UNIVERSIDADE FEDERAL DO CEARÁ
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
CENTRO DE TECNOLOGIA

UMA CONTRIBUIÇÃO AO DESENVOLVIMENTO E AVALIAÇÃO DA QUALIDADE DE SISTEMAS DE SUPERVISÃO INDUSTRIAL À LUZ DAS NORMAS ISO/IEC 9126 E 14598

ÉRICK ARAGÃO RIBEIRO

FORTALEZA
2013

ÉRICK ARAGÃO RIBEIRO

UMA CONTRIBUIÇÃO AO DESENVOLVIMENTO E AVALIAÇÃO DA QUALIDADE DE SISTEMAS DE SUPERVISÃO INDUSTRIAL À LUZ DAS NORMAS ISO/IEC 9126 E 14598

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática da Universidade Federal do Ceará como requisito para obtenção do grau de Mestre em Engenharia de Teleinformática.

Área de Concentração: Sinais e Sistemas

Orientador: Prof. Dr. George André Pereira Thé

FORTALEZA
2013

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca de Pós-Graduação em Engenharia - BPGE

R368c Ribeiro, Érick Aragão.
Uma contribuição ao desenvolvimento e avaliação da qualidade de sistemas de supervisão industrial à luz das normas ISO/IEC 9126 e 14598 / Érick Aragão Ribeiro. – 2013
163 f. : il. color., enc. ; 30 cm.

Dissertação (Mestrado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia de Teleinformática, Fortaleza, 2013.
Orientação: Prof. Dr. George André Pereira Thé.
Área de concentração: Sinais e Sistemas

1. Teleinformática. 2. Software – Controle de qualidade. 3. Lógica difusa. I. Título.

CDD 621.38

AGRADECIMENTOS

A Deus por ter sempre me orientado como fazer pra ser vitorioso e em quem deposito minha fé. Meu porto seguro, minha fortaleza. Louvado sejam seus conselhos e sua doutrina. Grato, Senhor, por ter muito mais a agradecer do que a pedir.

Aos meus pais pela paciência e companhia nas alegrias e tristezas da vida.

Aos meus irmãos pela alegria que me trazem quase sempre.

A toda minha família pela amizade e carisma.

Ao professor George Thé por sua orientação, motivação e entusiasmo de pesquisador.

Ao professor José Marques por sua co-orientação e amizade.

Aos professores Tarcísio, Guilherme e Giovanni pelo compartilhamento de seus conhecimentos nas disciplinas feitas ao longo do mestrado.

Aos amigos que fizeram disciplinas comigo e compartilharam muitas informações importantes para minha formação.

Aos amigos do Centauro que disponibilizaram um bom laboratório para meus estudos.

Aos amigos do Campus Limoeiro do Norte do IFCE (professores, alunos e servidores) por ter me auxiliado a fazer esse mestrado.

Às empresas Infitech, Ímpar Automação, Nexus Automação, IFCE e SENAI pela disponibilidade de ceder um tempo de seus profissionais para minha pesquisa.

Aos amigos da UDV por todo o bem que continuam fazendo a minha família.

“Todo aquele que finaliza o bom trabalho
que iniciou, recomeça fortalecido.”
(Érick Ribeiro)

RESUMO

Os sistemas supervisórios estão cada vez mais presentes no cotidiano da indústria, pois a garantia de se ter informação sobre os processos de produção em diversos locais simultaneamente é essencial para um bom monitoramento e controle. Contudo, a maioria dos desenvolvedores destas ferramentas não utilizam modelos de desenvolvimento, tampouco métodos de avaliação da qualidade de *softwares*, tornando os projetos mais dispendiosos, devido ao retrabalho, e causando insatisfação aos clientes devido ao não cumprimento aos requisitos fundamentais. Portanto, existe uma necessidade de organização do processo de desenvolvimento e avaliação de supervisórios. As normas ISO/IEC 9126 e 14598 são utilizadas na metodologia proposta, pois fornecem um bom suporte para a identificação das características essenciais de um supervisório, para a criação de um padrão de qualidade e para o desenvolvimento de uma metodologia de avaliação. Este trabalho apresenta uma proposta de modelo para o desenvolvimento de *softwares* supervisórios, acompanhado por um método de avaliação da qualidade baseado em lógica *fuzzy* para analisar as opiniões subjetivas de especialistas da área a respeito de requisitos de supervisórios. A eficiência do uso do modelo foi verificada em âmbito educacional através de um projeto realizado por estudantes. A avaliação revelou a qualidade de cada requisito e mostrou que o *software* completo atendeu a 69% do padrão de qualidade. Os atributos que estavam com ou sem falhas foram identificados e a avaliação mostra quais erros precisam ser sanados para que o *software* seja entregue ao cliente.

Palavras Chave: Lógica *Fuzzy*, *Softwares* Supervisórios, Qualidade de *software*, Método de avaliação, Metodologia de desenvolvimento, Padrão de Qualidade.

ABSTRACT

The supervisory systems are increasingly present in everyday industry, because the guarantee of having information about production processes in diverse locations simultaneously is essential to good monitoring and control. However, developers these tools do not utilize development models, nor methods for assessing software quality, making projects more expensive, due to rework, and causing client dissatisfaction for not meeting fundamental requirements. Therefore, there is a need to organize the process of development and evaluation for supervision softwares. ISO / IEC 9126 and 14598 are used in methodology, as they provide good support to identify essential characteristics of supervisory software, to create quality standard and to development of evaluation methodology. This research proposes model for development of supervisory software, accompanied by method of quality assessment based on fuzzy logic to analyze subjective opinions of specialists about supervisory requirements. The efficiency to use model was verified in educational field through a project realized by students. The assessment revealed quality of each requirement and showed that the complete software attended 69% of the standard of quality. The attributes that were with or without failures were identified, and the evaluation shows errors that need to be solved for software can be delivered for client.

Keywords: Fuzzy Logic, Supervision Softwares, Software Quality, Assessment Model, Development Model, Quality Standard.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

SCADA	*Supervisory Control and Data Aquisition*
ISO	*International Organization for Standardization*
NBR	Norma Brasileira
ERP	*Enterprise Resource Planning*
SIGE	Sistemas Integrados de Gestão Empresarial
CLP	Controlador Lógico Programável
LAN	*Lan Area Network*
IHM	Interface Humano-Máquina
CEP	Controle Estatístico de Processos
TCP-IP	*Transmission Control Process – Industrial Protocol*
OPC	*OLE Process Control*
JPEG	*Join Photograph Expert Group*
BMP	*Bitmap Picture*
PNG	*Portable Network Graphics*
ODBC	*Open Data Base Connectivity*
ABNT	Associação Brasileira de Normas Técnicas
IEEE	*Institute of Electrical and Electronic Engineers*
QDPE	Questionário de delimitação do perfil do especialista
QPCQS	Questionário do grau de presença de características de qualidade de um supervisório
QRCQS	Questionário do grau de relevância de características de qualidade de um supervisório
IFCE	Instituto Federal de Educação, Ciência e Tecnologia do Ceará

CR	Concordância Relativa
GCR	Grau de Concordância Relativa
CCE	Coeficiente de Consenso do Especialista
PE	Peso do Especialista
OPIN	Opinião por Atributo
PAPO	Pasta de Acompanhamento de Projeto e Operação
ORAR	Observar, Registrar, Analisar e Reportar
ISAM	Tipo de Conexão
ISA	*Instrument Society of American*
PQ	Padrão de Qualidade

SUMÁRIO

1. INTRODUÇÃO

Nas últimas décadas, os sistemas automatizados ganharam a preferência nas aplicações industriais por apresentarem vantagens como o aumento da velocidade e precisão no controle dos processos, o rápido monitoramento das variáveis, a ágil detecção de falhas, entre outras. A definição de automação segundo Groover (2010) é o procedimento ou processo, sem assistência humana, realizado com a utilização de um sistema de controle de instruções combinado com um programa de instruções para a consecução de um objetivo específico.

Automação Industrial é o conjunto das técnicas e dos sistemas fabris de produção baseado em máquinas com capacidade de executar tarefas previamente planejadas pelo homem e de controlar sequências de operações sem a intervenção do mesmo. Entre os dispositivos eletro-eletrônicos podem-se utilizar computadores, supervisórios, controladores, entre outros, substituindo algumas tarefas e realizando outras que o ser humano não consegue realizar.

O comportamento de um sistema de automação é semelhante à atitude de um operador humano, que pensa e executa a ação apropriada com base em informações de sensores. O controle automático de processos nesses sistemas possui a capacidade de integrar tecnologias diferentes permitindo o aparecimento de funções como o monitoramento, alarme, intertravamento, comparação de variáveis, operação lógica, armazenamento em memória, etc.

A característica essencial mais marcante das atividades industriais de hoje é a necessidade que as empresas têm de se adaptarem às rápidas mudanças que ocorrem na tecnologia de automação de processos para que possam se manter em um mercado extremamente competitivo. Do ponto de vista de supervisão e do controle de processos, as principais mudanças apontam para a utilização de sistemas integrados, chamados de sistemas SCADA (*Supervisory Control and Data Acquisition*), os quais relacionam todos os níveis de automação, disponibilizando as principais informações da planta industrial num ambiente chamado de supervisório.

O ambiente de modelagem conhecido como *software* supervisório, indispensável para o alcance do nível de complexidade dos sistemas de automação industrial modernos, proporciona o avanço de muitas empresas, e diminui ou até anula a grande dificuldade de se acompanhar a dinâmica do processo. Segundo Moraes e Castrucci (2007), esse *software* é

destinado à supervisão, ao controle e possibilita a realização do monitoramento das variáveis através da visualização de uma interface com telas gráficas elaboradas para qualquer processo industrial ou comercial, independente do tamanho da planta.

O desenvolvimento de aplicações com sistemas supervisórios têm sido objeto de inúmeros estudos, entre eles a abordagem ontológica para modelagem da Interface Humano-Máquina (IHM) de subestações elétricas (TORRES FILHO e VIEIRA, 2012), arquitetura de *hardware* e *software* para supervisão e controle de um carro autônomo (ARRUDA e PEREIRA, 2012), sistemas supervisório para poços de petróleo baseados no método de elevação artificial Plunger Lift (SOARES, 2010), a implementação de um módulo de supervisão para um sistema de detecção de vazamentos em dutos de petróleo (SILVA, 2009)

Outros estudos também podem ser citados, como a síntese e implementação de controle supervisório em uma célula flexível de manufatura didática (CURZEL, 2008), o desenvolvimento de um supervisório modular para uma célula flexível de manufatura (SANTOS, 2007b), a criação de um *framework* para construção de sistemas supervisórios em dispositivos móveis (PEROZZO, 2007), entre outros.

Os supervisórios possuem funcionalidades semelhantes aos *softwares* em geral, portanto os conceitos de Engenharia de *Software* podem ser aplicados a esses aplicativos. Conceitos como processos de desenvolvimento de *software*, qualidade de *software*, requisitos de *software* são importantes para o desenvolvimento de supervisórios.

O processo de desenvolvimento de um *software*, como por exemplo, um supervisório, é composto de um conjunto de atividades e resultados associados que levam à produção de um produto de *software* complexo, que como todos os processos intelectuais, dependem do julgamento humano para a determinação da qualidade requerida. Entretanto, existem atividades fundamentais comuns a todos os processos de *software*, como definir as funcionalidades e as especificações do *software* e atender às necessidades mutáveis dos clientes através da evolução do *software* (SOMMERVILLE, 2011).

Algumas definições precisam ser consideradas, entre elas, a definição de qualidade, amplamente aplicável na atualidade, que, segundo Rodrigues (2006), é o grau no qual um conjunto de características inerentes satisfaz a requisitos pré-estabelecidos ou necessidades implícitas. Qualidade é definida pela norma ISO 9000-3 (1990) como a totalidade das características de um produto propriamente dito que lhe confere a capacidade de satisfazer as necessidades explícitas, que são os objetivos propostos pelo produtor, e as

implícitas, que são condições subjetivas como o tempo de evolução e as necessidades de usuários diferentes.

A qualidade dos sistemas supervisórios desenvolvidos é o segredo da eficiência no monitoramento das variáveis e no controle de um processo, assim como é essencial para adequar o *software* às especificações do projeto. A qualidade pode ser atingida principalmente a partir da verificação, envolvendo especialistas, da existência de características que, combinadas, proporcionam o desenvolvimento de um supervisório com qualidade. O grau de importância dessas características essenciais pode ser determinado através de uma avaliação, por questionários, da relevância de atributos fundamentais.

O alcance da qualidade do supervisório está diretamente relacionado com a metodologia de desenvolvimento. Logo, o desenvolvimento de um supervisório baseado em um modelo que satisfaça os requisitos do *software* pode alcançar os objetivos esperados. Com vistas a saber se as características do supervisório estão satisfeitas, é aconselhável a análise, realizada por especialistas, do grau de presença dos atributos essenciais do aplicativo.

Os programadores de supervisórios estão preocupados em atender as necessidades dos clientes e precisam deste mecanismo de verificação da presença dos requisitos básicos de um *software*. A avaliação de *softwares* geralmente possui um caráter bem subjetivo, em que os avaliadores utilizam aspectos de representação imprecisa que são analisados com eficiência através da utilização da lógica *fuzzy* como arcabouço teórico, servindo de base para a caracterização, interpretação, tratamento e análise dos dados coletados.

1.1 Justificativas da Pesquisa

O processo de desenvolvimento de *software* com qualidade deixou de ser baseado somente nas experiências dos programadores ou na intuição dos mesmos, passando a ser estudado por inúmeros pesquisadores, que contribuíram com a tentativa de construir modelos que orientam o gerenciamento das fases de produção (MECENAS e OLIVEIRA, 2005).

O desenvolvimento dos *softwares* supervisórios já está incluso na maioria dos projetos de automação que visam o monitoramento e o controle de plantas industriais, portanto, pode-se considerar sua presença nas fases de projeto. Contudo, a necessidade de avaliações destes aplicativos ainda exige que sejam coletados características e atributos básicos de um supervisório com qualidade, que, combinados, possam proporcionar aos

programadores um padrão de qualidade a ser seguido. Esses métodos e modelos contribuem para a tomada de decisões corretas no processo de melhoria contínua.

A maioria dos programadores de *softwares* supervisórios que trabalham na indústria utiliza a experiência própria na área para desenvolver as aplicações necessárias para a supervisão das plantas industriais, pois não seguem as metodologias de desenvolvimento de *software* e não possuem disponível um método específico de elaboração de supervisórios que possua critérios e atributos de qualidade fundamentais que satisfaçam as especificações dos projetos de automação.

Um questionário de avaliação do cenário atual de desenvolvimento de supervisórios foi aplicado com alguns especialistas para justificar a proposição dos modelos desta pesquisa. As respostas dos especialistas atestam exatamente a não observância das metodologias de desenvolvimento de *softwares* e a ausência de métodos de organização de variáveis. Os resultados da aplicação deste questionário estão no apêndice D.

A criação de modelos para avaliação da qualidade de *software* é objeto de inúmeros estudos como o desenvolvimento de um modelo *fuzzy* para avaliação da qualidade de *software* (BELCHIOR, 1997), a qualidade de *software*: Teoria e prática (ROCHA, 2001), o desenvolvimento de uma ferramenta *fuzzy* para avaliação da qualidade de *software* (OLIVEIRA, 2002), desenvolvimento de uma metodologia para avaliação de usabilidade de sistemas utilizando a lógica *fuzzy* baseado na ISO (SANTOS, 2007a), o uso da teoria dos conjuntos *fuzzy* para análise do risco de desenvolvimento de *software* (PURIFICAÇÃO, 2008), a criação de um modelo *fuzzy* para avaliação da qualidade de produtos de *software* e da satisfação dos gerentes de projetos numa Fundação Pública Estadual (BOENTE, 2009), entre outros.

Por se tratar de um *software*, é natural que os documentos normativos que estabelecem as diretrizes básicas para as fases do ciclo de vida e para a avaliação da qualidade de *software* também sirvam de apoio para a criação de modelos que permitam analisar o quanto as características e os elementos fundamentais de um supervisório atendem às especificações de projetos de automação. A norma ISO/IEC 9126, que contempla as características básicas de qualidade e as principais métricas internas e externas, e a norma ISO/IEC 14598, que regulamenta as formas de avaliação da qualidade de *software*, fornecem o arcabouço teórico imprescindível para a proposição do modelo sugerido.

Percebe-se que as normas proporcionaram os padrões necessários para a garantia da qualidade de *softwares*. Logo, o cenário de desenvolvimento de sistemas supervisórios

precisa ser modificado com a criação de padrões para a melhoria da qualidade desses *softwares*.

Nesse contexto, a utilização do embasamento teórico extraído das normas de qualidade de *software* como a ISO/IEC 9126 e a ISO/IEC 14598 alinhado aos sistemas de avaliação *fuzzy* e combinado com os métodos já existentes de avaliação de *software* podem permitir a proposição de uma metodologia de avaliação da qualidade de sistemas supervisórios que possa se ajustar à realidade dos diversos tipos de desenvolvedores. Através da opinião de especialistas pretende-se também identificar critérios que sirvam como guia para o desenvolvimento desses aplicativos de supervisão.

A proposição de um método de avaliação envolve diversas fases de grande importância como a definição, através da opinião de especialistas, das características fundamentais; identificação das métricas necessárias, determinação de um padrão de qualidade através da identificação do grau de relevância de cada atributo, determinação da forma de tratamento dos dados, análise dos resultados e tomada de decisão.

Todas estas etapas formam um sistema de avaliação completo e o desenvolvimento desta pesquisa tem uma boa contribuição para a academia por propor um mecanismo de avaliação da qualidade de supervisórios através da utilização das bases teóricas de normas e da teoria *fuzzy* em conjunto com a subjetividade da opinião de especialistas.

Este estudo também contribui para o mercado, pois fica clara a necessidade de identificação de atributos fundamentais na construção dos aplicativos de supervisão. Portanto, os programadores são beneficiados com um modelo que pode evitar os principais motivos de descontentamento dos clientes como não satisfazer aos requisitos de *software*. Este trabalho torna possível também a avaliação de supervisórios desenvolvidos e em desenvolvimento através do consenso de diversos especialistas a respeito da qualidade do *software*.

1.2 Formulação dos Problemas da Pesquisa

Como definir um padrão de qualidade através da identificação de quais são os atributos e as características fundamentais de um *software* de supervisão e desenvolver uma metodologia de avaliação da qualidade de um sistema supervisório?

Como elaborar um modelo de desenvolvimento de supervisórios com os atributos considerados fundamentais para a qualidade do aplicativo?

1.3 Objetivos da Pesquisa

1.3.1 Objetivo Principal

Este trabalho é proposto com um objetivo de elaboração de um modelo de desenvolvimento de supervisórios e de uma metodologia de avaliação da qualidade desses aplicativos através da identificação dos atributos fundamentais, a partir das normas ISO/IEC 9126 e ISO/IEC 14598, e da utilização da lógica *fuzzy*.

1.3.2 Objetivos Específicos

- Elaborar questionários que coletem as opiniões subjetivas de especialistas na área de automação;
- Identificar os atributos fundamentais de um supervisório, assim como seus graus de relevância na análise da qualidade;
- Determinar um padrão de qualidade para cada atributo do supervisório;
- Elaborar um modelo de desenvolvimento de supervisórios com os atributos considerados fundamentais para a qualidade do aplicativo;
- Desenvolver um método de avaliação de supervisórios à luz das normas ISO/IEC 9126 e ISO/IEC 14598;
- Avaliar um supervisório desenvolvido numa instituição de ensino através da metodologia proposta.

1.4 Delimitações da Pesquisa

A pesquisa foi realizada através da coleta das opiniões de especialistas que trabalham em instituições de ensino ou em empresas prestadoras de serviço na área de automação no estado do Ceará. Os especialistas foram consultados para indicarem quais são os principais atributos da qualidade de um supervisório, qual é o grau de relevância deles em qualquer supervisório e qual é o grau de presença deles em um aplicativo já desenvolvido.

Segundo Rosário (2005), a indústria brasileira está precisando ajustar-se para competir internacionalmente principalmente em relação à melhoria dos recursos humanos, que é um dos principais obstáculos para a modernização de métodos e equipamentos do

parque industrial brasileiro. Diante disso, já existe um consenso quanto à necessidade de capacitação de profissionais para atuarem nos setores de automação industrial.

Para atender a esta necessidade, a pesquisa limitou-se a analisar sistemas supervisórios desenvolvidos por alunos da disciplina de Sistemas de supervisão que cursam o 7º semestre do curso superior de Mecatrônica industrial do IFCE, com base em projetos educacionais de automação de plantas industriais.

Esta escolha se deu porque estes profissionais serão futuros programadores e necessitam de ferramentas modernas de desenvolvimento e de avaliação de supervisórios desenvolvidos.

Além disso, a alternativa de análise de sistemas supervisórios elaborados em uma dessas empresas por especialistas de outras empresas do Ceará revelaria alguns segredos específicos de desenvolvimento de cada programador que são diferenciais nesse mercado competitivo e não devem ser expostos publicamente para garantir a ética profissional.

A primeira fase da consulta aos profissionais destinou-se a identificar o grau de experiência dos especialistas na área abordada, coletar a opinião deles a respeito de quais atributos são importantes na avaliação da qualidade dos supervisórios, bem como delinear o grau de relevância deles. Na primeira fase, os atributos essenciais e o padrão de qualidade foram definidos e, na segunda fase da pesquisa, o aplicativo de supervisão desenvolvido foi analisado pelos especialistas através dos questionários de medição do grau de presença dos atributos no sistema supervisório.

1.5 Organização da Pesquisa

A pesquisa proposta está organizada da seguinte forma:

O capítulo 1 inicia a discussão a respeito da importância do desenvolvimento de sistemas supervisórios para o monitoramento e controle das plantas industriais, assim como, retrata a necessidade de uma avaliação da qualidade desses aplicativos baseada no arcabouço teórico das normas de qualidade de *software* combinadas com a possibilidade de tratamento de informações subjetivas através da lógica *fuzzy*.

O capítulo 2 introduz os conceitos de sistemas supervisórios e sistemas SCADA com suas respectivas características, além de referenciar as normas e os modelos existentes de avaliação da qualidade de *software* e seus respectivos atributos e métricas. Ainda neste

capítulo são apresentados os fundamentos da teoria *fuzzy* com suas operações e variáveis linguísticas.

