as estações de coleta de dados do nosso modelo. Temos na imagem os sensores de temperatura e a umidade, que são compostos pelos sensores de frequência e o de falha, e mais externamente temos a estação de coleta que é composta pelos autômatos dos sensores e mais os de falha da estação e o de status da estação.

Marcado de verde estão os estados dos autômatos em sua configuração inicial:

- Estado do autômato do sensor NC, não conectado.
- Estado do autômato de falha do sensor W, *work* (funcionando).
- Estado do autômato da estação D, desligado.
- Estado do autômato de falha da estação W, *work* (funcionando).

É importante notar que, nesse caso, como todos os sensores das respectivas estações estão sem uso então de acordo com as regras a seguir:

$$\neg(\neg e1\_t\_s \wedge \neg e1\_h\_s) \vee (\neg e1\_s)$$

$$\neg(\neg e2\_t\_s \wedge \neg e2\_h\_s) \vee (\neg e2\_s)$$

Essas regras informa ao controlador que caso os sensores das estações estiverem sem uso colocar as estações no estado de desligado, assim sendo, o controlador ira desligar a estação, ate algum usuário solicitar a captura e o sensor começar a funciona, os autômatos estaria nos seguintes estados.

4.3.2 Cenários Sem falhas

Conforme foi mostrado na seção anterior, o modelo funciona, de forma bastante resumida, da seguinte forma: O usuário faz uma requisição de temperatura ou umidade, na frequência desejada, ou seja, baixa, média ou alta, o sistema pega essa solicitação e busca atendê-la da melhor maneira possível.

Tendo isso em mente vamos mostrar como seria a configuração do sistema de acordo com a requisição do usuário e as regras estabelecidas no modelo.

Vamos inicialmente pensar em um cenário em que o usuário faz um requisição para capturar dados relacionados a temperatura em uma frequência de captura baixa.

A figura 27 abaixo mostra a nova configuração que o sistema deve assumir:

Como podemos ver na imagem o controlador ligou a estação um e colocou o autômato do sensor de temperatura da estação em captura lenta, $L1$, assim como solicitou o usuário.

Como o sistema nesse cenário não apresenta nenhuma falha, o controlador conseguiu anteder o que o usuário solicitou sem maiores problemas, nesse caso ligar a estação e capturar dados da temperatura em frequência baixa.

Baseado na regra do modelo que diz:

$$\neg((\neg uth \wedge \neg utm \wedge utl) \wedge (\neg(e1\_t\_f \& e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l1 \vee e2\_t\_l1))$$