No capítulo 3 é apresentado um modelo de desenvolvimento de supervisórios composto pelas fases de elaboração do aplicativo baseado nos modelos de desenvolvimento de *software* já existentes.

No capítulo 4, o método de avaliação da qualidade de sistemas supervisórios é descrito com todas as suas características e sequências de tratamento de dados.

No capítulo 5, o supervisório desenvolvido é avaliado e os resultados são apresentados e discutidos de acordo com a metodologia proposta.

No capítulo 6, as conclusões do trabalho são apresentadas. Os trabalhos futuros também são expostos neste capítulo.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Sistemas SCADA

Sistemas de supervisão e controle industrial, também conhecidos no ambiente industrial como sistemas SCADA, permitem o monitoramento em tempo real dos sinais representativos de medidas e do estado atual do processo industrial, recolhendo estes dados em ambientes complexos e apresentando de modo amigável para o operador através de interfaces humano-máquina (ROSÁRIO, 2005).

Sistemas SCADA surgiram a partir da necessidade de integração de vários processos da indústria em um sistema único, que apresentasse aos operadores, de forma centralizada e em tempo real, a quantidade máxima de informações, armazenando-as em memória, possibilitando o melhoramento da eficiência de controle e monitoramento, e permitindo a geração de gráficos e relatórios que servem de apoio à tomada mais rápida de decisões.

Conforme Aihara et al. (2001), os sistemas de supervisão, responsáveis pelo controle e monitoramento de variáveis, tem como principal objetivo o fornecimento de elementos fundamentais para o operador monitorar e/ou controlar mais rapidamente um processo de automação, gerenciando o processo por completo e provendo a possibilidade de leitura das variáveis em tempo real.

Através da evolução das tecnologias computacionais, utilizadas para o desenvolvimento dos sistemas SCADA, houve um aumento da confiabilidade, flexibilidade e conectividade desses sistemas. Desta forma, as informações puderam ser coletadas por equipamentos de aquisição de dados, analisadas e disponibilizadas em ambientes afastados geograficamente, com a possibilidade de acesso remoto simultâneo de diversos operadores e administradores do processo.

A visualização do processo é realizada através de uma interface gráfica sofisticada com ferramentas que simulam a dinâmica do sistema e com recursos como o armazenamento em banco de dados, a geração de alarmes com ordem de prioridade, a criação de mecanismos de tolerância a falhas e interações entre *softwares* de diferentes fabricantes.

Segundo Vianna (2008), os sistemas SCADA proporcionam algumas vantagens descritas a seguir:

- Redução de gastos com montagem de painéis de controle e projeto;

- Redução de custos da aquisição de instrumentos de painel, pois no sistema SCADA eles são virtuais;
- Eliminação de custos com peças de reposição, pois se trata de instrumentos virtuais;
- Redução de espaço necessário para a sala de controle;
- Dados disponíveis em formato eletrônico, facilitando a geração de relatórios e a integração com sistemas ERP (*Enterprise Resource Planning*) ou Sistemas Integrados de Gestão Empresarial (SIGE);
- Praticidade de operação, pois os instrumentos são apresentados ao operador em um simples clique do dispositivo apontador;

A arquitetura de um sistema SCADA, ilustrada na figura 1 é um exemplo dado por Seixas (2004), composto por três níveis: a rede de informação é composta pela rede de computadores, é responsável pela interface entre o sistema e os operadores e pela comunicação entre as estações que hospedam o supervisório, os bancos de dados, a rede corporativa e outros *softwares* distintos; a rede de controle é composta pelos Controladores Lógicos Programáveis (CLPs) e é responsável pelo controle do processo físico; a rede de campo é composta pelos sensores e atuadores e é responsável pela aquisição dos dados. A rede de comunicação integra estes três níveis através dos mecanismos de comunicação que transportam a informação do nível mais baixo para os níveis superiores da hierarquia.

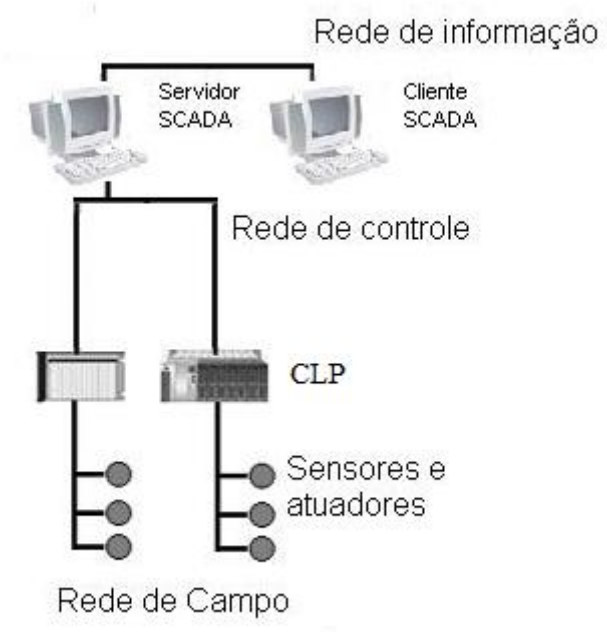


Figura 1 - Exemplo de arquitetura de um sistema SCADA

Fonte: Adaptado de Seixas, 2004

As necessidades do sistema, tais como, o acesso de muitos usuários ao mesmo sistema, o compartilhamento de dados e recursos, a maior confiabilidade no sistema, a consistência da informação e a redução de custos levou ao surgimento de outras arquiteturas de sistemas SCADA.

Outro tipo de arquitetura de sistemas SCADA, proposta por Souza (2005) e ilustrada na figura 2, é formada pela rede de campo, responsável pela aquisição, local ou remota, de dados do processo, e pela rede local de supervisão, que compartilha esses dados em uma LAN (Lan Area Network) a fim de que os usuários possam requisitá-los para realização do controle e supervisão do processo.

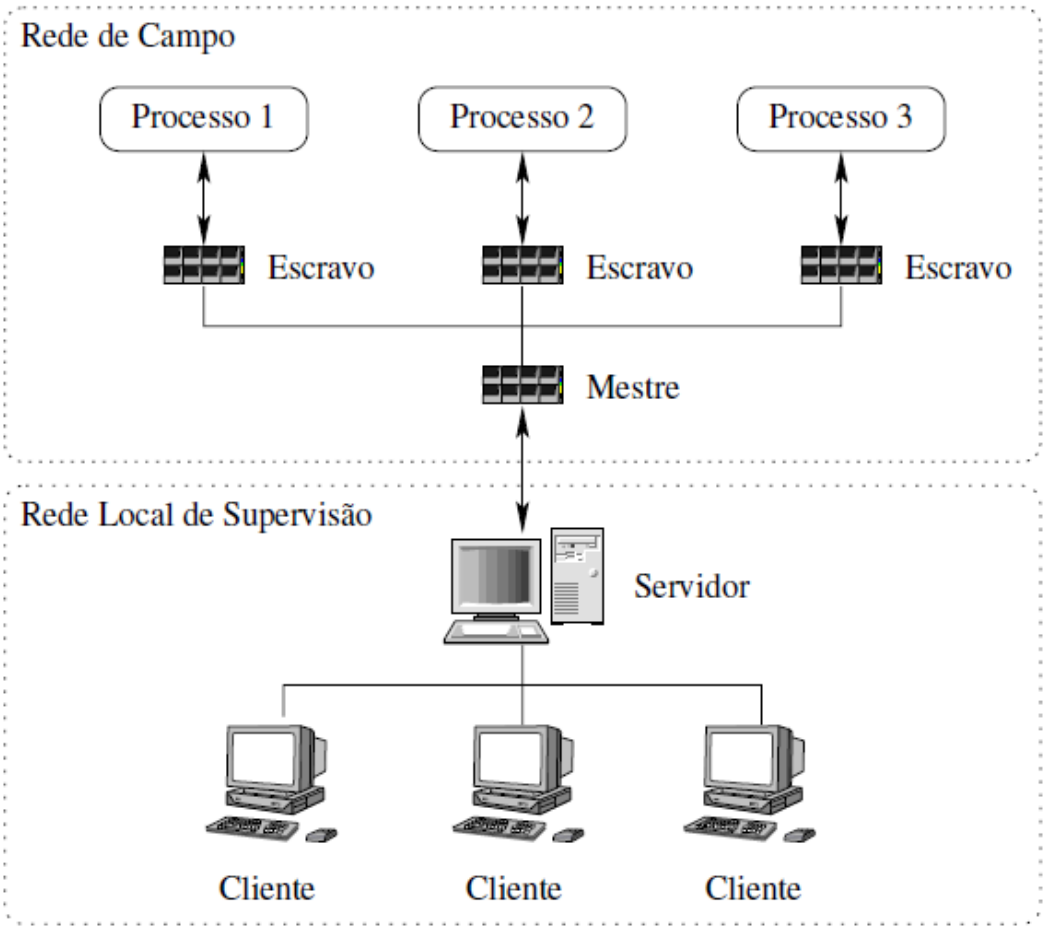


Figura 2 - Arquitetura de sistemas SCADA

Fonte: SOUZA, 2005

Segundo Souza (2005), a maioria das redes de campo utiliza o mecanismo de comunicação mestre/escravo com o intuito de obter uma comunicação determinística, onde os escravos respondem somente às solicitações feitas pela estação central mestre, que inicia uma transação, podendo endereçar uma mensagem para um único escravo ou para vários escravos

ao mesmo tempo. Já a rede local de supervisão geralmente utiliza o mecanismo de comunicação cliente/servidor, em que clientes distribuídos numa rede realizam a requisição de dados a uma unidade servidora, que coleta as informações requisitadas e as repassa ao servidor.

Os sistemas SCADA visam, principalmente, a produtividade, qualidade e segurança em um processo. Em um sistema típico, toda a informação dos sensores é concentrada em um controlador lógico programável, o qual, de acordo com o programa em memória, define o estado dos atuadores. Atualmente, com o advento da instrumentação de campo inteligente, funções executadas no controlador programável tem uma tendência de serem migradas para instrumentos de campo (SOUZA, 2005).

Uma contribuição importante dos sistemas SCADA é a conexão do sistema de supervisão e controle com sistemas corporativos de administração das empresas. Esta conectividade permite o compartilhamento de dados importantes da operação diária dos processos, contribuindo para uma maior agilidade na determinação de soluções e maior confiabilidade dos dados que suportam as decisões dentro da empresa. Segundo Pinto (2005), há, inclusive, a possibilidade de integrar outros níveis da instituição tais como o sistema de alimentação elétrica e os sistemas de segurança.

Os sistemas SCADA estão em uso em muitas instalações industriais, residenciais e empresariais, como sistemas de manufatura, geração de energia, refino de petróleo, distribuição de água, tratamento de esgoto, distribuição de energia elétrica, edifícios, proporcionando soluções para as mais diversas necessidades das indústrias do mundo todo. Dentre outras coisas, os benefícios das soluções de integração de tecnologias proporcionam uma grande economia de operações manuais que, por sua vez, produzem uma grande melhoria na qualidade dos produtos. A integração das salas de controle centralizadas com a instrumentação de campo, bem como aquela existente entre diversos sistemas de controle em uma vasta área da indústria são exemplos dessas soluções.

A utilização de centros de controle, constituídos de monitores e painéis de vídeo mostram ao operador as informações atualizadas do sistema, armazenando-as em banco de dados. Esses sistemas também possibilitam o comando remoto das diversas operações existentes pelo servidor e a visualização do processo pelo cliente, através do *software* supervisório, que é, na maioria das aplicações, o mestre da rede industrial. A utilização, qualidade e eficiência desses sistemas estão motivando os futuros profissionais a se

qualificarem nessa área através do estudo das ferramentas dos *softwares* supervisórios e sua integração com os controladores distribuídos na rede industrial.

2.1.1 Interface Humano-Máquina

A interface Humano-Máquina (IHM), também conhecida como interface de usuário, é o elemento de interação que proporciona aos operadores a possibilidade de visualização, manipulação, controle e supervisão em tempo real das variáveis e dos eventos do processo, permitindo a geração de gráficos, relatórios e o armazenamento de dados em históricos.

Quando as IHMs foram concebidas, eram plataformas proprietárias, que possuíam limitações de comunicação entre equipamentos de fabricantes diferentes. Na atualidade, estas interfaces são baseadas na plataforma PC, que permite comunicação com diversos fabricantes, garantindo comunicação com a rede corporativa, bem como provendo funcionalidades para controle estatístico de processos (CEP) e impressão de relatórios (VIANNA, 2008).

As IHMs podem ser estruturas de *hardware* como *displays*, coloridos ou bicolores, encapsulados em caixas metálicas com botões, botoeiras, teclado, chaves, portas de comunicação, podendo ou não possuir tela *touch screen* e placa eletrônica. Elas podem ser implantadas com terminais de dados para versões industriais, apropriadas para o chão de fábrica, com facilidades de comunicação com o CLP e do tipo *plug-and-play*, porém não muito amigáveis (RIBEIRO, 1999).

Podem também ser estruturas de *software* como os supervisórios, que utilizam o monitor do computador no lugar de *displays* e possuem telas interativas, chaves lógicas ou virtuais ao invés de botões, chaves ou botoeiras, anunciador de alarmes, recursos manipulados facilmente pelo usuário, e são compostos por pacotes de *software* com *drivers* para comunicação com controladores, banco de dados, computadores em rede, impressoras, internet, *softwares* diversos e servidores OPC (OLE *Process Control*).

Segundo Ribeiro (1999), a IHM permite que o operador veja, como se fosse um painel semi-gráfico, o diagrama esquemático do processo contendo os equipamentos, instrumentos e ligações. É importante enfatizar que podem ser mostradas as malhas em estado de alarme nas telas das IHMs inseridas nas estações de operação. Os painéis são exemplificados na figura 3 e estão localizados na estação de trabalho.

Figura 3 - Estação de operação com as IHMs
Fonte: www.macropainel.com.br

A apresentação do processo nas telas de animação modernas é realizada por meio de sinópticos, que são esquemas ou *layouts* de representação da dinâmica do sistema, podendo ser configurados e programados, e possibilitando, ainda, a importação de figuras nos formatos JPEG, BMP, PNG, entre outros. A hierarquização de telas é possível e, inclusive, necessária quando o próprio processo é composto de multiníveis de produção.

Segundo Vianna (2008), as IHMs também podem ser customizadas e aplicadas em ambientes perigosos como aqueles que, devido à presença de gases suspensos, possuem risco de explosão, e em ambientes que exigem que o equipamento tenha proteção contra partículas de poeira.

Dentre as alternativas, o monitor do computador, na maioria das aplicações atuais, substitui a IHM composta por *display* e teclado, devido a uma melhor relação de custo-benefício desta solução. Logo, os *softwares* de supervisão são os mais empregados na indústria de um modo geral e esta pesquisa apresenta suas principais características, funcionalidades, ferramentas e tipos, considerando os atributos fundamentais presentes na maioria das soluções.

2.1.2 *Softwares* Supervisórios

Em vista da necessidade de automação dos vários processos existentes em uma indústria com a finalidade de poder centralizar o máximo de informações no menor tempo possível e integrar diversos dispositivos que pertencem ao processo, foram desenvolvidos *softwares* para adquirir dados e fornecer informações ao operador do processo, reduzindo as dimensões dos painéis que existiam e melhorando a interface humano/máquina (SILVA, 2009).

A exigência por aplicativos de supervisão utilizados no monitoramento de variáveis de controle de processos industriais, capazes de implementar funções de controle complexas, gerenciar as informações do sistema e guardá-las em banco de dados é cada vez maior, pois as aplicações modernas necessitam de uma alta interatividade e de sistemas abertos que sejam rapidamente atualizados para o uso em diversas aplicações (ROSÁRIO, 2005).

Os *softwares* supervisórios, instalados em servidores centrais e/ou clientes distribuídos, são responsáveis pela obtenção dos dados do processo em tempo real, através da comunicação com os CLPs, e dispõem de uma interface que permite ao usuário a operacionalização, inspeção e supervisão de uma planta industrial ou instalação física em operação, tornando o processamento das variáveis de campo mais rápido e eficiente.

As telas de supervisão, *drivers* de comunicação, gráficos informativos, alarmes eventuais, históricos de dados, relatórios semanais e variáveis auxiliares desses *softwares* são totalmente configuráveis pelo programador (desenvolvedor) através de uma plataforma de desenvolvimento de aplicações de supervisão, em que esses itens aparecem como objetos.

Assim, através da configuração do *software*, o operador (funcionário que fará a operação em campo na empresa a qual o supervisório será entregue) pode selecionar telas que apresentam os valores numéricos das variáveis de processo de diferentes modos, à sua escolha. Os valores podem aparecer ao lado dos equipamentos associados. Por exemplo, o nível do tanque pode ser apresentado em porcentagem ao lado do desenho do tanque, a vazão que passa por uma tubulação pode ter o valor instantâneo mostrado junto da tubulação, a temperatura de um reator pode ser mostrada em diferentes posições, em valores digitais. Através das configurações de tela, os instrumentos virtuais podem se parecer com instrumentos convencionais, com escala analógica (gráfico de barras simula a escala analógica), com botões, chaves seletoras e chaves de atuação (RIBEIRO, 1999).

Segundo Silva (2009), um *software* supervisório deve possuir algumas funções básicas, como:

- Modificação dos parâmetros operacionais;
- Sinalização visual e sonora de alarmes;
- Gráficos de tendência das variáveis de processo;
- Alarmes com descrição e codificação de cores;
- Armazenamento de eventos;
- Relatórios;
- Apresentação das variáveis em tempo real;
- Telas de acompanhamento operacional;
- Sinalização do *status* de operação dos equipamentos;
- Armazenamento histórico e visualização de variáveis de processo e alarmes.

O armazenamento por longo prazo dos sinais envolvidos em um processo industrial pode se revelar bastante proibitivo em sistemas mais simples, contudo pode ser realizado facilmente com um *software* supervisório. O armazenamento automático de variáveis críticas de controle do processo em disco pode ser recuperado mais tarde para explicar as dificuldades de operação do processo. O computador também pode ser usado para gravar automaticamente características de um alarme como o tipo, a localização, o tempo em que esta ativo, o tempo de reconhecimento, e o tempo que foi aprovada pela intervenção do operador. Esta informação é útil para os supervisores na detecção de violação dos regulamentos de segurança ou no diagnóstico de problemas no processo (COUGHANOWR, 2009).

Os supervisórios identificam todas as variáveis do processo real (ex: temperatura, nível, vazão, etc.) envolvidas na aplicação através da declaração de *tags*, que são manipulados com operações computacionais. Após a estratégia configurada, o *software* básico deve executar, gerenciar e armazenar o resultado de cálculos e operações realizadas para apresentar ao usuário e armazenar no banco de dados.

Esses *softwares* podem identificar quando o valor de um *tag* ultrapassa uma faixa ou condição pré-estabelecida, desde que o programador adicione a função de geração de alarmes. Todos os alarmes podem ser configurados para gerar relatórios, que geralmente são armazenados em registros de bancos de dados. Os eventos imprevistos no processo são rapidamente detectados pelos alarmes, e medidas corretivas são imediatamente

providenciadas pelo operador, que visualiza o processo através das telas interativas e acompanha as alterações ocorridas.

Há uma gama diversificada de ambientes de desenvolvimento de supervisórios disponível atualmente no mercado e estima-se que os mais utilizados no Estado do Ceará são:

- Elipse SCADA da Elipse Software (www.elipse.com.br);
- Elipse E3 da Elipse Software (www.elipse.com.br);
- WinCC da Siemens (www.siemens.com);
- InTouch da Wonderware (www.wonderware.com);
- FactoryTalk da Rockwell Software (www.rockwellautomation.com).

2.1.3 Ambiente de desenvolvimento Elipse SCADA

O ambiente de desenvolvimento de supervisório Elipse SCADA desenvolvido pela empresa Elipse Software foi concebido para ser uma ferramenta ideal para a criação dos aplicativos de supervisão e controle nas mais diversas aplicações e continua em permanente atualização. Sua utilização proporciona ao processo eficiência, competitividade, interatividade, evita a necessidade de soluções demoradas e caras, e está disponível para as plataformas Microsoft Windows 98, ME, NT, 2000, XP, 2003 e Windows CE (ELIPSE, 2012).

As características e funcionalidades do Elipse SCADA para a construção dos supervisórios fornecem os requisitos necessários como telas configuráveis, *tags* de diferentes tipos, *displays*, botões e gráficos customizados, *driver* de comunicação com banco de dados ODBC (*Open Data Base Connectivity*), comunicação com servidores em rede ou com *softwares* que suportam o protocolo OPC (*OLE Process Control*), comunicação com a maioria dos CLPs vendidos no mercado, geração pré-programada de relatórios, ferramentas de visualização e armazenamento em memória de alarmes, programação através da linguagem Elipse Basic, entre outros.

A organização dos aplicativos elaborados no Elipse SCADA é realizada através do *Organizer*, ilustrado à esquerda da figura 4, que é a "árvore da aplicação, onde estão dispostas as ferramentas de um supervisório descritas anteriormente. Todo tipo de acesso às configurações pode ser feito através desta ferramenta de organização, inclusive o desenvolvimento de *scripts* (linhas de programação) associados aos objetos de tela (*display*, *setpoint*, *slider*, *animation*, *gauge*) do aplicativo. O programador pode fazer a configuração de

uma gama de propriedades de cada objeto, por exemplo a edição das características de um *display* através de uma janela de características também ilustrada na figura 4.

O Elipse SCADA possui suporte para comunicação com diversos tipos de CLPs e monitora as variáveis através de *tags* configuráveis pelo programador para cada tipo de dado. Os *tags* são identificados por nomes ou mnemônicos e podem ser do tipo temporizador, contador, auxiliar, matricial e de comunicação com a possibilidade de ser organizados em grupos.