$$\neg((e1\_t\_s \vee e1\_h\_s) \wedge (\neg e1\_f)) \vee (e1\_s)$$

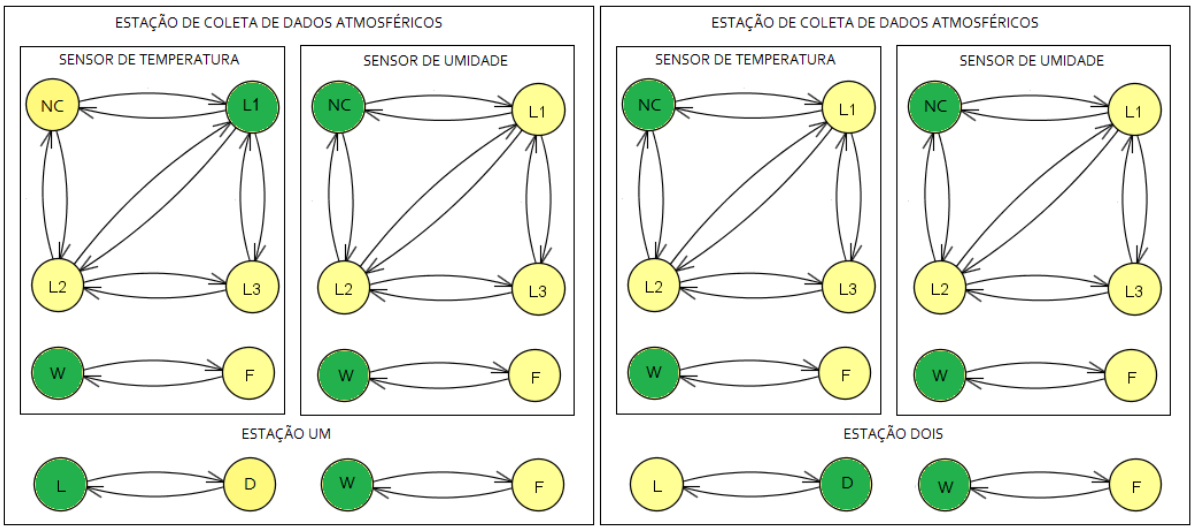

Figura 27 – Configuração dos autômatos após primeira requisição

Ou seja, de acordo com a primeira regra mostrada se usuário solicitar capturar temperatura numa frequência baixa e tanto os sensores como as estações estiverem em falha o controlador pode escolher capturar o dado usando o sensor de temperatura de qualquer estação, nesse caso ele escolheu capturar usado a estação um, para isso precisou ligar a estação de acordo com a segunda regra mostrada que diz que se o sensor de temperatura ou o sensor de umidade da estação estiver ligado e a estação não estiver em falha a estação deve ser ligada para realizar a captura como mostra a figura 27.

O autômato do sensor vai receber um evento que vai ocasionar a transição do estado não conectado em que estava e passar para o estado $L1$, que significa que está operando capturando o dado na frequência baixa. Os outros autômatos permanecem nos mesmos estados.

Agora imagine que esse mesmo usuário queira capturar a temperatura numa frequência alta e deseje agora também capturar umidade numa frequência alta. A figura 28 baixo mostra a nova configuração do sistema.

De novo, como não havia nenhum tipo de falha, para atender essa requisição o controlador se baseou nas seguintes regras:

$$\neg((uth \wedge \neg utm \wedge \neg utl) \wedge (\neg(e1\_t\_f \& e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l1 \vee e2\_t\_l1))$$

$$\neg((uhh \wedge \neg uhm \wedge \neg uhl) \wedge (\neg(e1\_t\_f \& e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l1 \vee e2\_t\_l1))$$

Essas regras nos informam o seguinte, a primeira garante que se o usuário solicitar capturar temperatura na frequência alta e os sensores e nem as estações estiverem em falha, o controlador pode usar qualquer sensor de qualquer estação para

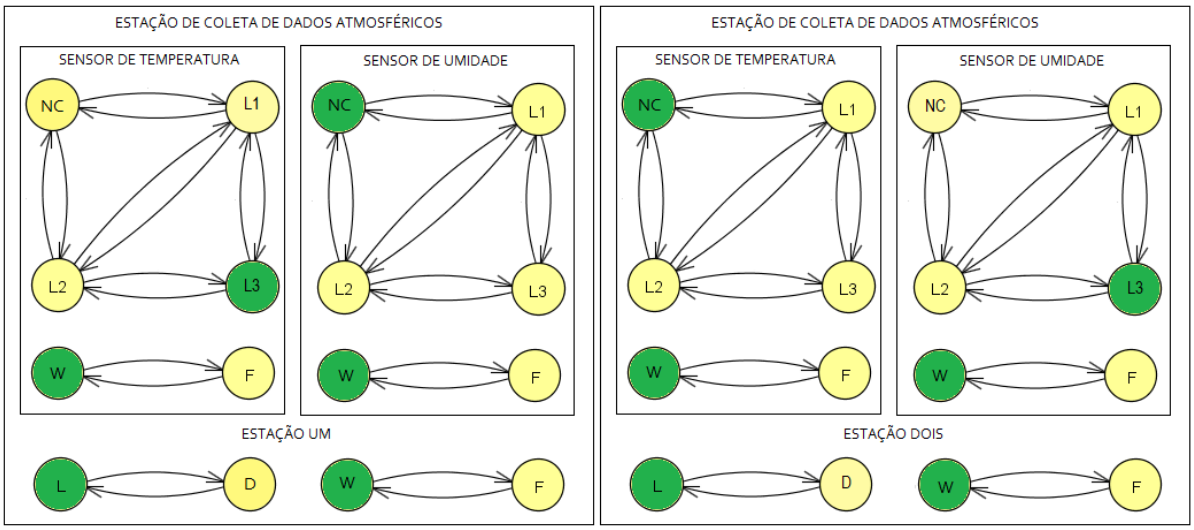

Figura 28 – Configuração dos autômatos após segunda requisição

realizar a captura, a outra regra diz a mesma coisa para umidade.

Observando a figura 28 nota-se que o controlador mudou a taxa de captura do sensor de temperatura da estação um para $L3$, ou seja, passou a capturar a temperatura em frequência alta, além disso utilizou o sensor de umidade da estação dois para realizar a captura, já que ambos os sensores e estações estão funcionado sem falhas o controlador fica livre para escolher qualquer uma para realizar a captura.

Nesses cenários onde não se tem nenhum dispositivo em falha, o controlador não terá que desabilitar nenhum evento. Na próxima seção iremos ver como o controlador se comporta caso algum dispositivo falhe.

4.3.3 Cenários com falhas

Após ver como funciona o modelo com seus dispositivos estão funcionando corretamente, nessa seção vamos mostrar o que acontece quando ocorre algum imprevisto, no caso quando a estação ou o sensor ou os dois falham e observar como o controlador se comporta mediante esse tipo de cenário.

Vamos pensar no mesmo cenário inicial descrito na seção anterior, onde o usuário faz um requisição para capturar dados relacionados a temperatura em uma frequência de captura baixa.

Agora vamos considerar para esse novo cenário que o sensor que é responsável pela captura de temperatura da estação um se encontra em falha, ou seja, o sistema estava capturando temperatura com uma frequência alta e o sensor entrou em estado de falha como mostra a figura 30 seguir.

Como foi mostrado no capítulo anterior, no modelo existe regras que o controlador tem que seguir para que o sistema funcione como previsto, portanto quando acon-

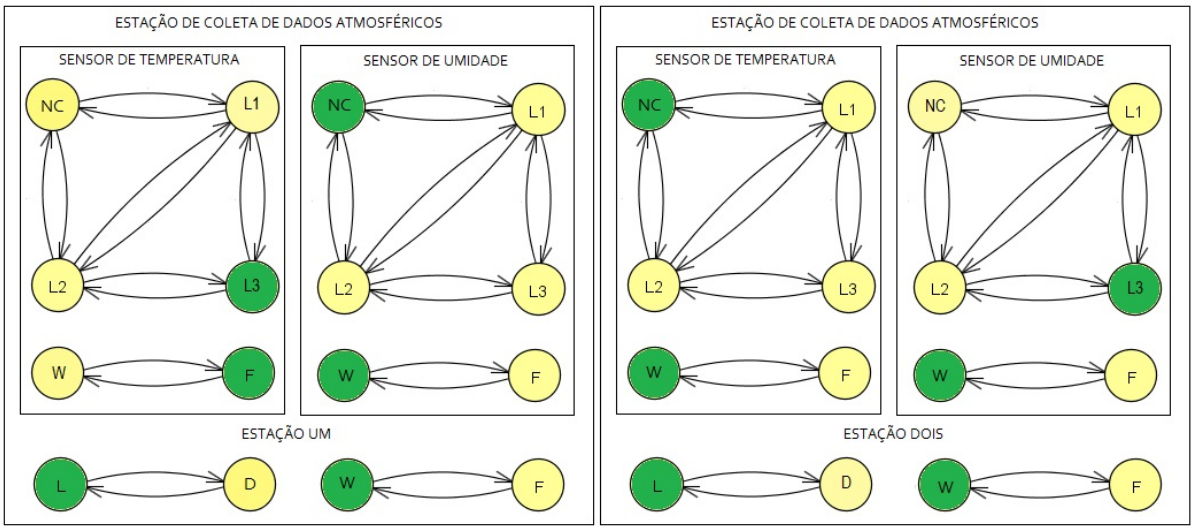

Figura 29 – Configuração dos autômatos em cenário com falha

tece uma falha, esse cenário tem que estar previsto nas regras, para que o controlador tome a decisão esperada.