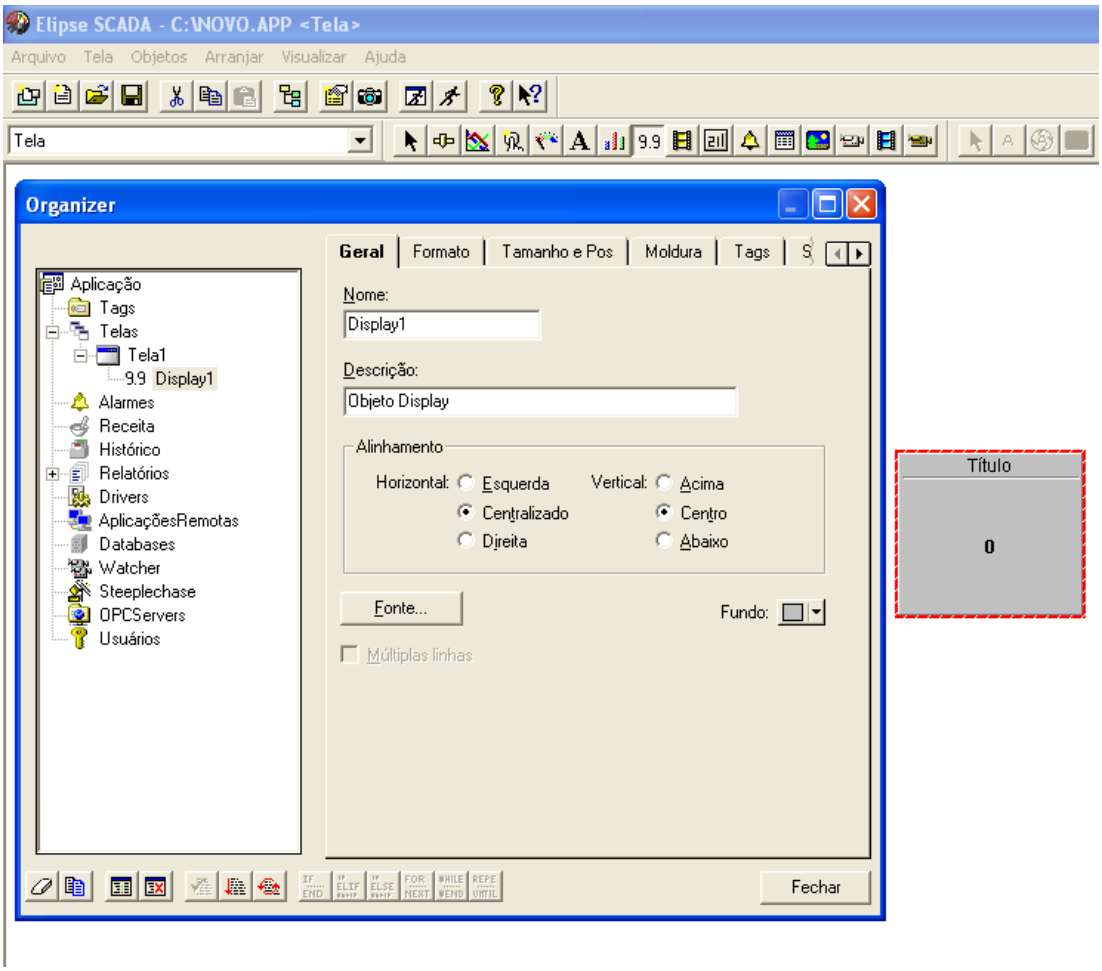


Figura 4 - Interface do Elipse SCADA

O ambiente de desenvolvimento Elipse SCADA dá suporte à construção de um supervisório. Sendo o *software* supervisório um produto de *software* que deve ser avaliado para que se possa garantir a qualidade de requisitos, é necessário aplicar conceitos de qualidade da área de Engenharia de *software*. Portanto, a próxima seção fornece o embasamento teórico para avaliar a qualidade de um *software*.

2.2 Avaliação da Qualidade de *Software*

Segundo Fernandes (2003), o desenvolvimento de *software* proporciona uma relação humana de troca de planos e necessidades entre os que usam, os que produzem e os que adquirem o *software*. O *software* é uma descrição de máquina, um artefato virtual que não é capaz de realizar trabalho desacoplado de uma máquina que interprete e carregue as instruções e informações contidas nele, mas tem a função própria de fornecer uma interface entre o usuário e esta máquina (computador).

Classicamente, um *software* é definido como sendo estruturas de dados que permitem a manipulação adequada de informações relativas ao problema a ser resolvido, ou um conjunto de instruções, que quando executadas, produzem o desempenho e a função desejados. Entretanto o *software* deve ser analisado como um produto a ser vendido e por esta razão deve passar por uma fase de controle de qualidade (MAZZOLA, 2008).

A consolidação da Informática nos últimos anos como ferramenta essencial para o planejamento, organização e supervisão de tarefas gerou a necessidade de desenvolvimento de *softwares* que atendam a requisitos de qualidade pré-estabelecidos. Desta forma, o sucesso das organizações nesse mercado competitivo está diretamente ligado à garantia da qualidade dos serviços ou produtos oferecidos.

A busca pelo alto nível de qualidade de *software* faz parte dos principais objetivos da maioria das empresas, pois, atualmente, os clientes não aceitam produtos com baixa qualidade ou reparos em produtos de *softwares* já entregues com problemas e deficiências. Essa exigência mostra a equiparação dos produtos de *software* com outros produtos manufaturados como geladeiras, veículos e televisões (SOMMERVILLE, 2011).

Conforme a norma ISO 9000-3 (1990), o desenvolvimento e a manutenção de *software* são diferentes da maioria dos outros produtos industriais, consequentemente, o estabelecimento de sistemas de avaliação de qualidade requer orientações adicionais como a identificação dos requisitos fundamentais necessários para a garantia da qualidade de um produto de *software*.

Segundo a norma ISO/IEC 9126 (1998), a correta operação de computadores é frequentemente crítica para a segurança humana e para o sucesso dos negócios em uma variedade de áreas de aplicação. Desta forma, a seleção ou o desenvolvimento de produtos de *softwares* de alta qualidade é de fundamental importância para a celeridade das tarefas realizadas com o computador.

Para que o *software* receba a garantia de qualidade adequada são necessárias a especificação e a avaliação do mesmo, realizadas pela identificação apropriada das características de qualidade relevantes em conjunto com a definição das métricas de avaliação que serão aplicadas.

De acordo com a norma ISO/IEC 14598 (1998), a avaliação da qualidade de *software* tem como propósito apoiar de forma direta o desenvolvimento e a aquisição de *software* para que os clientes e usuários tenham suas necessidades atendidas. A qualidade requerida precisa ser assegurada ao final do processo de forma a atender as necessidades explícitas e implícitas dos usuários, incluindo operadores, destinatários dos resultados do *software*, ou mantenedores de *software*.

2.2.1 Conceito de qualidade de *software*

O planejamento, a organização e a padronização do desenvolvimento surgiram através da necessidade de proporcionar qualidade aos produtos de *software* vendidos no mercado e de facilitar a manutenção dos mesmos por outros programadores. A qualidade é um fator importantíssimo para o ciclo de desenvolvimento e deve ter seu conceito bem definido para facilitar o entendimento do que vem a ser um *software* de qualidade.

O *software* deve oferecer ao usuário um bom desempenho funcional, todas as características requisitadas, a confiabilidade, a legibilidade, a documentação completa, a facilidade de uso e de manutenção. Essas propriedades, quando estruturadas em metodologias e em conjunto com outras, podem gerar bons resultados para o desenvolvimento segundo os princípios da Engenharia de *Software*.

Segundo Pressman (2006), qualidade de *software* é a adequação a padrões de desenvolvimento claramente documentados, a requisitos funcionais e de desempenho explicitamente declarados e a características implícitas que são inerentes a todo *software* desenvolvido profissionalmente.

Conforme a norma ISO/IEC 9126-1 (1998), um *software* de qualidade atende a requisitos de qualidade interna, qualidade externa e qualidade em uso. A qualidade interna é definida como o conjunto de todas as características do ponto de vista interno como documentos, código-fonte, que são usados para especificar as propriedades dos produtos intermediários. Já a qualidade externa é aquela avaliada e medida, através de métricas externas, quando o *software* é executado e está sendo testado num ambiente simulado com

dados simulados. A qualidade em uso é a visão do usuário, obtida em um ambiente e contexto definido, a respeito da qualidade do produto de *software*.

Os requisitos ou fatores necessários para um *software* ser de qualidade, segundo Mazzola (2008), são os seguintes:

- Correção: capacidade dos produtos de *software* de realizarem suas tarefas com a precisão requerida na especificação do projeto. Este fator possui uma grande importância, pois a elaboração de uma interface humano-máquina bem desenvolvida não compensa a execução de funções de forma incorreta;
- Robustez: capacidade de funcionamento do sistema mesmo em condições adversas. A maior importância da robustez é a garantia de que não ocorrerão situações catastróficas em condições anormais, principalmente porque essas condições serão alarmadas previamente;
- Extensibilidade: é a facilidade de modificação dos produtos de *software*. Este fator mostra o grau de flexibilidade, assim como revela que a simplicidade do projeto contribui efetivamente para a manutenção;
- Compatibilidade: é a facilidade com a qual os produtos de *software* podem se comunicar com outros. Este é um fator essencial para a comunicação, por exemplo, dos supervisórios com os bancos de dados, servidores OPC, etc.;
- Eficiência: está relacionada com a utilização racional dos recursos de *hardware* e de sistema operacional;
- Facilidade de uso: a fácil utilização é um dos parâmetros mais visível para os usuários. A documentação do *software* e a criação de um manual do usuário capaz de orientá-lo permitem que esse fator seja melhor atendido.

A garantia de qualidade de *software* é organizada em atividades, formando uma estrutura que estabelece ou seleciona padrões que devem ser aplicados ao processo de desenvolvimento ou ao produto de *software*. A integração dos padrões em procedimentos ou processos torna a avaliação da qualidade possível. Esses processos são apoiados pelo uso de ferramentas que agregam o conhecimento dos padrões de qualidade (SOMMERVILLE, 2011).

Um plano de garantia da qualidade deve ser seguido para que o processo de desenvolvimento possa ser mais célere e eficiente. A necessidade de adequação dos *softwares* a padrões e normas regulamentados internacionalmente deu origem a proposição de planos ou

modelos de avaliação da qualidade de *software* que servem como guias de procedimentos para os desenvolvedores.

Nesse sentido, muitas pesquisas são desenvolvidas com a intenção de criação de novos modelos e padrões de avaliação da qualidade de *software*, tais como: Modelo de Qualidade Rocha Estendida (BELCHIOR, 1997), Modelo Fuzzy para Avaliação da Qualidade de *Software* (BELCHIOR, 1997), Modelo de Qualidade ISO/IEC 9126-1, Processo de Avaliação da Qualidade da norma ISO/IEC 14598-1, Modelo de avaliação CMMI (CMMI, 2002), Processo de Ciclo de Vida de *Software* da norma ISO/IEC 12207, Processo de Desenvolvimento de *Software* da norma ISO/IEC 15504, entre outros.

2.2.2 Modelos de qualidade de *software*

A existência de requisitos fundamentais e características gerais em todos os produtos de *software* tornam possível a elaboração das normas, padrões e modelos, que possuem a finalidade de realizar o controle de qualidade de *softwares* distintos, através da escolha e utilização das métricas corretas.

O principal objetivo da criação desses modelos para o desenvolvimento de *software* é prover ferramentas que possam apoiar as empresas nos processos de desenvolvimento, garantia da qualidade e manutenção de seus produtos e serviços, com a finalidade de se manterem no mercado competitivo.

Os modelos também são recomendados a qualquer interessado em iniciar o ciclo de melhoria de seus processos em um ambiente de desenvolvimento de interfaces entre computador e usuário, utilizando um método de avaliação com a finalidade de identificar o grau de eficiência do desenvolvimento de *softwares* através de resultados consistentes.

Segundo Sommerville (2011), os modelos e padrões de *software* são importantes pelos seguintes motivos:

- Eles englobam as melhores práticas ou, pelo menos, as mais adequadas. Essas práticas são advindas de conhecimentos frequentemente adquiridos após muitas tentativas e erros. Os erros cometidos anteriormente à criação dos padrões são evitados, pois as normas registram a sabedoria obtida, proporcionando facilidades para a organização;
- Eles mapeiam os requisitos necessários aos usuários, estruturando as fases de construção de *software* e detectando as características faltantes para o desenvolvimento ideal;

- Eles provêm uma infraestrutura essencial para que o processo de garantia da qualidade de *software* possa ser implementado e avaliado. Considerando a alta relevância das características desses padrões, o controle da qualidade só precisa dar a garantia que os padrões foram corretamente seguidos;
- Eles proporcionam maiores facilidades na manutenção do *software*, quando o trabalho realizado por uma pessoa é assumido e continuado por outra pessoa. A padronização minimiza o esforço de aprendizado quando se inicia um novo trabalho, pois os engenheiros dentro de uma organização adotam as mesmas práticas.

2.2.3 Modelo de Qualidade ISO/IEC 9126

Segundo ABNT (1999), a definição de normalização é o processo de executar atividades de forma ordenada e aplicar regras estabelecidas. O desenvolvimento e avaliação de *softwares* apoiado pelas normas provêm benefícios como facilidade de uso, redução de custos, tempo e erros.

A norma ISO/IEC 9126 (1998), de título Engenharia de *Software* – Qualidade de Produto tem o objetivo de orientar os desenvolvedores de *software* na elaboração e avaliação dos seus produtos. A norma está dividida em quatro partes:

- ISO/IEC 9126-1: Modelo de qualidade;
- ISO/IEC 9126-2: Métricas Externas;
- ISO/IEC 9126-3: Métricas Internas;
- ISO/IEC 9126-4: Métricas de Qualidade em Uso;

A norma ISO/IEC 9126 (1998), de título Modelo de qualidade, descreve um modelo de qualidade de *software* que serve como um guia no estabelecimento de características ou atributos de qualidade organizados numa hierarquia. A especificação de requisitos funcionais e não funcionais do cliente e do usuário também é realizada através das características presentes na norma.

A norma ISO/IEC 9126-1 pode auferir proveito para os envolvidos com desenvolvimento, apoio, aquisição, uso, manutenção, requisitos, avaliação, garantia da qualidade e auditoria de *software*. Pode-se citar a utilização da norma por desenvolvedores, adquirentes, responsáveis por garantia da qualidade e avaliadores independentes.

O modelo de qualidade ISO/IEC 9126 (1998) pode ser utilizado para:

- Identificar requisitos de *software*;
- Definir critérios para garantia da qualidade;
- Validar a completitude de uma definição de requisitos;
- Identificar objetivos de projeto de *software*;
- Definir critérios de aceitação para produtos finais de *software*;
- Identificar objetivos para teste de *software*.

A norma ISO/IEC 9126 (1998) em conjunto com outras normas pode fornecer:

- Apoio, no processo de gestão, para o estabelecimento de objetivos de qualidade organizacionais;
- Apoio para validação, revisão e verificação nos processos de apoio do ciclo de vida de *software*;
- Apoio para identificação de metas de qualidade;
- Uma estrutura para avaliação quantitativa de qualidade, no processo de apoio;
- Uma estrutura para definição de qualidade do produto de *software*, no processo cliente-fornecedor.

O modelo de qualidade da norma não garante a obtenção da "qualidade total do *software*", entretanto compõe-se de uma estrutura que proporciona a qualidade suficiente para atender aos requisitos específicos de cada projeto. As características definidas na norma podem ser utilizadas em *softwares* de todos os tipos, inclusive os que são empregados como uma interface humano-máquina. Diante disso, os princípios da Engenharia de *Software* podem ser utilizados para o desenvolvimento de supervisórios e é nesse cenário que a proposta presente neste trabalho encontra motivação.

Para facilitar a avaliação, a norma classifica a qualidade dos produtos de *software* em duas partes: a) qualidade interna e qualidade externa e b) qualidade em uso. Os requisitos de qualidade interna podem incluir modelos estáticos e dinâmicos e são utilizados para realizar a especificação dos produtos intermediários. Já os de qualidade externa, são utilizados como meta para validação em vários estágios de desenvolvimento. A qualidade em uso especifica as características e necessidades que, normalmente, o usuário requer que estejam presentes no produto, com a finalidade de satisfazer seus objetivos (ISO/IEC 9126, 1998).

No ciclo de vida de *software* essas partes da qualidade interagem entre si, como ilustrado na figura 5.

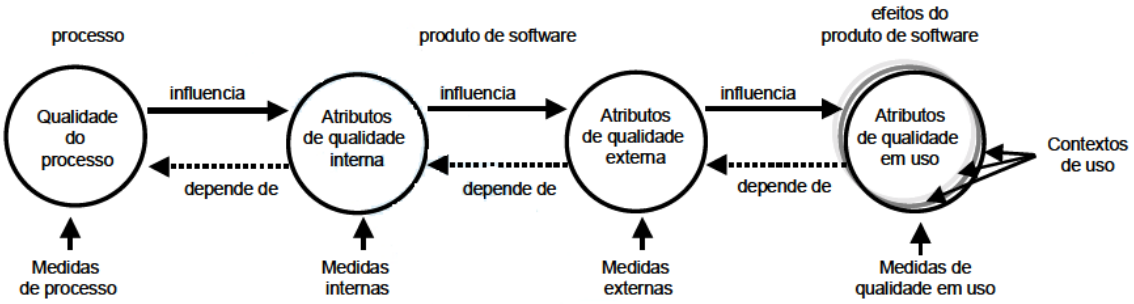


Figura 5 - Qualidade em produtos de software

Fonte: ISO/IEC 9126, 1998

O modelo de qualidade para qualidade externa e interna categoriza os atributos de qualidade em seis características: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade, que são subdivididas em subcaracterísticas medidas por meio de métricas externas e internas. As subcaracterísticas são definidas na norma e estão ilustradas na figura 6. Algumas características e atributos foram utilizados nesta pesquisa para o desenvolvimento do modelo de avaliação de sistemas supervisórios e são detalhadas no capítulo 4.

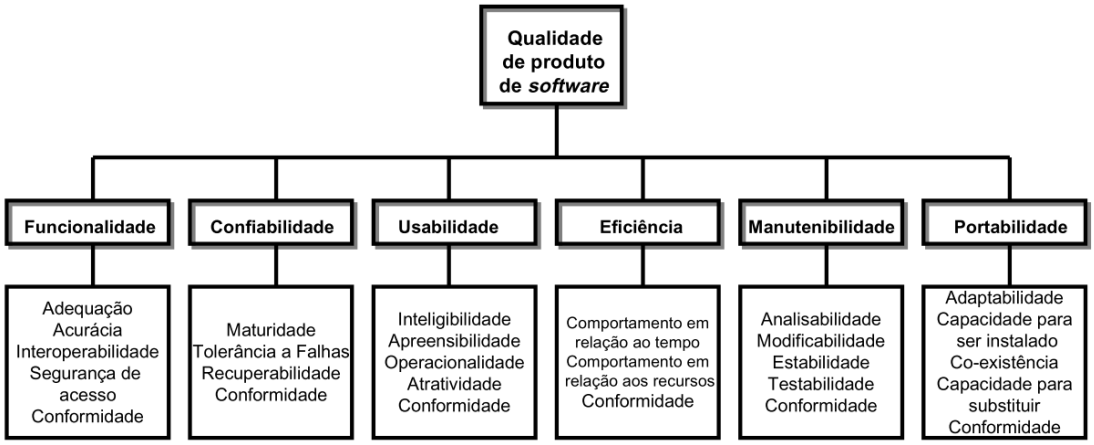


Figura 6 - Modelo de qualidade para qualidade interna e externa

Fonte: ISO/IEC 9126, 1998

2.2.4 Norma ISO/IEC 14598

A norma ISO/IEC 14598 – Tecnologia de Informação – Avaliação de Produto de *Software* estabelece uma estrutura para especificação e avaliação da qualidade dos produtos de *software* e determina os requisitos gerais para os métodos de medição. A partir da aplicação desta norma, alguns resultados são obtidos e podem ser utilizados por

desenvolvedores, analistas, gerentes e mantenedores para realizar a medição da adequação aos requisitos, para executar as melhorias necessárias e para determinar a relação entre métricas externas e internas.

A norma ISO/IEC 14598 é dividida em seis partes:

Parte 1 – Visão geral;

Parte 2 – Planejamento e gestão;

Parte 3 – Processo para desenvolvedores;

Parte 4 – Processo para adquirentes;

Parte 5 – Processos para avaliadores;

Parte 6 – Documentação de módulos para avaliação.

A segunda e sexta parte da norma, geralmente, estão relacionadas ao suporte e gestão da avaliação em nível departamental ou corporativo. Esse suporte está relacionado às atividades como desenvolvimento, controle, normalização, aquisição, transferência e *feedback* da experiência de avaliação dentro da organização. Os módulos da sexta parte da norma contêm a especificação do modelo de qualidade, os dados e informações relativos à aplicação prevista do modelo e informações sobre quais os módulos adequados para as aplicações.

A terceira, quarta e quinta partes desta norma provêm orientações e requisitos para avaliação em nível de projeto, identificando os relacionamentos de cada métrica (externa ou interna) com suas características e subcaracterísticas correspondentes conforme ilustra a figura 7.

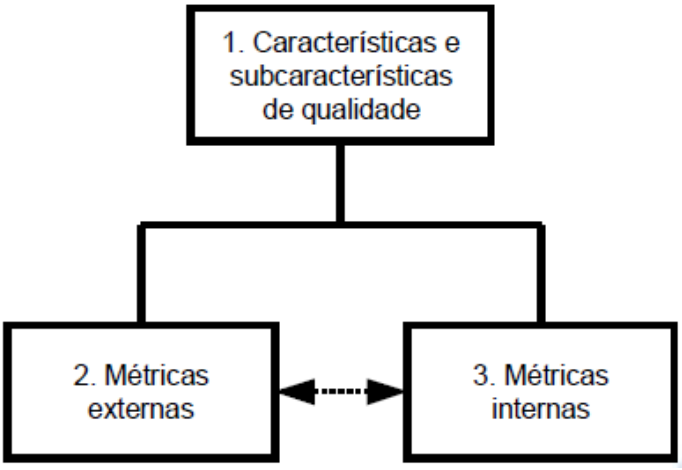


Figura 7 - Características de qualidade e métricas de *software*

Fonte: ISO/IEC 14598, 1998

A visão geral do processo de avaliação da qualidade de *software*, mostrada na norma ISO/IEC 14598-1, é ilustrada na figura 8.