Vamos supor que, nesse cenário, o usuário solicitou alterar a captura da temperatura para uma frequência média. Nesse caso como esse sensor de temperatura da estação um que estava realizando a captura se econtra em falha o controlador vai ter que obedecer as seguintes regras:

$$\neg((\neg uth \wedge utm \wedge \neg utl) \wedge (\neg(e1\_t\_f \wedge e2\_t\_f)) \wedge (\neg(e1\_f \wedge e2\_f))) \vee ((e1\_t\_l1 \vee e2\_t\_l1))$$

$$\neg(e1\_t\_f) \vee (\neg e1\_t\_s)$$

Essas regras informam ao controlador, primeiro, que se o usuário requisitar capturar temperatura numa frequência média, o controlador precisa checar se as estações ou sensores estão em falha, nesse caso o sensor da estação um se encontra no estado de falha, com isso o controlador vai desabilitar o evento que iria fazer com que esse sensor passasse a capturar a temperatura em frequência media, $L2$ e utilizar a estação dois, que o sensor de temperatura não está em falha, para realizar a captura do dado.

A regra seguinte diz que se o sensor de temperatura da estação um estiver em falha não use esse sensor, ou seja, desligue esse sensor.

Agora como a estação um está com o sensor de temperatura em falha e o de umidade não esta em uso, o controlador desliga a estação um como diz a regra a baixo:

$$(\neg e1\_t\_s \wedge \neg e1\_h\_s) \vee (\neg e1\_s)$$

Tendo em vista essas três regras do modelo, o controlador mediante a esse cenário, configura o sistema como mostra a figura a baixo:

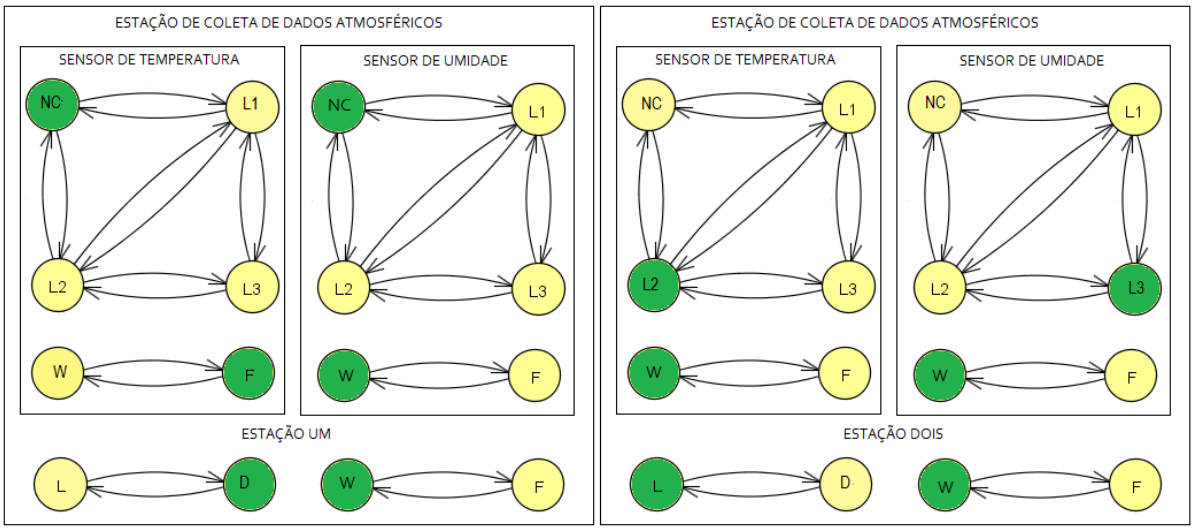

Figura 30 – Configuração dos autômatos em cenário com falha

A figura 30 mostra a nova configuração do sistema onde a estação um está desligada, pois o sensor de temperatura está em falha e o sensor de umidade está sem uso, e a estação dois funcionado normalmente capturando os dados de temperatura e umidade.

É importante ressaltar que enquanto o autômato de temperatura estiver em falha todo evento que seria direcionado a ele vai ser desabilitado pelo controlador até que ele volte para o estado de funcionando.

Resumindo um pouco o que acontece, sempre que o sistema receber uma requisição de um usuário, o controlador vai fazer uma espécie de verificação de status tanto nos sensores como nas estações e verificar as regras para saber o que fazer frente a determinadas situações.