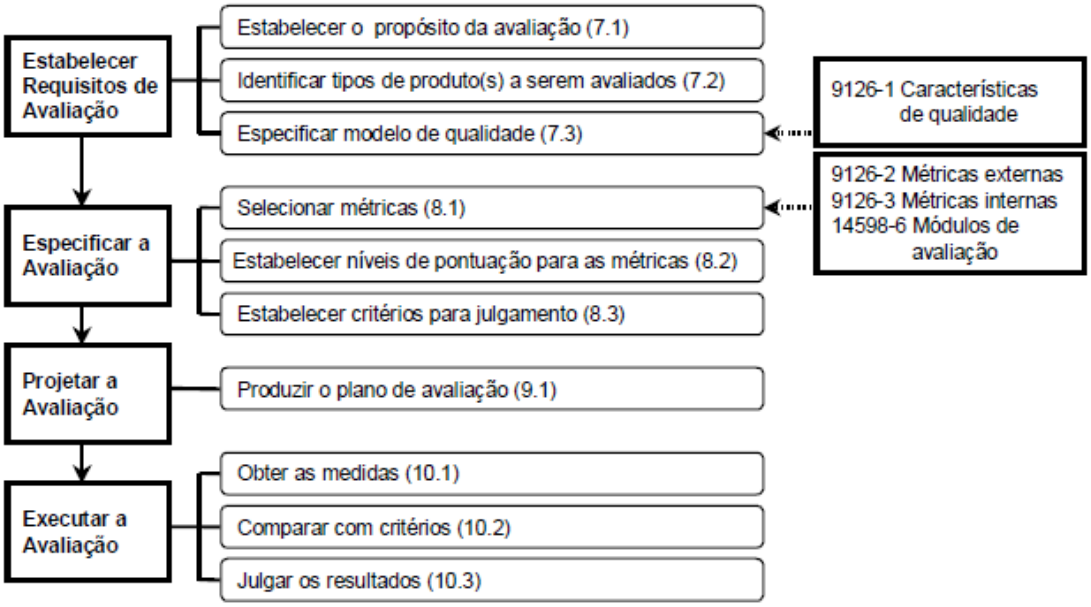


Figura 8 - Processo de avaliação de *software*

Fonte: ISO/IEC 14598, 1998

A determinação de requisitos de avaliação é composta pela escolha do propósito geral da avaliação, pois a norma contém diversos propósitos que podem ser utilizados de acordo com o objetivo da avaliação em produtos intermediários ou em produtos finais. Ainda nesta fase os tipos de produtos a serem avaliados são identificados e o modelo de qualidade a ser utilizado é especificado.

Na especificação da avaliação, as métricas e os seus níveis de pontuação são selecionados para que a mensuração possa ser realizada. Os critérios de julgamento também devem ser estabelecidos e descritos nesta fase. A partir das informações anteriores, o plano de avaliação deve ser projetado.

A execução do plano de avaliação resulta na obtenção das medidas, comparação destas com os critérios de qualidade pré-estabelecidos e julgamento de todos os resultados obtidos com a finalidade de determinar o grau de qualidade daquele *software* desenvolvido.

O processo de avaliação de *software* presente nesta norma é utilizado como referência na proposição de um modelo de avaliação da qualidade de supervisórios e algumas das fases do modelo de avaliação da norma ISO/IEC 14598-1 são mais detalhadas no capítulo 4.

2.2.5 Mensuração e Métricas de Qualidade de *Software*

A eficiência de um modelo de avaliação da qualidade de *software*, como o da norma ISO/IEC 14598-1, depende do processo de medição da presença de características ou requisitos essenciais de produtos e processos de *software*. Para que se possa medir é necessário que se estabeleçam métricas internas e externas.

Conforme Pressman (2006), na maioria das organizações, as medições e as métricas proporcionam o entendimento do processo técnico usado para desenvolver um produto, pois torna possível a quantificação e, por conseguinte, o planejamento do desenvolvimento de *software*. O produto ou o processo são medidos num esforço de torná-los melhores e de aumentar sua qualidade, e as métricas de *software* dão referência a uma grande diversidade destas medidas.

A medição ou mensuração de *software* proporciona a obtenção de valores numéricos para alguns atributos de um produto ou processo de *software*. Esses valores são comparados uns com os outros e também com padrões de um modelo com a finalidade de produzir resultados a respeito da qualidade de *software* ou dos processos de *software*. Uma métrica de *software* é utilizada em qualquer tipo de medição que faça referência a um processo de *software*, sistema ou documentação relacionada (SOMMERVILLE, 2011).

Segundo ABNT (1999), a forma de definição das características de qualidade de *software* não permite sua medição direta, por isso o estabelecimento de métricas que se correlacionem com os requisitos de *software* é necessário. Todo atributo interno ou externo quantificável, que interage com o seu ambiente e se correlaciona com uma característica, pode ser definido como uma métrica.

O padrão IEEE 1061 (1998) indica que o estabelecimento de um programa de métricas é importante para a realização da mensuração, por isso descreve uma metodologia para validação das métricas, onde classifica as mesmas em duas categorias: de processo e de produto. A utilização das métricas de processo permite a medição de características dos métodos, ferramentas e técnicas usadas no desenvolvimento de *software*. Já as métricas de produto podem ser utilizadas para medir as características do código e da documentação do *software*.

A utilização de métricas, antes que o *software* seja entregue, provêm uma base quantitativa para se tomar decisões referentes a projetos e testes. O uso de métricas de

qualidade é descrito como a componente chave da garantia estatística da qualidade (PRESSMAN, 2006).

A métrica de *software*, segundo Boente (2009), tem como princípios estabelecer as funções de coleta de dados de avaliação e desempenho, atribuir essas responsabilidades a toda a equipe envolvida no projeto, reunir dados de desempenho pertencentes à complementação do *software*, analisar os dados dos projetos anteriores para especificar o efeito desses fatores e utilizá-los em previsões futuras.

A definição de métricas, segundo Belchior (1997), é descrita como um processo em que símbolos ou números são atribuídos a características de entidades do mundo real, descrevendo-as segundo regras claramente definidas.

As métricas devem ser escolhidas e analisadas de acordo com a natureza dos atributos específicos de qualidade de *software* e das características do projeto. Alguns atributos são essencialmente quantitativos como, por exemplo, o número de linhas de código, número de mensagens de erro e as horas de trabalho. Entretanto algumas características não possuem a mesma precisão, e a subjetividade de alguns requisitos torna a utilização de métricas objetivas uma tarefa difícil, pois requer um nível de abstração maior para definir e aplicar medidas (OLIVEIRA, 2002).

Segundo Belchior (1997), métricas de qualidade são divididas em categorias. As principais categorias são:

- Métricas subjetivas: são medidas relativas baseadas em estimativas pessoais ou de grupo, sendo obtidas por meio de termos linguísticos como muito, médio, pouco;
- Métricas objetivas: são facilmente quantificadas e medidas através de expressões numéricas ou representações gráficas dessas expressões;
- Métricas explícitas ou diretas: não dependem da medida de outro atributo, quantificando um fator observado no produto;
- Métricas derivadas ou indiretas: envolvem medidas de um ou mais atributos a ele relacionados;
- Métricas de produto: indicam, objetivamente, características mensuráveis do produto de *software* tal como tamanho, complexidade e acoplamento;
- Métricas de processo: medem aspectos de desenvolvimento e manutenção e são, geralmente, usadas para caracterizar os custos destas atividades;
- Métricas absolutas: são tipicamente invariantes para a medição de novos itens;
- Métricas relativas: podem variar quando novos itens são medidos;

- Métricas dinâmicas: possuem uma dimensão temporal;
- Métricas estáticas: permanecem invariáveis a despeito do tempo;
- Métricas explanatórias: conhecidas também como métricas de resultado. São geradas depois do fato ocorrido, baseadas em dados coletados, indicando, simplesmente, o estado atual do produto;
- Métricas preditivas: podem ser obtidas ou geradas previamente, para realizar prognósticos do valor de uma propriedade do sistema, que somente se tornará diretamente observável em um estágio posterior a seu desenvolvimento;

As métricas mais usuais são padronizadas pelas normas ISO/IEC 9126 (segunda e terceira parte) e ISO/IEC 14598-6 para as aplicações em diversos tipos de *software*. Os objetivos das normas ISO/IEC 9126-2 e 9126-3 são: apresentar métricas internas e externas para medir os atributos das seis características de qualidade de *software* da norma ISO/IEC 9126-1; definir quais características mensuráveis contribuem para as características de qualidade de software; e identificar as propriedades desejáveis para uma métrica. As métricas internas medem propriedades intrínsecas e tem a função de assegurar que a qualidade externa e a qualidade em uso sejam alcançadas.

A norma ISO/IEC 14598-6 tem por finalidade definir a estrutura e o conteúdo da documentação a ser utilizada para descrever um módulo de avaliação, o qual especifica os métodos aplicáveis à avaliação de características de qualidade (ABNT, 1999).

Segundo a norma ISO/IEC 14598, utilizando as métricas de qualidade, podem-se medir quantitativamente as particularidades quantificáveis. O valor medido, ou seja, o resultado é mapeado numa escala que precisa ser dividida em faixas correspondentes aos diversos graus de satisfação dos requisitos para que o nível de satisfação seja visualizado. São exemplos:

- Dividir a escala em duas categorias: satisfatória e insatisfatória;
- Dividir a escala em quatro categorias, como ilustrado na figura 9, delimitadas por: o pior caso, o nível atual para um produto existente ou alternativo, e o nível planejado. O nível atual é o estabelecido para controlar se o novo sistema não se deteriora em relação à situação atual. O nível planejado é o que é considerado alcançável com os recursos disponíveis. O pior caso é o limite para a aceitação pelo usuário, no caso em que o produto não alcance o nível planejado.

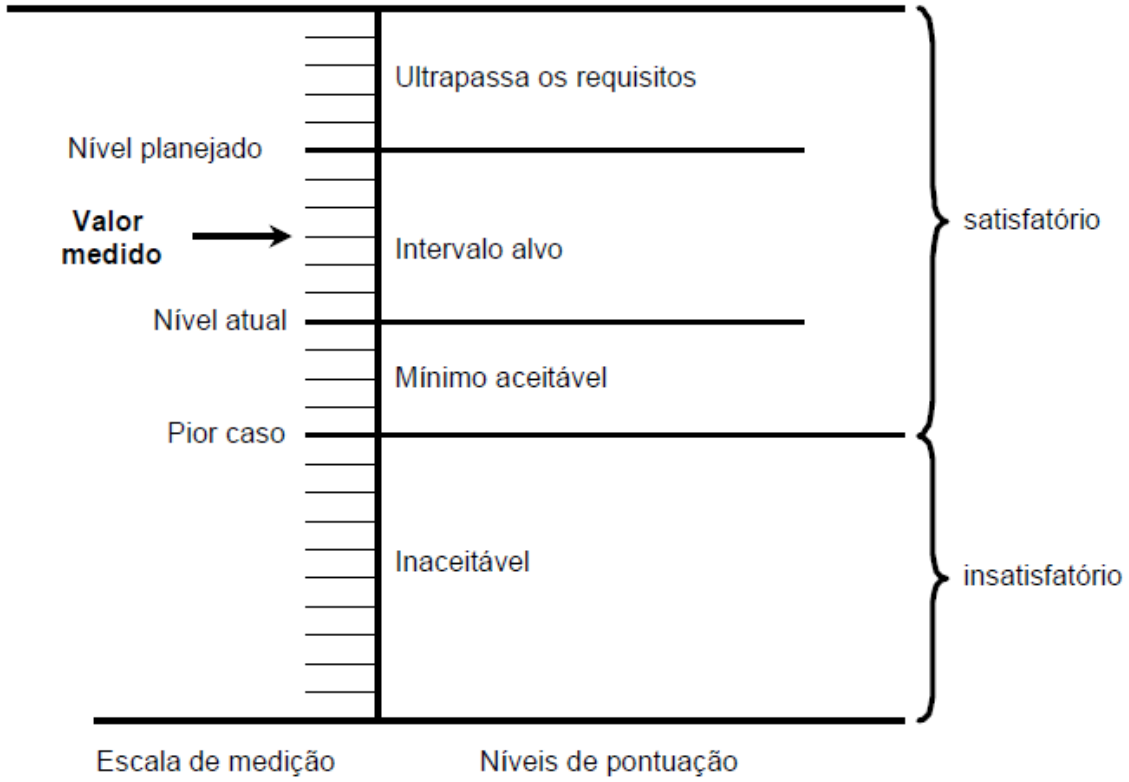


Figura 9: Níveis de pontuação para as métricas

Fonte: ISO/IEC 14598, 1998

Um avaliador deve observar dois pontos importantes ao elaborar uma escala (ABNT, 1999):

- Cada métrica está relacionada a uma escala específica. Um grande número de escalas pode acarretar um grande trabalho para estabelecer e documentar todas as escalas;
- Os níveis de pontuação (ou faixas de corte – "mínimo aceitável", intervalo alvo, inaceitável) podem não ser conhecidos previamente. Esses níveis são particulares de cada caso e de cada empresa.

Segundo a ISO/IEC 14598-1, especificações de requisitos de qualidade de *software* devem ser definidas usando um padrão de qualidade apropriado e bem definido. Convém que, para este propósito, o modelo de qualidade e as definições da NBR ISO/IEC 9126-1 sejam usados completamente ou em parte.

A ISO/IEC 14598-1 ainda aconselha o avaliador a sintetizar o resultado da avaliação de cada característica, preparando um procedimento para isto, com critérios diferentes para características de qualidade diferentes, onde cada característica pode estar representada em termos de suas subcaracterísticas.

O avaliador pode avaliar os atributos do padrão de qualidade de *software* através da opinião subjetiva de especialistas da área. Os termos utilizados nos questionários desta pesquisa são, por exemplo, moderadamente presente, pouco importante, etc. Essa subjetividade pode ser convertida em uma escala e transformada em números através da lógica *fuzzy*. Portanto, o tratamento dos dados necessita de um arcabouço teórico que dê suporte para o julgamento de qualidade dos supervisórios desenvolvidos.

2.3 Teoria *Fuzzy*

O filósofo grego Aristóteles estabeleceu, na Grécia Antiga, a Lógica clássica como ciência que estuda a relação de consequência dedutiva com um conjunto de regras rígidas, que através de premissas proporcionavam conclusões logicamente válidas. Com a criação do Silogismo, a Lógica foi iniciada através das noções de possibilidade e necessidade. Exemplo: Considerando a primeira premissa como "João acorda cedo todos os dias para trabalhar" e sabendo que "Quem começa a trabalhar de manhã bem cedo tem uma vida saudável" (segunda premissa), pode-se chegar a conclusão de que "João tem uma vida saudável".

Essas suposições ou premissas, quando verdadeiras, levam a uma conclusão verdadeira e quando falsas, produzem uma conclusão falsa. Portanto a lógica clássica, booleana ou binária é composta de afirmações falsas ou verdadeiras e não permite declarações parcialmente verdadeiras ou muito falsas.

A precisão da lógica clássica pode ser inútil em diversos casos, enquanto que as instruções não precisas podem ser melhores interpretadas e realizadas. As teorias mais conhecidas para tratar os aspectos da imperfeição da informação como a imprecisão e a incerteza são a teoria dos conjuntos e a teoria das probabilidades, respectivamente, contudo nem estas possuem a compatibilidade completa com a riqueza das informações humanas (COSTA, 2010).

As ambiguidades e multiplicidades de sentidos compõem a maior parte da linguagem natural. Como exemplo pode-se citar os adjetivos que as pessoas utilizam para caracterizar objetos ou situações que não são suficientemente claras, sendo ambíguas em termos de amplitude de significados. Percebe-se que em Engenharia, os adjetivos que descrevem estados ou condições são, quase sempre, relacionados a quantidades. A maioria

dos adjetivos é quantificada em valores abstratos através de sentidos como extensão, altura ou idade (MORÉ, 2004).

A necessidade de considerar as informações subjetivas do conhecimento humano e a ausência de ferramentas que auxiliassem em situações de incertezas motivaram Zadeh, no ano de 1965, a criar a lógica *Fuzzy*. Os conceitos subjetivos do raciocínio impreciso tais como muito, pouco, alto, baixo, velho e novo são incorporados em classes de objetos na teoria *fuzzy*, onde a pertinência ou não de um elemento a um conjunto acontece de forma gradual e não abrupta (ZADEH, 1990).

A capacidade da lógica *fuzzy* de armazenar e tratar informações que não são suficientemente claras, através de variáveis linguísticas, permite a conversão das informações vagas, imprecisas ou mal definidas em valores numéricos, que podem ser tratados por computadores com o objetivo de realizar extração de tendências, inferências estatísticas, controle de processos e geração de gráficos.

A lógica *fuzzy* é uma das tecnologias atuais mais bem sucedidas para o desenvolvimento de sistemas, para realizar inferências estatísticas ou controlar processos sofisticados (CARULO, 2008).

A teoria, propriedades e funções da lógica *fuzzy* compõem uma estrutura com ferramentas robustas que podem ser aplicadas no tratamento de informações fornecidas pelos seres humanos em diversas áreas do conhecimento, tais como:

- Automação e controle de processos industriais;
- Inferência estatística;
- Processos de manufatura;
- Avaliação de *software*;
- Supervisão da produção;
- Navegação e localização de robôs;
- Sistemas especialistas;
- Medicina;
- Reconhecimento de padrões.

A lógica *fuzzy* juntamente com a teoria dos conjuntos *fuzzy* fornecem as ferramentas necessárias para o desenvolvimento de produtos de *software* que modelam o raciocínio humano (também chamado de raciocínio aproximado). Nos conjuntos *fuzzy*, um elemento possui um grau de pertinência a um conjunto, que indica a certeza ou incerteza da

associação (ENGELBRECHT, 2007). Todo esse arcabouço teórico permite a criação de sistemas de inferência baseado em regras, onde cada regra é representada por uma relação *fuzzy* que em conjunto descreve o comportamento do sistema. Neste tipo de sistema não é necessário conhecer o modelo matemático do processo (FERREIRA JÚNIOR, 2009).

2.3.1 Conjuntos *Fuzzy*

Na teoria clássica bem como na lógica de Boole, um conjunto clássico, também chamado de *crisp*, é formado por diversos objetos e cada objeto só tem duas possibilidades de possuir uma relação de pertinência com cada conjunto: ou o objeto pertence ao conjunto, ou o objeto não pertence. Já a teoria dos conjuntos *fuzzy* ou conjuntos nebulosos permite que um objeto pertença a mais de um conjunto simultaneamente, através da atribuição a este objeto de diversos graus de pertinência para cada um dos conjuntos a que ele pertence (COSTA, 2010).

Nos conjuntos clássicos a função está relacionada somente com os valores 0 ou 1, pode ser dada pelo gráfico ilustrado na figura 10 e pode possuir a representação a seguir.

$I_A(x) = 1$ se e somente se $x \epsilon A$

0 se somente se x não pertence A

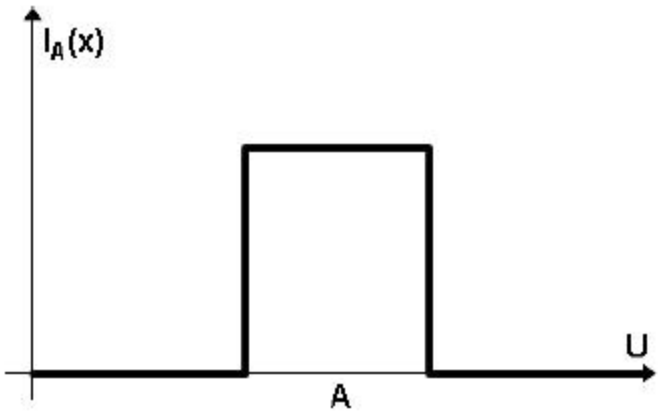


Figura 10 - Função característica de A

Fonte: COSTA, 2010

Nos conjuntos *fuzzy* a função entre elementos e conjuntos é chamada de função de pertinência e é definida por $\mu_A(x)$: $X \rightarrow [0,1]$ e possui pares ordenados que a representam:

$A = \{(\mu_A(x) / x)\ x\ \varepsilon\ X\}$, onde μ_A é um valor entre 0 e 1 que indica o quanto o elemento x pertence ao conjunto A, considerando que a função possui valor 1 quando x pertence totalmente ao conjunto e 0 quando não pertence (MALVEZZI, 2010).

A representação de um elemento *fuzzy* é denotada pelo par $(x, \mu_A(x))$ ou pelo par $(\mu_A(x) / x)$. Um conjunto *fuzzy* pode ser exemplificado como segue (OLIVEIRA, 2002):

$$A = \{(b; 0,7), (d; 0,1), (f; 1)\} \text{ ou } A = \{(0,7 / b), (0,1 / d), (1 / f)\}$$

Essa representação diz que o elemento "b" pertence ao conjunto A com um grau de 70%, o elemento "d" pertence somente 10% e o elemento "f" pertence totalmente (100%) ao conjunto.

As funções de pertinência podem ser de diferentes tipos. Os tipos mais utilizados são a função triangular, crescente, gaussiana, decrescente e a função trapezoidal (CÁRDENAS, 2011). As figuras 11 e 12 mostram o gráfico e a expressão algébrica da função triangular e trapezoidal.

A função triangular é definida pelos parâmetros (a, m, b), tal que $a \leq m \leq b$.

$$\mu_F(x) = \begin{cases} 0 & se\ \ x \leq a \\ \dfrac{x-a}{m-a} & se\ x \in (a,m) \\ 1 & se\ \ x = m \\ \dfrac{b-x}{b-m} & se\ x \in (m,b) \\ 0 & se\ \ x \geq b \end{cases}$$

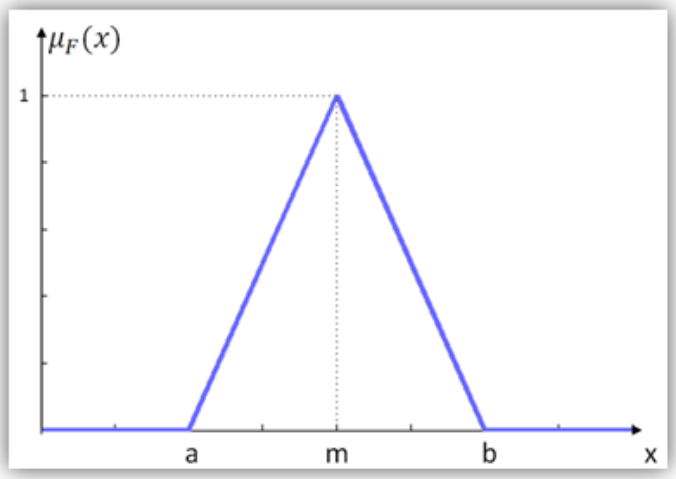


Figura 11 - Função triangular *fuzzy*

A função trapezoidal é definida pelos parâmetros (a, m, n, b), obedecendo à expressão: $a \leq m < n \leq b$.

$$\mu_F(x) = \begin{cases} 0 & se\ x \leq a \\ \dfrac{x-a}{m-a} & se\ x \in (a,m) \\ 1 & se\ x \in [m,n] \\ \dfrac{b-x}{b-m} & se\ x \in (n,b) \\ 0 & se\ x \geq b \end{cases}$$

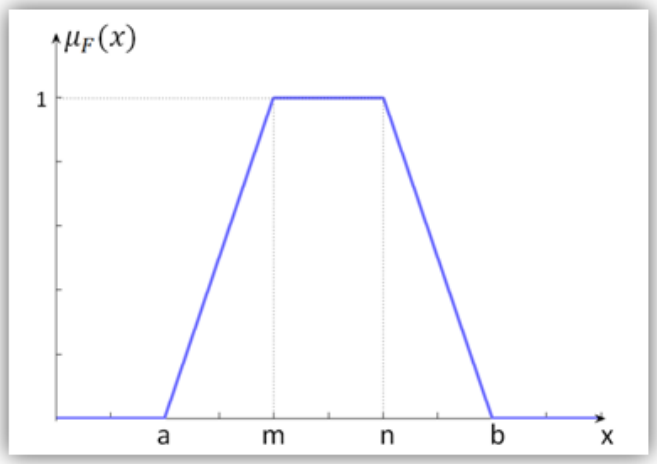


Figura 12 - Função trapezoidal *fuzzy*

2.3.2 Operações com Conjuntos *Fuzzy*

A teoria dos conjuntos *fuzzy* também define propriedades e operações básicas entre conjuntos como união, interseção e complemento. Desta forma, estas definições proporcionam a expansão e estendem os conceitos da teoria clássica, transformando-as em casos particulares da lógica *fuzzy*.

Segundo Munakata (2008), seja um conjunto $A = \{u/m_A(u) \mid u \in U\}$ e um conjunto $B = \{u/m_B(u) \mid u \in U\}$, onde u é um elemento e $m_A(u)$ e $m_B(u)$ são as funções de pertinência. O conjunto *fuzzy* F, que é formado pela união de A e B, é definido por:

$$F = A \cup B = \{x / \max (m_A(x), m_B(x)) \mid x \in U\}$$

A utilização do operador max (máximo) e do operador min (mínimo) proporciona a seleção, respectivamente, das maiores e menores funções de pertinência dos conjuntos A e B. A Figura 13 ilustra as operações de união, interseção entre dois conjuntos e a operação de complemento. O conjunto *fuzzy* G formado pela interseção dos conjuntos A e B é definido por:

$$G = A \cap B = \{x / \min (m_A(x), m_B(x)) \mid x \in U\}$$

Logo a função de pertinência de G é definida pela equação 1.

$$m_{A \cap B}(x) = m_A(x) \bullet m_B(x) \qquad (1)$$

Onde o operador "•" equivale à função booleana AND.

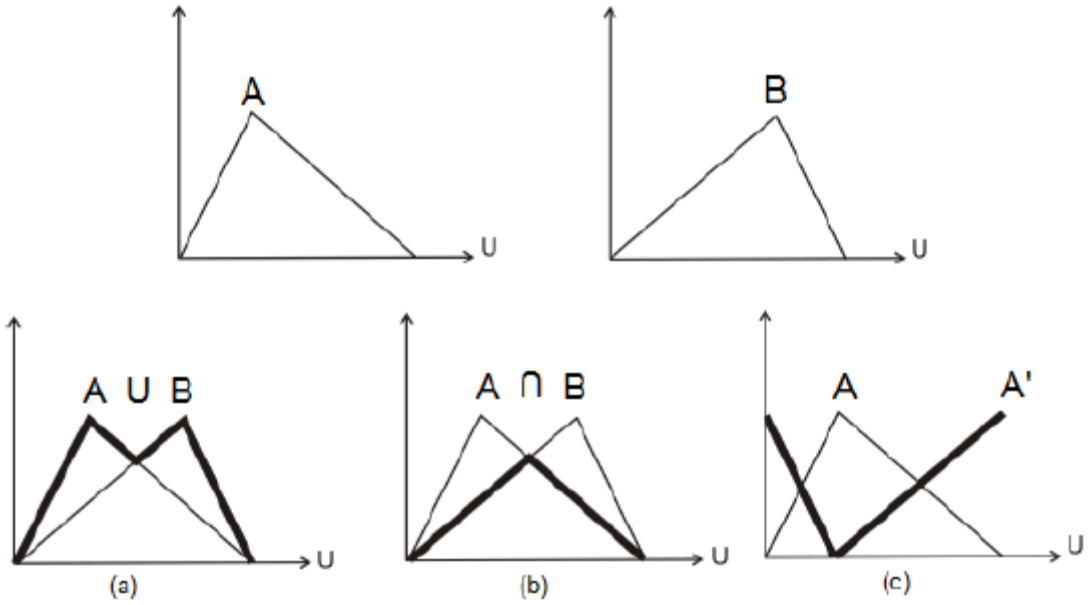


Figura 13 - Operações entre conjuntos *fuzzy*. (a) união; (b) interseção; (c) complemento
Fonte: MENEGOTTO, 2011

O complemento de um conjunto *fuzzy* A, denotado por A’ ou $\bar{A}$ é definido por:

$$A' = \{x / (1 - m_A(x)) \mid x \varepsilon U\}$$

Um exemplo de operações entre conjuntos *fuzzy* é mostrado na tabela 1, onde dois conjuntos A e B representam o grupo de pessoas altas e pessoas baixas.

Tabela 1 - Exemplo de operações entre conjuntos *fuzzy*.

Altura de Pessoas	**Altas (A)**	**Baixas (B)**	**A U B (max)**	**A ∩ B (min)**	**$\bar{A}$**
Altura de Júlio	0,75	0,40	0,75	0,40	0,25
Altura de Norma	0,55	0,65	0,65	0,55	0,45
Altura de Carlos	0,90	0,20	0,90	0,20	0,10

2.3.3 Números *Fuzzy*

Os números *fuzzy* são utilizados para quantificar atributos físicos da realidade que estão associados à imprecisão ou mesmo a conceitos humanos vagos (MORÉ, 2004).

Um número *fuzzy* Ň (ou um intervalo *fuzzy*) é um conjunto convexo e normalizado definido no conjunto dos números reais R, tal que sua função de pertinência está entre 0 e 1. A concepção intuitiva de números ou intervalos aproximados, tal como valores em torno de um dado intervalo, é capturada por um número *fuzzy*. Esses conceitos são fundamentais para a caracterização dos estados das variáveis *fuzzy* e as aplicações de inferência *fuzzy* (BELCHIOR, 1997).

Para que um conjunto *fuzzy* A possa ser qualificado como um número *fuzzy*, ele deve possuir, no mínimo, as seguintes propriedades (BELCHIOR, 1997):

I) A deve ser um conjunto *fuzzy* normalizado;
II) Aα deve ser um intervalo fechado para todo α ϵ (0,1], isto é, todo número *fuzzy* é convexo;
III) O suporte de A deve ser limitado.

2.3.4 Variáveis Linguísticas

Termos linguísticos como muito, pouco, alto, baixo, médio, que determinam o estado de uma variável, são empregados frequentemente. Os conjuntos *fuzzy* podem representar esses conceitos linguísticos e essa variável é denominada variável linguística ou variável *fuzzy* (SILVEIRA, 2011).

Variáveis linguísticas, segundo Zadeh (1965), são variáveis em que os valores são sentenças de uma linguagem natural, ou seja, cada palavra x em uma linguagem natural L pode ser vista como uma descrição sumarizada de um subconjunto *fuzzy* M(x) de um universo de discurso U, com M(x) representando o significado de x. Neste sentido a linguagem como um todo pode ser vista como um sistema pela atribuição de etiquetas atômicas e compostas (isto é, palavras, frases e sentenças) para os subconjuntos *fuzzy* de U.

Variável linguística é um tipo de variável que tem seu valor expresso qualitativamente por termos linguísticos, que fornece um determinado conceito à variável e quantitativamente por uma função de pertinência. As sentenças em uma linguagem especificada, construída a partir de termos primários (grande, pequeno, médio, alto, baixo, por exemplo), de conectivos de lógica *fuzzy*: negação, conectivos “e” e “ou”, de modificadores (muito, pouco, velozmente, frequentemente), podem caracterizar os valores de uma variável linguística (MENEGOTTO, 2011).

A variável linguística temperatura, por exemplo, pode ser estruturada pelos termos *fuzzy* muito frio, frio, quente e muito quente. Cada um desses termos linguísticos associa-se a um conjunto *fuzzy* que o caracteriza, e o domínio desta variável é o intervalo [T_{MIN}, T_{MAX}] (MENEGOTTO, 2011). A figura 14 ilustra o exemplo descrito anteriormente que tem como escala: [T_{MIN}, T_{MAX}] = [-10, 50].

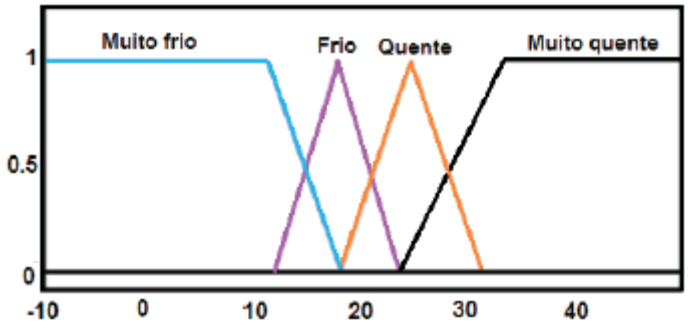


Figura 14 - Representação gráfica da variável temperatura
Fonte: MENEGOTTO, 2011

Utilizando o exemplo da variável temperatura, pode-se caracterizar totalmente a variável linguística por uma quíntupla:

X: Temperatura

T(x): {muito frio, frio, quente, muito quente}

U: escala da temperatura de -10 até 50

G: temperatura "muito frio", por exemplo.

M: relaciona o valor a um conjunto *fuzzy*

Nesta quíntupla, o nome da variável é X. O conjunto dos termos linguísticos de x é T(x), ou simplesmente T, que se referem a uma variável base u, cujos valores estão no conjunto universo U. G é uma regra sintática para a geração dos termos linguísticos. M é uma regra semântica que associa a cada termo linguístico t ϵ T o seu significado. M(t) é um conjunto *fuzzy* em U (BELCHIOR, 1997) (PEDRYCZ e GOMIDE, 1998).

A modelagem das variáveis é feita com a construção de funções de pertinência que podem assumir as formas triangular, trapezoidal, gaussiana, entre outras. A transição gradual entre estados e a conversão de informações qualitativas em formas que podem ser implementadas computacionalmente são permitidas com a utilização de variáveis *fuzzy* ou variáveis linguísticas (SILVEIRA, 2011).

2.3.5 Sistemas *Fuzzy*

Uma das principais utilidades da teoria *fuzzy* é o mapeamento de um espaço de entrada para um espaço de saída, e o principal mecanismo necessário para realizar este feito é uma lista de declarações chamadas regras *fuzzy*. Todas as regras são úteis porque se referem às variáveis e aos adjetivos que descrevem essas variáveis. A avaliação dessas regras é realizada em paralelo e a ordem de avaliação faz diferença no resultado (MALVEZZI, 2010).

Ribeiro de Sá (2007) define um sistema *fuzzy* como um sistema composto de entrada, fuzzificação, base de regras, procedimento de inferência, defuzzificação e saída.

Conforme Costa (2010), sistemas de inferência *fuzzy* são ferramentas computacionais utilizadas nas mais diversas áreas da engenharia e da ciência, e agregam os conceitos de conjuntos *fuzzy*, variáveis linguísticas e raciocínio aproximado, processando dados por meio de mecanismos de inferência.

Segundo Munakata (2008), os principais passos para o desenvolvimento de um sistema *fuzzy* são:

i) Constatação de que o conhecimento sobre o ambiente da aplicação em questão é descrito de forma aproximada ou com regras heurísticas;
ii) Identificação das entradas e saídas do sistema e seus respectivos intervalos de valores.
iii) Definição de uma função para cada parâmetro de entrada e saída. A quantidade de funções requeridas depende do desenvolvedor e do ambiente do sistema;
iv) Construção de uma base de regras pelo projetista, que determine quantas regras serão necessárias e quando parar de adicionar novas regras;
v) Verificação das saídas da base de regras e se seus intervalos de valores estão corretos e em conformidade com o conjunto de entradas usado, permitindo uma posterior validação.

A compreensão dos sistemas de inferência *fuzzy* está vinculada a alguns conceitos da teoria *fuzzy*. Fuzzificação, segundo Ribeiro de Sá (2007), é o processo de transformação da entrada em graus de pertinência produzindo uma interpretação da entrada, os quais caracterizam o estado do sistema (variáveis de estado). Estes valores são então fuzzificados, com a transformação da entrada, por exemplo, ruim, péssimo, importante, etc., em conjuntos nebulosos para que possam se tornar instâncias de variáveis linguísticas.

Na teoria dos conjuntos *fuzzy*, a defuzzificação, segundo Menegotto (2011), é um processo que permite representar um conjunto *fuzzy* por um valor *crisp* (número real). Os

métodos de defuzzificação mais comuns são centro de gravidade, centro dos máximos e média dos máximos.

Na estrutura básica de um sistema de inferência *fuzzy*, ilustrado na figura 15, as entradas são normalmente precisas e advêm de medições, testes, avaliações, percepções observadas em um conjunto de dados (MALVEZZI, 2010).

Na primeira fase de inferência, o fuzzificador relaciona o valor preciso da entrada com um valor nebuloso. Desta forma os valores nebulosos são obtidos e enviados para um mecanismo chamado motor de inferência, que é responsável por realizar as inferências do sistema, gerando um valor nebuloso para a saída. Cada regra possui uma consequência e o resultado global da etapa inferência depende da combinação das consequências chamada de agregação *fuzzy* (COSTA, 2010).

A ideia principal do processo de agregação é obter um grau de consenso entre as informações disponíveis, calculando-se um valor final. Quando estes dados forem extraídos de especialistas será permitido o cálculo da taxa de aceitação ou rejeição entre eles, ou seja, o grau em que os especialistas concordam em suas estimativas (BELCHIOR, 1997). Já a fase de defuzzificação, que pode ser realizada através de diversos métodos, compreende a transformação do resultado da agregação *fuzzy* em saídas não nebulosas ou saídas precisas.

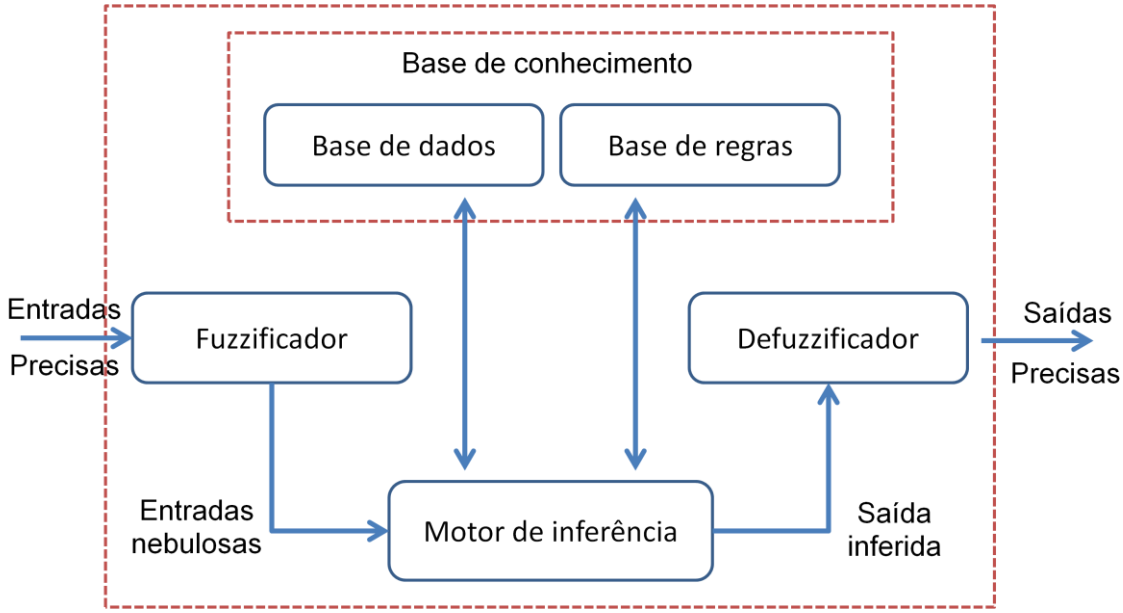


Figura 15 - Sistema de Inferência *Fuzzy*

3. MODELO DE DESENVOLVIMENTO DE SUPERVISÓRIOS

Um processo de *software* é um conjunto de atividades relacionadas que levam à produção de um aplicativo. Essas atividades podem envolver o desenvolvimento de *software* a partir do zero em uma linguagem padrão. Os processos de *software* são planejados através de modelos, que são representações simplificadas e representam uma perspectiva particular de um processo (SOMMERVILLE, 2011). Dentre os modelos de desenvolvimento mais utilizados podem-se citar a modelagem ágil, os modelos orientados a reuso e os modelos tradicionais.

Os modelos de desenvolvimento de *software* não servem indiscriminadamente para qualquer caso e, por isso, é necessário analisá-los individualmente. Segundo Sommerville (2011), embora não exista um processo "ideal", há espaço, em muitas organizações para melhorias no processo de *software*. A maioria dos processos atuais é composta por técnicas ultrapassadas e as empresas não aproveitam as melhores práticas de engenharia de *software* na indústria.

Os modelos tradicionais mostram-se eficientes em diversas aplicações. Os modelos ágeis introduziram novas ideias para abordar projetos únicos (sob encomenda) associados à criatividade e baseados em conhecimento (MARTINS, 2007).

Já os modelos orientados a reuso proporcionam a possibilidade de reutilização de estruturas, objetos, componentes dos softwares.

Os modelos são bem distintos, o modelo de desenvolvimento ágil é uma metodologia moderna de desenvolvimento, que se baseia na ideia de modelar somente o necessário e nada mais, de forma a tornar o desenvolvimento mais dinâmico e capaz de lidar rapidamente com mudanças nos requisitos.

Segundo Sommerville (2011), esses processos de desenvolvimento rápidos são concebidos para produzir, rapidamente *softwares* úteis. O sistema é desenvolvido como uma série de versões (incrementos), de maneira que cada versão acrescenta funcionalidade à anterior. O desenvolvimento ágil tem vantagens importantes quando comparado aos modelos tradicionais:

- A quantidade de documentação a ser refeita é muito menor e o custo de realizar mudanças nos requisitos é reduzido;
- É mais fácil obter *feedback* dos clientes sobre o desenvolvimento que foi feito;
- É possível obter entrega e implementação rápida de um *software* útil ao cliente;

- É possível mudar os requisitos de *software* ao longo do projeto.

Já os modelos tradicionais são exemplos de processos dirigidos a planos, em que o desenvolvedor deve projetar todas as atividades do processo antes de começar a trabalhar nelas. Em princípio, o resultado de cada estágio é a aprovação de um ou mais documentos. O estágio seguinte não deve ser iniciado até que a fase anterior seja concluída (SOMMERVILLE, 2011). Percebe-se que, por causa dos custos de produção e aprovação de documentos, as iterações podem ser dispendiosas e envolver significativo retrabalho apesar de este modelo proporcionar bastante organização (KOSCIANSKI e SOARES, 2006).

Esses modelos não são mutuamente exclusivos e muitas vezes são usados em conjunto, especialmente para o desenvolvimento de sistemas de grande porte. Logo, faz sentido combinar algumas das melhores características dos modelos tradicionais e dos modelos ágeis. É preciso ter informações sobre os requisitos essenciais do sistema para projetar uma arquitetura de *software* que dê suporte a esses requisitos (SOMMERVILLE, 2011).

O modelo de desenvolvimento proposto neste trabalho deve considerar algumas informações de supervisórios mostradas a seguir:

1. O projeto de um supervisório, embora seja na maioria das vezes para um sistema novo, se apoia sobre um ambiente de desenvolvimento já existente e é composto de inovações e recursos já conhecidos;
2. Projetos anteriores são bastante semelhantes a novos projetos;
3. Os requisitos de *software* geralmente não se alteram;
4. É necessário um plano que detalhe as principais realizações que precisam acontecer e as estimativas globais;
5. Geralmente os clientes não têm tempo de acompanhar o projeto com uma frequência semanal;
6. O desenvolvimento de um supervisório pode ser realizado por mais de um programador (programação em par) através de uma clara divisão de tarefas.
7. O cliente participará de algumas fases do projeto, fornecendo *feedback* para os programadores.
8. Os supervisórios desenvolvidos raramente sofrem futuras modificações, exceto no caso de expansões do processo supervisionado.

9. A entrega e implantação de um *software* podem ser feitas por incrementos (uma parte de cada vez).

Essas características do desenvolvimento de um supervisório estão relacionadas com os modelos ágeis, os modelos orientados a reuso e os modelos tradicionais através da tabela 2. Esta tabela indica o modelo em que a característica mais se aproxima quanto à forma de desenvolvimento e demonstra as especificações dessas características.

Tabela 2 - Relação entre características do supervisório

Característica	**Modelo**	**Especificações**
1º Reuso de componentes	Reuso	Alguns recursos são reutilizáveis, otimizando o tempo de desenvolvimento de novos supervisórios. Exemplos: Figuras que representam botões de menu personalizáveis, equipamentos da planta, campos de preenchimento de relatórios, entre outros.
2º Reutilização de requisitos	Reuso	Os requisitos de projetos de supervisórios são bastante semelhantes entre si. Devido a essa característica, os novos projetos têm prazos menores de entrega. Exemplos: Reutilização de telas de alarmes, históricos, relatórios de supervisórios anteriores. Reutilização da configuração da comunicação confiável com controladores de mesmo fabricante, entre outros.
3º Definição inicial dos requisitos	Tradicional	Como os processos industriais são difíceis de ser alterados e a maioria dos requisitos entre projetos de supervisórios são parecidos, então, a maioria dos requisitos podem ser especificados integralmente no início do projeto.
4º Planejamento inicial de etapas	Tradicional	Um planejamento inicial bem feito pode facilitar muito o projeto, pois a sequência de fases de desenvolvimento de um supervisório é bem definida e permite a elaboração de um plano de atividades que oriente o programador antes da implantação.