Lembrando que esses passos estão sendo mostrado no autômato individual de cada subsistema, mas para o controlador tudo ocorre em um único autômato, por conta da composição paralela que é realizada.

Caso a falha seja da estação o procedimento é o mesmo, a única mudança é que o autômato que relaciona a falha da estação passará para o estado de falha e a estação é desligada.

Levando em consideração o cenário atual imagine que agora a estação dois que estava realizando a captura entre em falha. A figura 31 mostra como ficou a nova configuração dos autômatos.

Nessa imagem temos controlador agindo baseado na seguinte regra:

$$\neg(e2\_f) \vee (\neg e2\_s)$$

Essa regra informa ao controlador para não usar a estação dois para realizar a

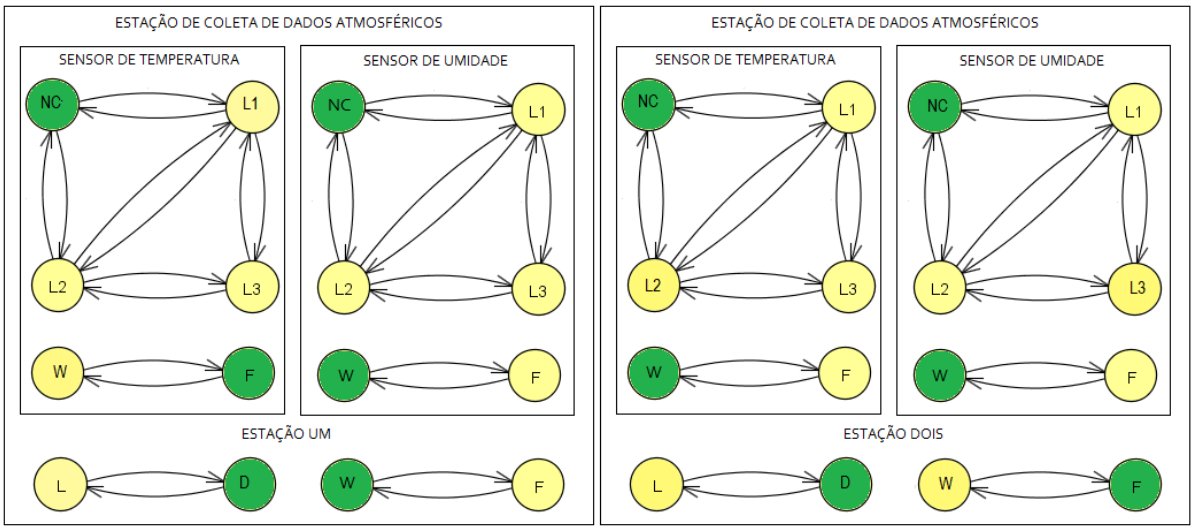

Figura 31 – Estado atual do autômato da estação

captura dos dados tanto de temperatura quanto de umidade se a estação estiver em falha, ou seja, a partir disso o controlador vai passar a desabilitar todos os eventos que ocorreriam nessa estação e passar a capturar na estação de coleta um, caso seja possível.

5 Conclusão

5.1 Objetivos alcançados

O objetivo principal que é a síntese automática de controladores em ambientes de sensoriamento como serviço, foi alcançado, foi possível mostrar que a abordagem de controle de sistemas inteligentes a partir do uso da teoria do controle supervisório é satisfatória.

Foi demostrado a partir do desenvolvimento e validação experimental em um sistema real que é possível através do uso da teoria do controle supervisório, a partir de certas especificações no modelo, gerar um controle para esse sistema que garante seu funcionamento correto mediante os diferentes tipos de cenário propostos no modelo.

A partir da modelagem dos dispositivos, da modelagem do ambiente e da criação regras de controle foi possível sintetizar esse controlador de maneira automática. Esse controlador sintetizado a partir da teoria do controle supervisório foi capaz de garantir com sucessor todas as regras de funcionamento proposta no modelo garantindo assim o bom funcionamento das estações de coleta de dados frente a diversos cenários de funcionamento diferentes.