5º Acompanhamento dos *stakeholders*	Tradicional	Os clientes, que julgam o preenchimento dos requisitos no supervisório, são muito ocupados e dispõem de pouco tempo para acompanhar semanalmente o projeto. Verifica-se também que os custos de projeto já são bastante elevados e é inviável contratar um profissional que fique acompanhando o desenvolvimento. Geralmente, quando se contrata esse profissional, eles estão sujeitos a diversas pressões e não podem participar plenamente do desenvolvimento do supervisório.
6º Programação em par	Ágil	A programação em par acelera bastante o desenvolvimento. Um digita enquanto o outro revisa, corrige e sugere. Essa prática leva a uma redução drástica de *bugs*. As habilidades da equipe de desenvolvimento devem ser reconhecidas e exploradas.
7º *Feedback* do cliente	Ágil	O cliente acompanhará o projeto através de poucas fases. Esse acompanhamento pode sanar possíveis erros nos requisitos. A interação do cliente com o *software* pode acontecer através da visualização do *software* ou por vídeos criados para mostrá-lo em funcionamento.
8º Não alteração dos requisitos	Tradicional	A maioria dos supervisórios desenvolvidos não evolui com o tempo. Então, dificilmente, os requisitos daquele *software* específico se alteram após a implantação. Não havendo necessidade de estabelecimento de contratos de escopo variável.
9º Entrega incremental do produto	Ágil	As telas principais e as medidas de segurança de um supervisório podem ser entregues logo, desde que supervisionem parcialmente o processo industrial. Em seguida, as outras telas podem ser implantadas.

Nota-se que os requisitos elencados para o desenvolvimento de sistemas supervisórios não podem ser atendidos por um só tipo de modelo e a adoção de diversas características dos modelos pode proporcionar o aproveitamento das vantagens de múltiplas abordagens. A escolha pela metodologia tradicional implicaria num alto custo de

desenvolvimento de documentação e retrabalho, pois o supervisório só seria mostrado ao cliente no momento da implantação. A escolha de um modelo ágil não proporcionaria a organização necessária da documentação de forma a permitir o acompanhamento dos gerentes durante o desenvolvimento, e nem sempre os clientes estão dispostos a avaliar versões intermediárias. Diante disso, um modelo que combine diversas características pode ser mais eficiente para o desenvolvimento de supervisórios.

3.1 Descrição do Modelo de Desenvolvimento de um Supervisório

Segundo Vianna (2008), é necessário elaborar um planejamento detalhado antes da escolha de um ambiente de desenvolvimento de supervisório para que a adoção do mesmo seja a melhor possível.

Alguns critérios de construção de interfaces humano-computador, presentes na norma NBR 9241-10 (2000), são interessantes de serem aplicados no desenvolvimento de supervisórios, e são listados a seguir:

- Mensagens devem ser claras, explicitas e autossuficientes;
- Convém que as respostas ou explicações do sistema sejam apresentadas em uma terminologia consistente e adequadas ao nível de conhecimento do usuário;
- Convém que o usuário seja informado sobre as mudanças no estado do sistema;
- Mensagens usadas para tarefas similares devem ser similares, de modo que o usuário possa desenvolver procedimentos comuns para resolver estas tarefas;
- Os erros devem ser explicados para ajudar o usuário a corrigi-los;
- Necessidades e características do usuário podem requerer que situações de erros sejam postergadas, deixando ao usuário a decisão de quando tratá-las.

Os programadores podem elaborar aplicativos de supervisão seguindo uma sequência de implementação detalhada que pode ter variações quanto ao tipo e finalidade de aplicação, grau de instrução do operador, nível hierárquico do sistema, tipo de controle realizado (monitoramento, supervisão ou controle propriamente dito), localização do processo a ser supervisionado, entre outros (ALBUQUERQUE e ALEXANDRIA, 2009).

Os sistemas supervisórios devem ser desenvolvidos com base em regras claras. Por esse motivo, a elaboração de um modelo consistente, que aborde os passos a serem

seguidos no desenvolvimento, pode servir de guia para profissionais da área que desejam praticidade e eficiência em seus projetos.

A proposição desse modelo é baseada no estudo do processo de desenvolvimento de supervisórios na indústria e dos métodos de desenvolvimento de *software*. O modelo é ilustrado na figura 16 e é descrito por um fluxograma que detalha as fases de planejamento, desenvolvimento e implantação de um supervisório.

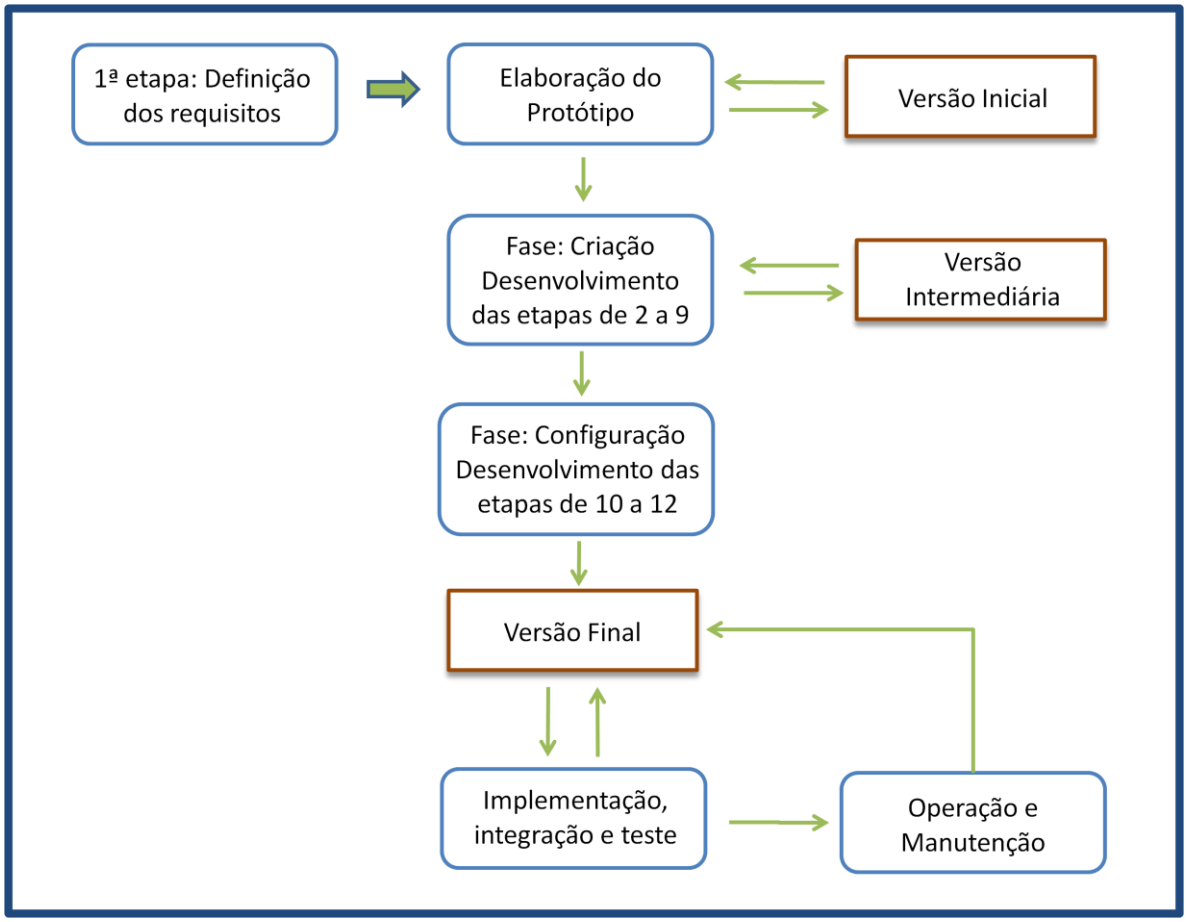


Figura 16 - Fluxograma de Metodologia de Desenvolvimento de Supervisório

O processo inicia-se com a definição dos requisitos fundamentais para o projeto do supervisório. Esta fase é descrita detalhadamente na seção 3.1.1.

Após a identificação dos requisitos, é necessária a elaboração de um protótipo que mostre os elementos principais em uma versão inicial do supervisório ao cliente. Esse protótipo deve conter as bases das nove primeiras etapas de desenvolvimento para que o cliente possa avaliar se o *software* atenderá suas necessidades. Esta prática tem como objetivo validar os requisitos o mais cedo possível, evitando o retrabalho após muitas fases de desenvolvimento.

A exposição do protótipo pode ser realizada através de um vídeo que mostra a operação do *software* supervisório. Desta forma, o cliente verificará de forma dinâmica e rápida o estado atual do *software*, e a documentação detalhada da operação não necessita ser entregue imediatamente, economizando o esforço, diminuindo os custos iniciais do projeto e também a necessidade de reedição dos documentos.

Após a fase de prototipação, o programador deve executar a fase de criação, que é constituída pelas etapas 2 até a 9, de forma a deixar o aplicativo bem estruturado e organizado para que o cliente possa avaliar essa versão intermediária e fornecer um parecer com sugestões de alterações e identificação dos pontos que estão atendendo aos requisitos pré-estabelecidos. O produto da versão intermediária desta fase é o supervisório com todos os objetos, telas e animações.

A versão intermediária não contempla as etapas 10 a 12 porque elas são geralmente realizadas próximo do momento de implantação, integração do *hardware* com o *software* e teste. Essa atitude justifica-se também pela ideia de mostrar a versão intermediária ao cliente o mais cedo possível para sanar possíveis erros. Novamente esta versão pode ser mostrada na forma de um vídeo de operação do supervisório.

A fase de configuração (Desenvolvimento das etapas 10 até 12) só deve ser iniciada quando a anterior for concluída para que o desenvolvedor receba o *feedback* do cliente e possa começar uma nova fase com mais informações. Esse procedimento proporciona organização no desenvolvimento do *software*.

As etapas 10, 11 e 12 são descritas nas subseções deste capítulo, e, quando forem concluídas, o programador deve mostrar a versão final do supervisório ao cliente para que, sendo aprovada, possa dar início a implantação, integração dos componentes de hardware ao software e teste do supervisório em campo para verificar a eficiência do *software* em funcionamento. Neste momento, eventuais problemas que apareçam devem ser sanados o mais breve possível para que o *software* possa logo ser testado por um longo prazo.

Após a correção de eventuais erros, o desenvolvedor pode expor novamente a versão final do supervisório ao cliente para que se proceda aos ajustes finais e à fase de operação do *software* (feita pelos usuários). Observa-se no fluxograma que essas duas últimas fases podem gerar ajustes e manutenções para que a versão final seja reeditada e atenda, completamente, aos requisitos preestabelecidos. Na fase de operação e manutenção, o aplicativo permanece em execução, fazendo exatamente todas as tarefas previstas no projeto.

As etapas de desenvolvimento de um supervisório citadas no fluxograma, essenciais para o planejamento de um *software* de supervisão, podem ser enumeradas da seguinte forma:

1 – Conhecer e documentar a descrição completa do processo a ser supervisionado;
2 – Identificar e configurar as variáveis e os *tags* (entradas e saídas);
3 – Escolher o ambiente de desenvolvimento do *software* de supervisão;
4 – Planejar históricos, relatórios e banco de dados;
5 – Elaborar e definir o *layout* das telas;
6 – Definir a quantidade, hierarquia e navegação das telas;
7 – Identificar e configurar os alarmes necessários;
8 – Elaborar *scripts* e animações que simulam a dinâmica do processo;
9 – Elaborar gráficos de tendências;
10 – Configurar a infraestrutura de comunicação entre *hardware* e *software*;
11 – Elaborar sistemas de controle de acesso;
12 – Elaborar a documentação do sistema desenvolvido.

O conjunto de etapas de desenvolvimento mais conhecido para o desenvolvimento de supervisórios está descrito em Moraes e Castrucci (2007). O modelo proposto neste trabalho abrange a maioria das etapas propostas na citada literatura e adiciona outros aspectos importantes, tais como a escolha do ambiente de desenvolvimento de supervisórios e as ações enumeradas nos itens 5, 8, 10, 11 e 12 desta seção. O uso de fluxogramas, organização das variáveis e alarmes por tabelas e descrição da funcionalidade dos relatórios também são contribuições do modelo proposto.

Nesta pesquisa, todos os passos listados foram executados pelos programadores, de modo a permitir a avaliação e a validação do modelo.

3.1.1 Descrição completa do processo

O primeiro contato do programador com o processo a ser supervisionado deve ser o mais detalhado possível, pois a noção do sistema como um todo facilita muito o entendimento do projeto e o planejamento de uma possível solução para aquele problema. O programador deve documentar todas as informações a respeito do processo, que podem ser adquiridas através dos:

- operadores do processo (se já existir um sistema automatizado em operação), que acompanham diariamente o funcionamento do mesmo e geralmente conhecem os principais problemas associados ao sistema físico e à falta de supervisão da planta;
- gerentes ou supervisores, do setor onde se localiza a planta, que podem indicar os elementos de segurança necessários, a forma de controle de acesso, a disponibilidade dos dados nas telas, a maneira de representação da dinâmica do processo e dos seus elementos e as possibilidades de expansão da planta automatizada;
- empresários que contrataram o serviço, pois eles podem contribuir com informações gerenciais, análises do custo-benefício do supervisório a ser implantado e vantagens que almejam com o desenvolvimento do aplicativo.

Todos os usuários do sistema devem ser identificados e envolvidos no projeto de forma a fornecerem dados relevantes e avaliarem soluções. O envolvimento e participação dos usuários aumenta a satisfação do cliente e minimiza o risco da aplicação não corresponder às necessidades deles, pois o *feedback* quanto à qualidade do *software* é uma fonte de informação essencial no projeto, inclusive no uso operacional, com a identificação de problemas a longo prazo que fornece dados para projetos futuros (PROJETO ISO, 2011).

Os programadores devem adquirir a competência de escrever e reportar todas as características do processo para que o projeto esteja bem detalhado. O mnemônico ORAR: Observar, Registrar, Analisar e Reportar é uma sequência de passos seguida pelos programadores para sistematizar e destacar a importância do processo de documentação. Esta prática pode ser realizada através do agrupamento de todos os documentos e anotações em uma pasta denominada PAPO – Pasta de Acompanhamento de Projeto e Operação (POLITO-BRAGA et al, 2012).

A descrição completa do processo pode ser melhor representada através de um fluxograma que organizará o problema de controle em uma sequência de atividades ou fases necessárias para o funcionamento correto e completo do sistema. As informações coletadas de segurança, alarmes, gráficos, dados e usuários, quando associadas às fases do fluxograma elaborado, proporcionam um entendimento global do projeto a ser desenvolvido e estimulam a identificação e análise das soluções e ferramentas que poderão ser utilizadas nos supervisórios.

3.1.2 Identificar e configurar as variáveis e os *tags* (Entradas e saídas)

Após a documentação e entendimento da descrição do processo completo é possível realizar um levantamento da quantidade necessária de entradas e saídas físicas para a execução do projeto de supervisão. Um processo que já está automatizado, geralmente possui a quantidade de saídas e entradas documentadas nos manuais, no entanto, um projeto completo de automação de um sistema necessita da enumeração dessas variáveis de modo a facilitar a programação da lógica de controle e do sistema de supervisão a serem desenvolvidos.

Segundo Moraes e Castrucci (2007), o programador deve ter em mente um limite superior para o número de dados. Um grande tráfego na comunicação pode prejudicar o desempenho total (velocidade e integridade de informação). Em sistemas com grande número de variáveis, a contribuição da taxa de atualização das variáveis analógicas (tempo de *scan*) no desempenho geral da comunicação costuma ser o primeiro parâmetro verificado na rede industrial.

A comunicação entre um sistema supervisório e um CLP é realizada utilizando endereços de memória do próprio CLP. Cada variável é acompanhada por um *tag* (etiqueta). Este *tag* é utilizado em diversos locais: nas listas de *tags* do supervisório; na lista de materiais; nos desenhos e diagramas do projeto; nas animações; entre outros (MORAES e CASTRUCCI, 2007).

Os *tags* são todas as variáveis numéricas ou alfanuméricas envolvidas na aplicação podendo executar funções computacionais (operações matemáticas, lógicas, com vetores ou *strings*, etc.) ou representar pontos de entrada e saída de dados do processo que está sendo supervisionado (MARTINS, 2012).

A organização dessas variáveis pode ser bem realizada com a utilização de tabelas de entradas e saídas, que mostram a funcionalidade, o endereçamento e o tipo de sensor ou atuador utilizado de cada variável. Esta sugestão é uma proposição deste trabalho. As tabelas 3 e 4 mostram exemplos de como organizar as variáveis de entrada e saída.

A organização das variáveis em tabelas facilita muito a compreensão dos programadores e serve de consulta no momento do desenvolvimento da programação. A associação entre o endereçamento das variáveis no *software* supervisório e no CLP é de extrema importância para a configuração da comunicação entre ambos, pois é preciso considerar que geralmente um profissional desenvolve o programa do CLP e outro o do

supervisório. Portanto, a elaboração dessas tabelas é necessária para que os programadores possam fazer as associações corretas das variáveis no momento da integração do controlador com o supervisório.

Tabela 3 - Organização das variáveis de entrada.

Endereçamento no CLP	**Mnemônicos dos *tags***	**Tipo de sensor**	**Funcionalidade**
I:0.0	Intrusão	Sensor de fim de curso	Sensor digital que detecta intrusão no quadro de automação
I:0.1	Contador	Sensor óptico de presença	Sensor digital utilizado para contagem de recipientes
IW:1.0	Volume	Sensor de nível	Sensor analógico que detecta o volume de líquido dentro do recipiente
IW:2.0	Vazão	Sensor de vazão	Sensor analógico que mede a vazão de líquido que entra no recipiente

Tabela 4 - Organização das variáveis de saída.

Endereçamento no CLP	**Mnemônicos dos *tags***	**Tipo de atuador**	**Funcionalidade**
Q:0.2	Avançocilindro	Válvula eletropneumática	Saída digital que aciona o avanço do cilindro pneumático
Q:0.3	Recuocilindro	Válvula eletropneumática	Saída digital que aciona o recuo do cilindro pneumático
QW:1.0	Vazaosaida	Válvula proporcional de vazão	Saída analógica que controla a vazão de líquido para o recipiente
QW:2.0	Pressaosaida	Válvula proporcional de pressão	Saída analógica que controla a pressão do líquido que vai para o recipiente

A criação dos *tags* que receberão os sinais das entradas e saídas do processo, em intervalos de tempo definidos pelo programador (tempo de *scan*), pode ser realizada com mnemônicos, como mostrado nas tabelas 3 e 4, que indicam a funcionalidade do mesmo. Geralmente os programadores, para facilitar a manutenção, também organizam os *tags* em grupos de acordo com o tipo, ou utilização em cada tela, ou funcionalidade.

3.1.3 Escolha do ambiente de desenvolvimento do supervisório

Atualmente, existem muitos fornecedores de ambiente de desenvolvimento de *softwares* de supervisão para sistemas SCADA. Muitos deles são dedicados, ou seja, ambientes específicos para determinados equipamentos, porém existe uma grande parte deles que são de uso geral, extremamente flexível, permitindo a elaboração de aplicativos para os mais diversos fins. Os *softwares* SCADA disponibilizam diversos recursos prontos, que são utilizados normalmente por um aplicativo de supervisão em automação (ALBUQUERQUE e ALEXANDRIA, 2009).

A escolha de qual ambiente utilizar depende de uma série de fatores listados a seguir:

- Finalidades às quais o *software* se destina: Dependendo da aplicação pode-se escolher um supervisório específico como ELIPSE POWER (utilizado em sistemas elétricos), LAB VIEW (aplicações para laboratórios) ou supervisórios genéricos.
- Recursos disponíveis: Os recursos de cada supervisório também podem ser fatores de decisão, pois algumas aplicações precisam de interface OPC, conexão com a *internet*, conexão com câmeras de monitoramento, possibilidade de configuração de diversos servidores e clientes, monitoração remota, etc. Nem todos os supervisórios possuem estas funcionalidades, portanto é preciso levar estes aspectos em consideração no momento da especificação do *software*.
- Experiência de utilização do *software*: Este é um dos fatores mais preponderantes para a escolha do supervisório, pois o programador vai preferir vender a solução com a qual possui mais experiência, considerando que a eficiência e celeridade de desenvolvimento de um aplicativo são diretamente proporcionais à experiência do programador com o *software*.
- Relação custo-benefício: Eis um dos fatores que deve ser decidido em conjunto (empresários, programadores e clientes), pois os fornecedores de *software* possuem diversos tipos de pacotes com limitações e preços diferentes. Há de se averiguar qual é a solução mais adequada para os clientes e como ela pode influenciar no custo da mão de obra.
- Preferência do cliente: Em diversos casos o cliente já conhece um determinado ambiente de desenvolvimento de supervisório porque o viu em alguma indústria ou em um congresso e aprovou suas funcionalidades e recursos. Desta forma, o cliente prefere a

utilização de um sistema que já conhece e solicita que a solução seja concebida com um ambiente específico.

3.1.4 Planejamento de históricos, relatórios e banco de dados

Alguns dados de um sistema supervisório são guardados em instantes determinados de tempo com o propósito de realização de análises do comportamento do processo, de auditorias internas e externas, de investigação das causas de possíveis acidentes, etc. Esses dados são valores das entradas e saídas, valores de variáveis auxiliares, eventos de disparo, reconhecimento e saída de alarmes, outros eventos importantes, etc.

A maioria dos ambientes de desenvolvimento de supervisórios possui conexão com os bancos de dados mais utilizados e/ou possui tipos de arquivos específicos que gravam as informações, como é o caso do Elipse SCADA, que armazena dados em arquivos com extensão ".dat". Esses dados são gravados em intervalos de tempo definidos pelo programador com data, hora e nome do usuário que está acessando, para que possam ser identificados com celeridade.

Os bancos de dados são criados com suas respectivas tabelas e a conexão com o ambiente de desenvolvimento precisa ser configurada para o estabelecimento da comunicação entre ambos. O programador cria uma nova conexão com banco de dados e escolhe o tipo de conexão e o tipo de arquivo que vai fazer o armazenamento. A maioria dos supervisórios pode realizar uma conexão ODBC ou uma conexão ISAM, dando suporte aos arquivos de banco de dados com extensão ".mdb", ".xls", ".dbf", ".mdx", ".accdb", etc.

Os relatórios são utilizados para filtrar os dados de forma a disponibilizá-los visualmente na tela ou impressos de acordo com a necessidade de quem precisa analisá-los. Os relatórios devem possuir os filtros de data e hora para o início e o fim da geração, e, após o operador configurar esses parâmetros, ele pode ter a opção de configurar a impressão dos documentos.

3.1.5 Elaboração e definição do layout das telas

As diversas telas de um supervisório são janelas para o monitoramento do processo, em que são inseridos os objetos para compor a interface do operador com o sistema.

Os objetos são elementos gráficos relacionados com os *tags* de forma a realizar uma interface amigável com o processo.

Uma tela da aplicação pode conter figuras de fundo e objetos. Para a elaboração das telas pode-se importar figuras criadas em qualquer editor gráfico (CorelDraw, Autocad, Paint). Alguns ambientes de desenvolvimento de supervisórios já disponibilizam bibliotecas com figuras diversas (ALBUQUERQUE e ALEXANDRIA, 2009).

A boa elaboração e organização de um supervisório dependem do desenvolvimento das telas, pois uma interface humano-máquina deve combinar a apresentação das telas com a correta funcionalidade dos recursos inseridos nela, de forma a facilitar ao máximo o entendimento do processo por parte do operador.

As telas de um supervisório devem representar sinteticamente o processo, procurando chamar a atenção do operador através do destaque dos elementos mais importantes. Vale lembrar que a ausência ou o excesso de informações nas telas são características indesejadas que provocam o desinteresse e a falta de atenção, respectivamente.

O programador deve lembrar que os operadores são levados ao desinteresse quando os sinópticos (representação gráfica resumida do processo) são pouco representativos do processo e sem atrações de animação ou quando são muito cheios, tornando difícil o processamento da informação. É sabido também que o nível alto do som da buzina de alarmes e o exagero na quantidade de elementos piscantes geram cansaço aos operadores (VIANNA, 2008).

Uma boa prática para os programadores é criar um padrão de aparência de telas, destinando áreas na tela para elementos repetitivos como menus, alarmes, processos, históricos, entre outros, pois os operadores ficarão habituados a localizá-los. Alguns ambientes de desenvolvimento de supervisórios já possuem ferramentas para fazer essa atividade rapidamente para o programador. A utilização de cores distintas para essas áreas também contribui para uma melhor distinção por parte do operador.

3.1.6 Definição da quantidade, hierarquia e navegação das telas

Segundo Moraes e Castrucci (2007), a hierarquia de navegação consiste em uma série de telas que fornecem progressivamente detalhes das plantas e seus constituintes à medida que se navega através do aplicativo. A boa estratégia de organização torna o sistema claro e consistente com a realidade.

A quantidade de telas é definida conforme a necessidade de detalhamento do processo, porém existem algumas telas, descritas a seguir, necessárias para o desenvolvimento de um supervisório com organização e qualidade.

Para que a navegação das telas seja bem planejada, é necessário dar especial atenção à distribuição das mesmas em nível de acesso segundo a função principal a que se destinam (MORAES e CASTRUCCI, 2007). Portanto, as telas podem se dividir em grupos que possuem finalidades diferentes. Esses grupos, neste trabalho, são definidos como:

- Telas de Apresentação e Menu;
- Telas de Armazenamento de dados (Alarmes, Relatórios, Históricos e Receitas);
- Telas de Processos e de Gráficos;
- Telas de Detalhe.

As **Telas de Apresentação e de Menu** são as *interfaces* de inicialização e visão geral do sistema, respectivamente. Nessas telas, os logotipos das empresas e imagens dos processos são geralmente utilizados como plano de fundo, de forma a destacar a imagem empresarial. A figura 17 ilustra um exemplo de tela de apresentação.

A tela de apresentação é essencial para que o operador possa entrar com seu *login* e senha de modo a acessar o supervisório com as suas permissões. A tela de menu é responsável por disponibilizar os links que direcionam o operador para todas as telas e para fazer *logoff* no sistema.

As telas de menu podem ser disponibilizadas em uma área específica de todas as telas, como é ilustrado à esquerda da figura 18, ou pode ser uma tela específica do supervisório que direciona o usuário para todas as outras telas, como é mostrado na figura 19.

Figura 17 - Exemplo de tela de apresentação

Fonte: ELIPSE, 2012

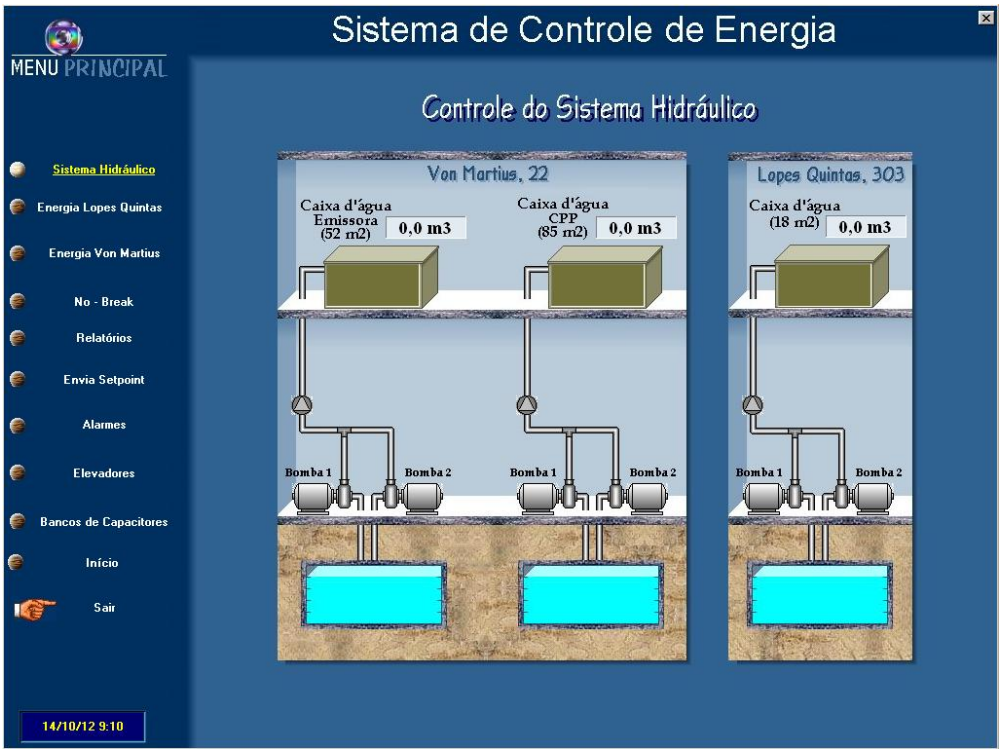


Figura 18 - Exemplo de tela de menu

Fonte: ELIPSE, 2012

Figura 19 - Exemplo de tela de menu específica.

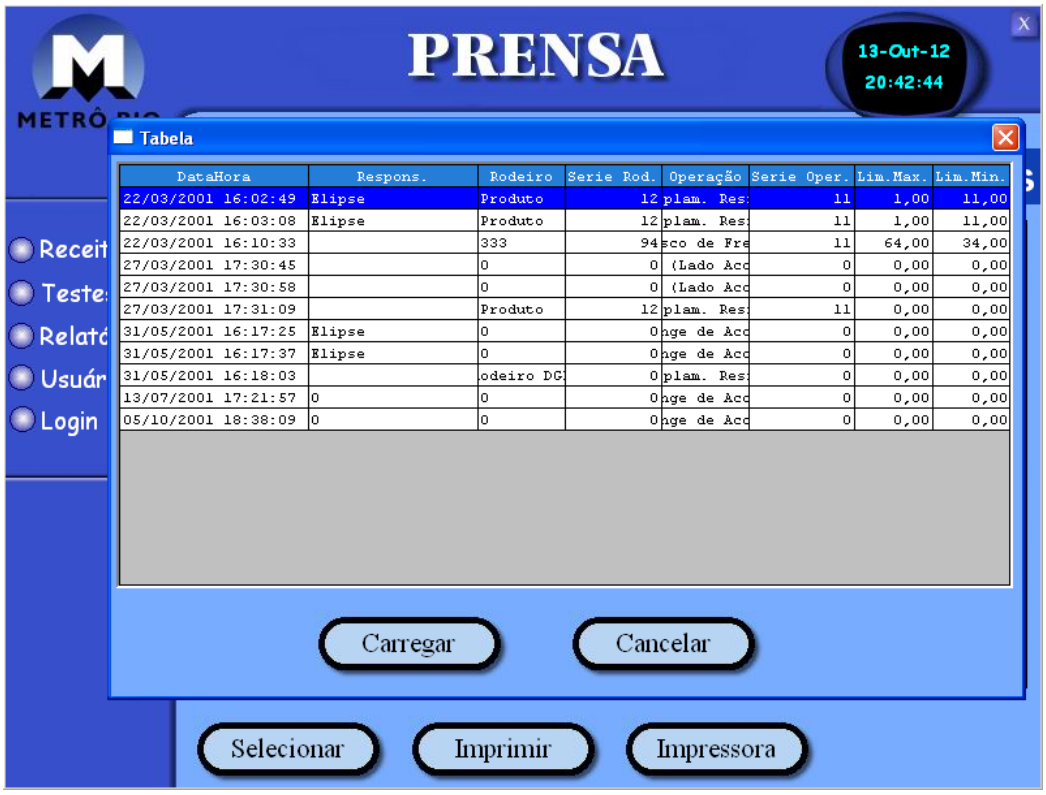


Figura 20 - Exemplo de tela de Históricos

Fonte: ELIPSE, 2012

As **Telas de armazenamento de dados** são compostas pelas telas de históricos, alarmes, relatórios e receitas. Estas telas disponibilizam aos usuários as informações relativas ao processo, tais como eventos, alarmes, variáveis auxiliares, de entradas e de saídas, mensagens de avisos, entre outros.

As telas de Históricos disponibilizam visualmente ao usuário as variáveis do processo, como ilustrado na figura 20. As telas de alarmes mostram ao usuário o histórico de todos os eventos de alarmes, como mostrado na figura 21.

dd/mm/yyyy hh:mm:ss	Tipo	EstadoAlm	Comentário	NomeTag	Valor
13/10/2012 20:33:31	RET		F-450 B - Retornou ao Nível Normal	expTk02	9053,29
13/10/2012 20:33:31	RET		F-405 - Retornou ao Nível Normal	expTk15	5384,98
13/10/2012 20:33:30	RET		F-430 - Retornou ao Nível Normal	expTk07	2023,67
13/10/2012 20:33:30	RET		F-432 - Retornou ao Nível Normal	expTk08	2530,73
13/10/2012 20:33:30	RET		F-450 A - Retornou ao Nível Normal	expTk01	2707,41
13/10/2012 20:33:30	RET		F-455 - Retornou ao Nível Normal	expTk10	1541,33
13/10/2012 20:33:30	RET		F-440 - Retornou ao Nível Normal	expTk09	2589,95
13/10/2012 20:33:30	LOW	UNACK	F-405 - Atingiu Nível Baixo	expTk15	2165,02
13/10/2012 20:33:30	HIGH	UNACK	F-404 - Atingiu Nível Alto	expTk12	34449,72
13/10/2012 20:33:30	LOW	UNACK	F-455 - Atingiu Nível Baixo	expTk10	1336,54
13/10/2012 20:33:30	LOW	UNACK	F-440 - Atingiu Nível Baixo	expTk09	2358,23
13/10/2012 20:33:30	LOW	UNACK	F-432 - Atingiu Nível Baixo	expTk08	2059,76
13/10/2012 20:33:30	LOW	UNACK	F-430 - Atingiu Nível Baixo	expTk07	1000,32
13/10/2012 20:33:30	HIGH	UNACK	F-601 B - Atingiu Nível Alto	expTk05	24389,14
13/10/2012 20:33:29	HIGH	UNACK	F-601 A - Atingiu Nível Alto	expTk04	22201,02
13/10/2012 20:33:29	HIGH	UNACK	F-450 C - Atingiu Nível Alto	expTk03	20130,34
13/10/2012 20:33:29	LOW	UNACK	F-450 B - Atingiu Nível Baixo	expTk02	2274,61
13/10/2012 20:33:29	LOW	UNACK	F-450 A - Atingiu Nível Baixo	expTk01	2033,25
05/04/2002 14:45:56	RET		F-404 - Retornou ao Nível Normal	expTk12	32414,81
05/04/2002 14:45:56	RET		F-601 B - Retornou ao Nível Normal	expTk05	15845,27
05/04/2002 14:45:56	RET		F-601 A - Retornou ao Nível Normal	expTk04	6131,96
05/04/2002 14:45:56	RET		F-450 C - Retornou ao Nível Normal	expTk03	9084,39
05/04/2002 14:45:55	RET		F-450 B - Retornou ao Nível Normal	expTk02	8869,34
05/04/2002 14:45:55	LOW	UNACK	F-450 B - Atingiu Nível Baixo	expTk02	8632,34
05/04/2002 14:45:55	RET		F-450 B - Retornou ao Nível Normal	expTk02	8874,85
05/04/2002 14:45:53	RET		F-405 - Retornou ao Nível Normal	expTk15	5275,85
05/04/2002 14:45:53	RET		F-430 - Retornou ao Nível Normal	expTk07	2171,61
05/04/2002 14:45:53	RET		F-432 - Retornou ao Nível Normal	expTk08	2588,48

Figura 21 - Exemplo de tela de Alarmes

Fonte: ELIPSE, 2012

As telas de relatórios, geralmente janeladas (menor tamanho que as telas normais), possuem os campos de filtro de históricos para realizar impressões desses dados, como ilustrado na figura 22.

O sinóptico dessas telas é constituído basicamente de tabelas de dados e uma pequena área reservada para o menu e para a impressão dos relatórios.

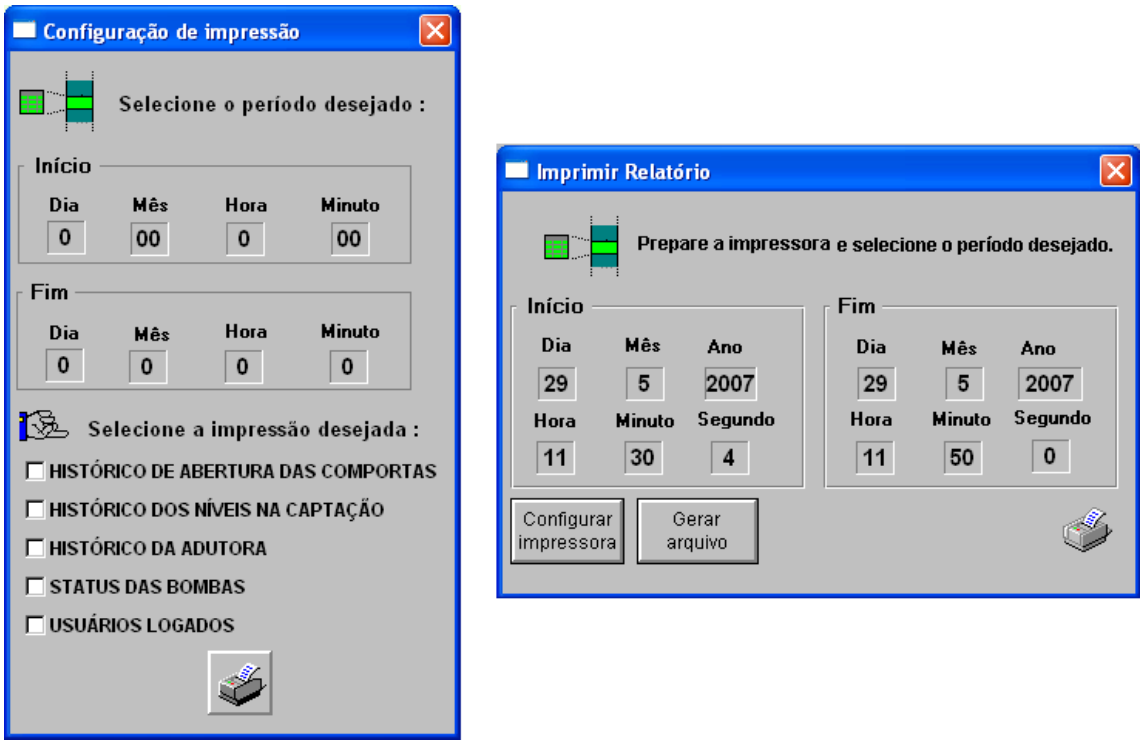


Figura 22 - Exemplo de telas de Relatórios

As telas de receitas carregam o supervisório com dados de inicialização do sistema ou informações para processos em bateladas. Um exemplo de tela de receita é ilustrado na figura 23.

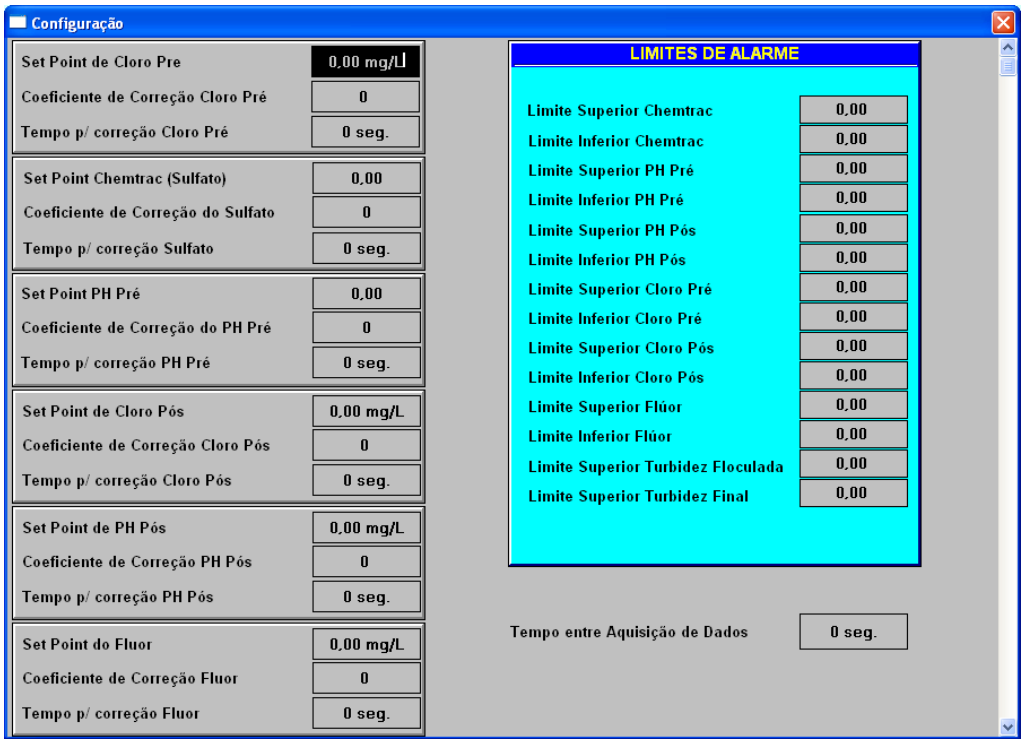


Figura 23 - Exemplo de tela de Receitas

Fonte: ELIPSE, 2012

As **Telas de Processos e de Gráficos** constituem os elementos principais do supervisório, pois elas provêm a interface entre o processo e o operador. Portanto, elas precisam ser projetadas e desenvolvidas com a maior participação possível do cliente, para atender à sua preferência.

As telas de processos disponibilizam para o usuário toda a infraestrutura de representação da dinâmica do processo através de figuras, animações, *displays*, indicadores e objetos. Todos esses elementos devem ser o mais parecido possível com a realidade para que o operador tenha facilidade de identificá-los na planta real. Um exemplo de tela de processo é ilustrado na figura 24.

Figura 24 - Exemplo de tela de Processo

Fonte: ROCKWELL, 2013

As **Telas de Detalhe** são janelas abertas a partir de cliques em objetos da tela como botões, figuras, *displays*, entre outros. Elas mostram detalhes de um determinado objeto como motores, atuadores, sensores, esteiras, tanques, etc. As janelas devem mostrar detalhadamente o *status*, as características e configurações do objeto em foco.

O exemplo da figura 25 ilustra a tela de detalhe de um Transmissor de vazão de código FTQ 002. Essa tela é mostrada quando o usuário clica com o mouse no ícone da planta em diagrama ISA.

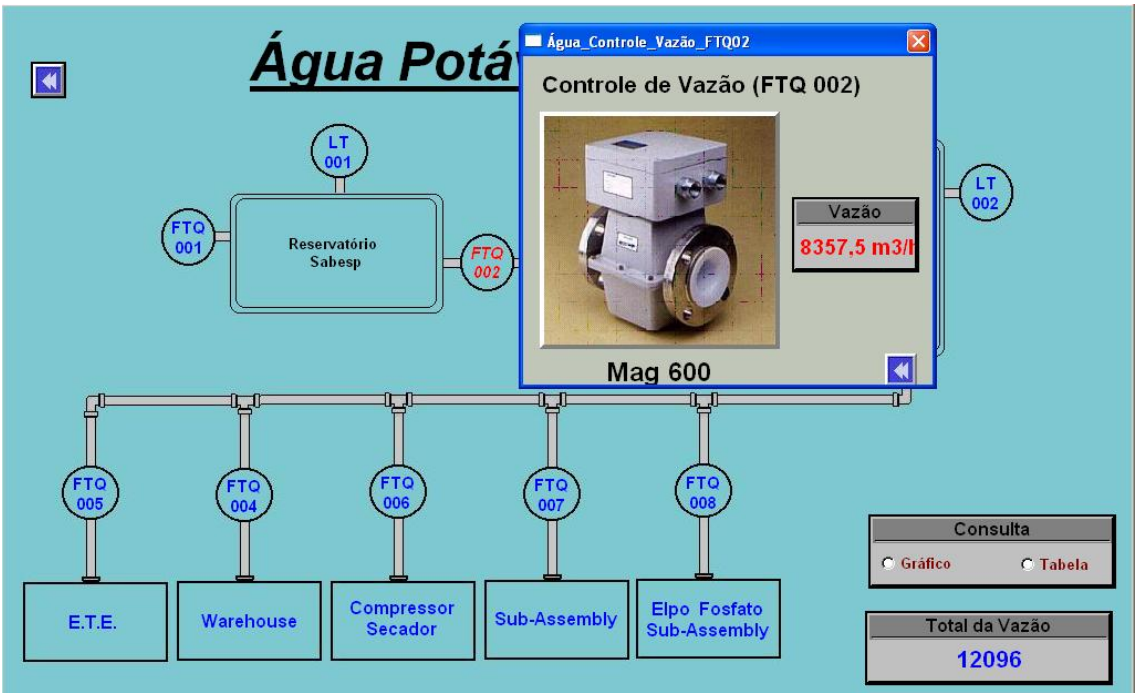


Figura 25 - Exemplo de tela de Detalhe

Fonte: ELIPSE, 2012

3.1.7 Identificação e configuração dos alarmes necessários

A necessidade de identificação de alarmes no sistema geralmente é indicada pelos supervisores ou especialistas no processo a ser supervisionado, pois eles sabem os pontos críticos que precisam ser alarmados caso aconteça algum problema. Independente da opinião dos especialistas, um estudo do processo deve ser feito, levando em consideração a segurança do sistema quando houver problemas em sensores e atuadores.