Foram estipulados alguns objetivos secundários a fim de se alcançar o objetivo principal.

- Modelar sensores e estações de coleta de dados: Tanto as estações como os sensores foram devidamente modelados e apresentados durante esse TCC. Foram modelados quatro autômatos, uma pra os sensores, uma para a frequência, um para a estação de coleta de dados e um para a falha tanto dos sensores como da estação, o sistema e composto de duas estações cada estação com dois sensores e foi feita a composição paralela de todos esses autômatos gerando aproximadamente sessenta e cinco mil e quinhentos e trinta e seis estados na composição levando cerca de uma hora e meia para a compilação do modelo.
- Modelar regras de controle para estação e sensores: Foram criadas vinte regras no modelo para cobrir vários cenários de funcionamento, regras para as estações, sensores, frequências e falhas como foi mostrado no decorrer do TCC.
- Sintetizar controladores: por fim com os modelos das coisas, o modelo do ambiente e as regras de controle, com isso foi possível sintetizar o controlador e alcançar o principal objetivo proposto no trabalho.

5.2 Futuras implementações

Para futuras implementações, têm-se ideias de cobrir mais cenários de funcionamento nas regras do modelo.

Foi pensando em regras para o controle de energia que influenciariam nas capturas dos dados, tantos os sensores quanto às estações usam algum tipo de energia, digamos que possuam uma bateria, o nível dessa bateria poderia influencia a captura da seguinte maneira: caso a bateria esteja com pouca carga o controlador desabilitaria os eventos relacionada às capturas de dados em frequências medias e altas visto que consumiriam mais energia.

O nível da bateria também poderia influenciar a quantidade disponível de sensores que poderiam esta capturando os dados simultaneamente, digamos, caso a bateria esteja baixa somente um sensor poderia capturar os dados por vez na estação.

Foi pensando também em um meio de enviar os dados capturados, que não foi coberto na presente modelagem. Determinar a melhor forma possível de fazer o envio do dado capturado, quais e quantos sensores usar, quais seriam os eventos para envio de dados, qual o estado da rede para envio de dados entre outras questões necessárias para realizar o envio dos dados.

A relação da bateria com os sensores poderia interferir tanto na captura como no envio dos dados, imagine a situação em que a estação esta capturando dados e a bateria fica no estado de bateria fraca, e esses dados ainda precisariam ser enviados, poderia se imaginar cenários em que o sistema priorizaria o envio, ou seja, pararia naquele momento a captura e focaria o que restar de bateria no envio dos dados já capturados ou continuaria capturando e gravando esses dados e só enviaria após a bateria ser trocada ou recarregada.

O modelo teria novos e diferentes cenários a se cobrir estabelecendo novas regras de funcionamento que levaria em conta tanto energia da estação e dos sensores, como o envio de dados, e a relação entre esses dois eventos.

5.3 Dificuldades encontradas

Para se alcançar o objetivo proposto, foi necessário um estudo muito grande para poder dominar uma serie de conceitos e teorias para o desenvolvimento desse trabalho, como por exemplo, o uso da teoria do controle supervisórios pra sintetizar o controlador e otimizar o desempenho do modelo, modelagem sistemas de eventos discretos, processos dirigidos a eventos, tendo isso em vista um ponto que dificultou essa aprendizagem foi o pouco material sobre os assuntos abordados no TCC disponibilizados na internet.

Outro ponto de dificuldade no decorrer do TCC, foi no uso das ferramentas pra modelagem dos autômatos e para a síntese dos controladores, que são o Heptagon e o Sigali, muito sobre essas ferramentas foi aprendido na prática, na tentativa e erro, visto também o pouco de material sobre essas ferramentas disponível, sendo seus manuais a principal fonte de conhecimento, a quantidade de fóruns, artigos entre outras fontes que poderiam ajudar são bastante escassas o que dificulta muito a resolução de problemas relacionada ao uso dessas ferramentas na modelagem.

As limitações do ambiente criado pelo Cloud9 também dificultaram um pouco, pois devido o alto consumo de memoria por conta da composição paralela dos autômatos modelos levava muito tempo para se compilar. Isso dificultou a realização de testes uma vez que a compilação demorava tanto não valia a pena fazer pequenas mudanças e ir testando aos poucos, na hora da modelagem teria que fazer uma espécie de compilação na mete do que possivelmente ocorreria e tratar antes que de fato colocasse o modelo pra compilar e verificar se realmente estava tudo certo, por conta disso o processo de modelagem demorou em ser finalizado.

Essas foram as principais dificuldades enfrentas para realização desse trabalho. Mas devido a essas dificuldades muito foi aprendido durante esse processo tanto na questão teórica nos conceitos da teoria do controle supervisório, modelagem e da criação de SED como na pratica, sendo uma experiência bastante positiva, uma excelente oportunidade e aprender sobre modelos, e pensamentos computacional, assuntos bastante interessantes que om certeza via ser bastante útil no decorrer da vida profissional.

Referências

BRANDIN, B. A.; MALIK, R.; MALIK, P. Incremental verification and synthesis of discrete-event systems guided by counter examples. *IEEE Transactions on Control Systems Technology*, v. 12, n. 3, p. 387–401, May 2004. ISSN 1063-6536. Citado na página 29.

BRANDIN B.A; CHARBONNIER, F. The supervisory control of the automated manufacturing system of the aip. *International Conference on Computer Integrated Manufacturing and Automation Technology*, p. 319–324, 1994. Citado na página 40.

CASSANDRAS, C. G.; LAFORTUNE, S. *Introduction to discrete event systems*. 2. ed. ed. New York, NY: Springer, 2008. OCLC: 255370614. ISBN 978-0-387-33332-8 978-1-4419-4119-0 978-0-387-68612-7. Citado 8 vezes nas páginas 7, 22, 23, 28, 29, 31, 32 e 33.

CURY, J. *Teoria de Controle Supervisório de Sistemas a Eventos Discretos*. Canela-RS: V Simpósio Brasileiro de Automação Inteligente, 2001. v. 1. Citado 4 vezes nas páginas 7, 26, 37 e 39.

HEPTAGON/BZR manual. 2017. Citado na página 55.

INMET. Rede de estações meteorológicas automáticas do inmet. 2011. Citado 3 vezes nas páginas 7, 44 e 45.

LIMA, S. T. D. S. *Bases mínimas para o diagnóstico de falhas em sistemas a eventos discretos*. 2008. Citado 4 vezes nas páginas 7, 35, 36 e 37.

PERERA A. ZASLAVSKY, P. C. D. G. C. Sensing as a service model for smart cities supported by internet of things. 2014. Citado na página 15.

QUEIROZ, J. E. R. C. Max H. de. Controle supervisório modular de sistemas de manufatura. *Sba: Controle & Automação Sociedade Brasileira de Automatica*, v. 13, n. 2, p. 123–133, 2002. Citado na página 40.

Ramadge, P.J.G.; Wonham, W.M. *The control os Discrete Event Systems*. 1989. Citado 7 vezes nas páginas 7, 38, 39, 40, 41, 42 e 43.

SIGALI User's manual. 2004. Citado na página 55.

SILVA, D. B. *Aplicaçao da teoria de controle supervisório no projeto de controladores para sistemas de rota variável centrado em robô ppgeps*. Tese (Doutorado) — Pontifícia Universidade Católica do Paraná, 2007. Citado 5 vezes nas páginas 7, 33, 34, 35 e 41.

VIEIRA, A. D. Método de implementação do controle de sistemas e eventos discretos com aplicação da teoria de controle supervisório. 2007. Citado na página 30.

WEISS S. DELAERE, W. H. L. M. B. H. Sensing as a service: An exploration into practical implementations of dsa. 2010. Citado na página 15.

ZHAO, M. *Discrete Control in the Internet of things and Smart Environments through a Shared Infrastructure*. Tese (Doutorado) — Université Grenoble Alpes, 2015. Disponível em: <https://tel.archives-ouvertes.fr/tel-01163806/>. Citado na página 15.

ZHAO, M. et al. Discrete control for the internet of things and smart environments. In: *Presented as part of the 8th International Workshop on Feedback Computing*. [s.n.], 2013. Disponível em: <https://www.usenix.org/node/174698>. Citado na página 15.