Segundo Moraes e Castrucci (2007), algumas características importantes de alarmes devem ser observadas, tais como:

- Escolha e notificação de operadores;
- Envio de mensagens;
- Providência de ações;
- Chamar a atenção do operador para uma modificação do estado do processo;
- Sinalizar um objeto atingido;
- Fornecer indicação global sobre o estado do processo.

Após a identificação de quais serão os alarmes do processo, os mesmos devem ser divididos por níveis de segurança e prioridade, de preferência com a organização de uma tabela que descreva quais são os alarmes mais críticos do sistema e como o operador deve reconhecer esses alarmes e tomar as providências necessárias para sanar o problema. A tabela 5 mostra um exemplo de elaboração da tabela de alarmes que pode ser disponibilizado para o operador no próprio supervisório, de forma a guiá-lo principalmente em momentos de apreensão.

A organização desses alarmes em tabelas permite também uma facilidade maior em configurá-los no *software* supervisório e auxilia os futuros programadores que, porventura, precisem fazer manutenção nesse *software*. Esta sugestão é uma proposta deste trabalho.

Tabela 5. Organização dos alarmes.

Alarme	**Prioridade**	**Reconhecimento**	**Solução**
Nível máximo excedido	100%	Imediato	Fechar válvula de entrada d'água e verificar se o sensor está medindo corretamente
Temperatura elevada	80%	Imediato	Verificar se o sensor está medindo corretamente e acionar equipe de manutenção.
Válvulas com defeito	50%	Diário	Verificar qual é o defeito e, se necessário, trocar as válvulas
Nível baixo	40%	Diário	Verificar se a medição do sensor está correta.

Observa-se na tabela 5 que os níveis de prioridade estão em porcentagens para que o alarme que precise ser reconhecido imediatamente (100% e 80% no exemplo) possa ser visualizado mais rapidamente pelo usuário com o objetivo de acelerar a execução das providências necessárias. Desta forma, os alarmes menos prioritários podem ser sanados ao longo do dia, após o atendimento aos mais prioritários.

Sendo assim, o operador terá uma facilidade maior em saber de imediato qual solução tomar em emergências, pois a descrição das atividades necessárias com a ocorrência do alarme está disponível no próprio *software* supervisório.

A notificação dos operadores pode ser também diferenciada conforme o nível de emergência inserido, como por exemplo, mensagem de erros na tela do supervisório, ou sinal

sonoro ativado até reconhecimento do operador, ou direcionamento da atenção do operador para a tela onde está representado o setor problemático do processo. Essa última medida é geralmente configurada para alarmes de emergência que mostram que a segurança do sistema está gravemente afetada.

A configuração de todos os alarmes deve ser realizada com base em todas as informações do processo discutidas e aprovadas com os especialistas para que exista o mínimo de retrabalho possível. Lembrando que todos os eventos de alarmes (acontecimento, reconhecimento e saída do alarme) devem ser registrados e armazenados em históricos ou banco de dados com a finalidade de facilitar o trabalho de futuras auditorias.

De acordo com Moraes e Castrucci (2007), o planejamento de um questionário prévio para tentar detectar pontos críticos dentro do processo é recomendável. As seguintes perguntas podem ser feitas:

- Quais as configurações de alarmes que impõem ao operador dirigir-se a um grupo de indicadores a fim de julgar o estado real do processo?
- Como conceber o aparecimento do alarme para que este guie o operador em direção ao grupo de indicadores envolvidos?
- Como agrupar os indicadores para que o operador possa facilmente realizar uma representação do estado do processo, podendo escolher entre diversas opções?

3.1.8 Elaboração dos scripts e das animações que simulam a dinâmica do processo

As animações devem ser planejadas e desenvolvidas conforme a observação da dinâmica do processo. Logo, os programadores devem perguntar aos operadores, supervisores, gerentes, clientes em geral, quais são todos os detalhes de funcionamento ou especificá-los, caso o projeto inclua a criação do processo desejado.

Como já foi explanado anteriormente, a representação do processo deve ser o mais parecida possível com a realidade. Deve-se manter a consistência das figuras representativas e cores de destaque de forma a realçar os elementos mais importantes do sistema.

Para representar nas animações os sensores, atuadores e controladores da planta, podem-se utilizar os símbolos da ISA (*Instrument Society of American*), conhecidos internacionalmente. Assim sendo, a maioria dos profissionais da área reconhecerão com facilidade os elementos importantes do projeto. Uma planta que serve de exemplo de utilização dos símbolos da ISA é ilustrada na figura 26.

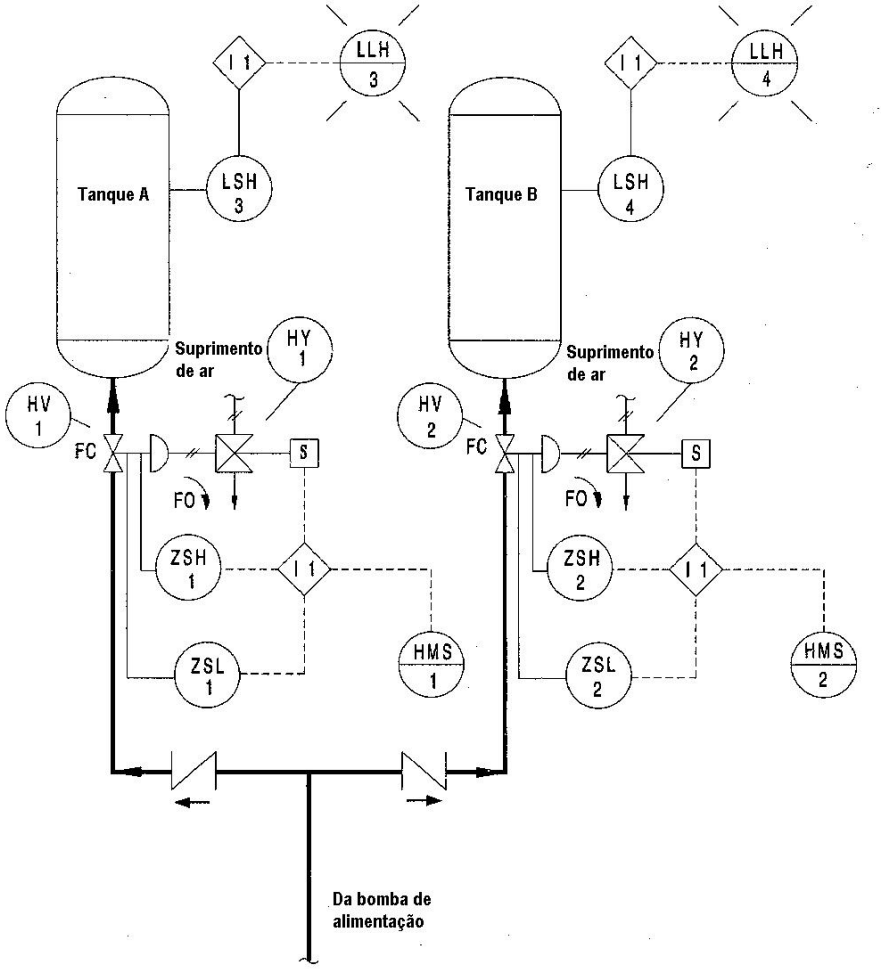


Figura 26 - Exemplo de planta com símbolos da ISA

Fonte: RIBEIRO, 1999

Segundo Albuquerque e Alexandria (2009), as linhas de código que permitem associar ações a eventos específicos são editadas para a criação de scripts. Cada item de um aplicativo possui uma lista de eventos específicos já previamente associados ao objeto. O programador pode determinar que, na ocorrência de um desses eventos, uma tarefa específica relacionada àquele evento pode ser executada.

Os scripts possibilitam a programação de ações para as situações ocorridas no processo e permitem uma flexibilidade muito grande aos supervisórios, pois possibilitam maior proximidade com operações de baixo nível e acesso a dispositivos e sistema operacional. Essa linguagem geralmente é proprietária, mas segue um conjunto de comandos de uma linguagem de programação conhecida, como o Basic ou o C (MARTINS, 2012).

Os eventos citados são ocorrências relacionadas a um objeto. Eles podem ser físicos, como por exemplo, alguma ação no teclado; ou se o evento vem do *mouse*, a

informação relevante seria a posição do cursor na tela. Os eventos também podem ser internos, como a mudança do valor de uma variável (*tag*) ou mudança de tela (ALBUQUERQUE e ALEXANDRIA, 2009).

3.1.9 Elaboração dos gráficos de tendências

As telas de Gráficos fornecem ao operador a visualização do valor das variáveis ao longo do tempo, possibilitando que seja feito um controle estatístico do processo (CEP) visando detalhar o planejamento de melhorias na eficiência da produção. Estes gráficos devem possuir nomenclatura, descrição dos eixos e unidades de medida. As telas podem ser organizadas com o intuito de posicionar os gráficos uniformemente, como ilustrado na figura 27.

Esse tipo de tela pode possuir vários tipos de indicadores de diferentes cores para facilitar a comparação entre dados. A plotagem dos dados pode ser obtida *on-line*, de acordo com o tempo de *scan* dos CLPs, ou podem ter origem de um histórico arquivado (VIANNA, 2008).

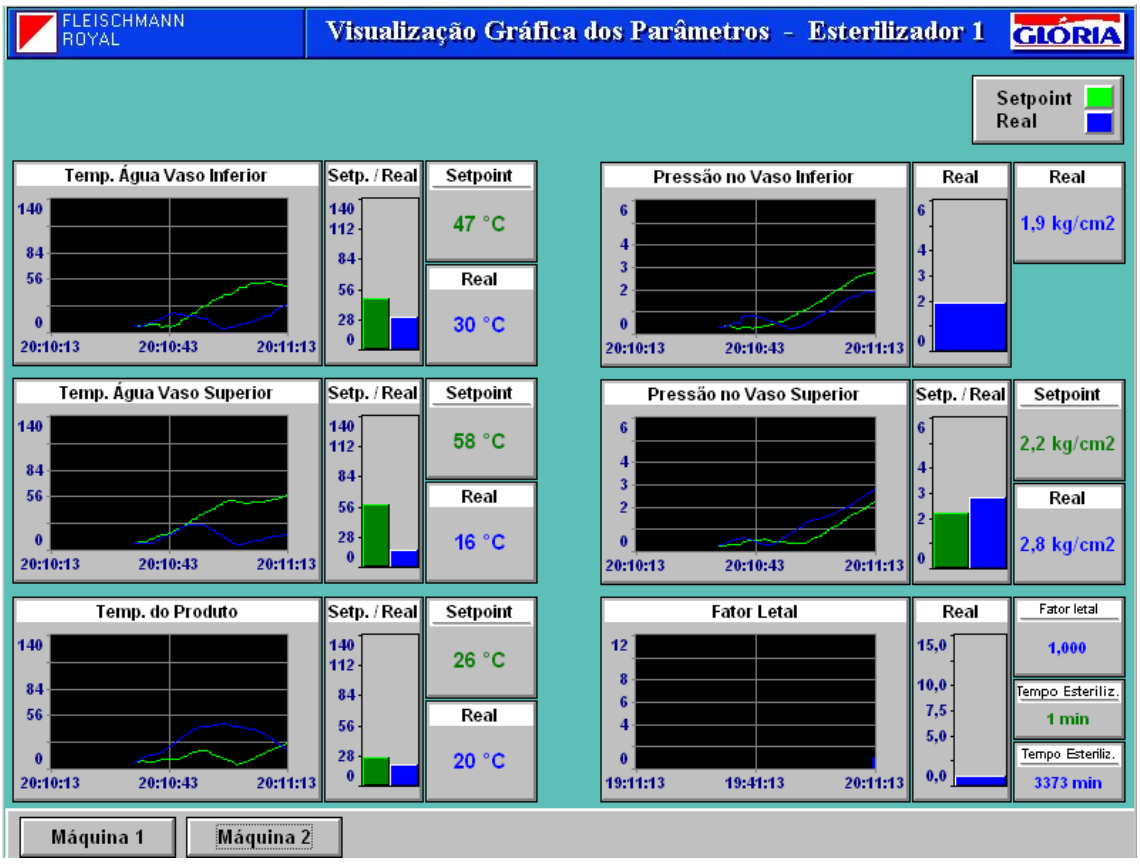


Figura 27 - Exemplo de tela de Gráficos

Fonte: ELIPSE, 2012

A maioria dos supervisórios já possui ferramentas para gerar os gráficos que serão disponibilizados no aplicativo. Cabe ao programador apenas configurá-los conforme o seu critério e inserir os *tags* que guardam o valor das variáveis nas "penas de desenho" para que o gráfico possa começar a ser gerado.

"Penas de desenho" são as ferramentas utilizadas para configurar as características dos gráficos que serão disponibilizados visualmente na tela. A maioria dos ambientes de desenvolvimento de supervisórios já possui este tipo de ferramenta como um padrão.

3.1.10 Configuração da infraestrutura de comunicação entre *hardware* e *software*

Os *softwares* de supervisão utilizam *drivers* de comunicação para "conversar" com os dispositivos da planta como controladores lógicos programáveis, sensores inteligentes, câmeras IP, *softwares* de automação em geral que suportam o protocolo OPC, entre outros.

Os *drivers* são geralmente disponibilizados para teste pelo próprio fabricante do ambiente de desenvolvimento do *software* supervisório. Eles devem ser associados aos *tags* previamente configurados e conforme indicação do manual do fabricante, através de parâmetros como protocolo e modo de comunicação, tipo de checagem de erro, número de bits de dados, velocidade (*Baud rate*), existência de *bits* de paridade e *stop bits*, endereço do CLP, *timeout*, porta de comunicação. Aconselha-se testar a comunicação antes da execução do aplicativo para corrigir possíveis erros.

Existem tipos de *tags* específicos para receber sinais externos. São os chamados *tags* de comunicação e possuem parâmetros de configuração, descritos no manual do *driver*, que devem ser corretamente preenchidos no momento da instalação para que o *tag* receba o sinal pretendido. Os parâmetros são:

- Endereço do CLP;
- Tipo de variável;
- Número do elemento inicial do arquivo;
- *Timeout*;

3.1.11 Especificação do perfil de usuários com níveis diferentes de acesso

Os operadores, que acompanham o funcionamento da planta a ser supervisionada, são cadastrados previamente no supervisório para realizarem o monitoramento do processo.

Esse cadastramento também pode ser realizado quando o aplicativo de supervisão estiver executando, desde que esta ação seja configurada pelo programador. O procedimento pode ser realizado apenas pelo administrador do supervisório, que, na maioria das vezes, é o gerente. Esta prática deve-se à busca por organização e centralização das informações.

Conforme Moraes e Castrucci (2007), quando se tem ideia do sistema de segurança, devem ser consideradas as seguintes questões: a quem o acesso deve ser restrito? O acesso será restrito por áreas do processo? Nessas condições, o sistema de segurança permite:

- somar, mudar ou desabilitar contas individuais de usuários ou grupos de operadores;
- restringir o acesso aos comandos e telas específicas do supervisório;
- fornecer proteção de telas escritas para determinados *tags*.

O próprio gerente pode estabelecer quais são os níveis de acesso para cada usuário e quais serão os recursos de acesso restrito do supervisório, que são reservados, por segurança, a um administrador. De acordo com as informações recebidas, o programador pode fazer as configurações necessárias para manter a segurança na operação do sistema.

Segundo Vianna (2008), a restrição do acesso de pessoas ao sistema é fundamental para a segurança do sistema. Todos os acessos dos operadores devem ser registrados para auditorias futuras. Por este motivo, o operador deve ser obrigado a entrar, no campo referente ao cadastro de usuários, com um usuário e uma senha previamente cadastrada.

3.1.12 Elaboração da documentação do sistema desenvolvido

Os sistemas supervisórios, como todos os produtos de *software*, necessitam de um conjunto de documentos que forneçam facilidades como:

- Auxílio à operação do aplicativo supervisório através de manuais;
- Detalhamento da descrição do processo que foi automatizado ou supervisionado;
- Auxílio à manutenção do aplicativo através de tabelas de variáveis, alarmes e observações de desenvolvimento.

Essa documentação é de fundamental importância para a manutenção do sistema, para a tomada rápida de decisões em momentos de alarmes críticos e para facilitar o desenvolvimento de futuras expansões do processo.

Todos esses documentos devem ser disponibilizados através de *links* no próprio supervisório com a finalidade de auxiliar a localização dos mesmos e evitar a perda de todo esse material de apoio ao operador.

Os manuais de operação do sistema supervisório podem e devem ser utilizados no treinamento dos operadores, pois eles ficarão habituados a consultar com frequência o manual e seguir a sequência de atividades corretas, diminuindo, assim, as chances de erro e aumentando a eficiência de operação do sistema.

A documentação de supervisórios deve conter:

- Manual do usuário, com instruções de operação do supervisório;
- Manual de descrição do processo;
- Tabelas de alarmes e variáveis;
- Observações de desenvolvimento.

O processo de desenvolvimento de um supervisório está descrito e é necessário um método de avaliação desses aplicativos para que seja garantida a presença dos atributos fundamentais. Portanto, o próximo capítulo descreverá o método de avaliação de supervisórios SuperviQuest.

4. MÉTODO DE AVALIAÇÃO DE SUPERVISÓRIOS (SuperviQuest)

4.1 Objetivos do modelo proposto

A avaliação da qualidade de um *software* como o supervisório pode ser bem estruturada se for baseada na norma ISO/IEC 14598-1, que descreve um processo de avaliação da qualidade de softwares como mostrado na figura 10. Portanto, as etapas deste processo são utilizadas neste método com o objetivo de proporcionar organização e eficiência à avaliação.

A metodologia de avaliação está também descrita em diagramas de caso de uso e diagramas de atividades no anexo 1. O trabalho descrito no anexo foi desenvolvido por uma estudante de Engenharia de Teleinformática, que utilizou a ideia dos trabalhos futuros desta pesquisa para desenvolver sua monografia (MESQUITA, 2013).

A monografia mostra a modelagem com a ferramenta UML com a finalidade de servir de suporte para o desenvolvimento de uma ferramenta de avaliação de *softwares* que utiliza o método SuperviQuest para avaliar supervisórios.

4.2 Primeira etapa de avaliação

A primeira etapa da avaliação da qualidade da norma ISO/IEC 14598-1 está ilustrada na figura 28.

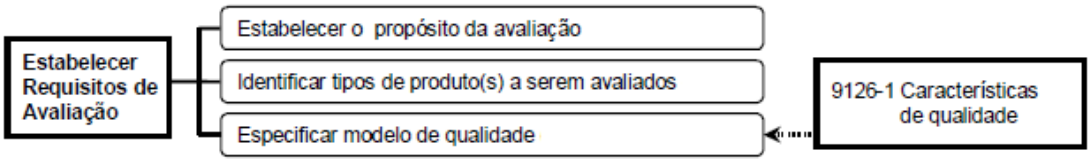


Figura 28 - Primeira etapa do processo de avaliação

Fonte: ISO/IEC 14598, 1998

Para avaliar a qualidade do *software*, primeiro se estabelecem os requisitos da avaliação, então se especifica, projeta e executa a avaliação (ISO/IEC 14598, 1998). O propósito da avaliação da qualidade de um supervisório foi estabelecido neste trabalho e é definido a seguir: “fornecer apoio diretamente ao desenvolvimento e à aquisição de supervisórios, auxiliando o programador a decidir quando liberar o produto através da

estimativa da qualidade final e provendo ao cliente a capacidade de certificar se o produto atende aos critérios de qualidade requeridos".

O tipo de produto (**intermediário ou final**) a ser avaliado, no contexto do trabalho aqui apresentado, foi identificado, sendo este os sistemas supervisórios desenvolvidos em projetos de automação. Para os propósitos de desenvolvimento de supervisório foram definidos características e atributos de qualidade interna e externa que permitem verificar a qualidade durante e após o desenvolvimento.

A seleção dessas características e atributos relevantes para a qualidade final do supervisório foi realizada através de um modelo de qualidade para avaliação de *software* que contém diversos atributos classificados em uma estrutura de árvore hierárquica de características e subcaracterísticas. Esse modelo está descrito na norma ISO/IEC 9126-1 e contempla a definição de seis amplas categorias de características de qualidade de *software*: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade.

Os requisitos de qualidade de *software* expressam as necessidades do usuário e são definidos antes do desenvolvimento. No estágio inicial da avaliação, convém que esses requisitos sejam estudados e identificados, para o planejamento e implementação da avaliação (ISO/IEC 14598, 1998).

Um estudo de requisitos de qualidade foi feito pelo especialista avaliador, que adaptou, para a avaliação de supervisórios, o modelo da norma ISO/IEC 9126-1 através da escolha das características funcionalidade, confiabilidade e usabilidade, e através da proposição da característica configuração. A criação da característica configuração justifica-se pela necessidade de avaliar as configurações de parâmetros de comunicação do supervisório com os controladores ou outros *softwares*. As características e atributos de qualidade escolhidos são mostrados na tabela 6 e também podem ser visualizados nos questionários, respondidos pelos especialistas, de relevância e de presença de características de qualidade de um supervisório que estão nos apêndices B e C.

Os especialistas consultados tiveram a opção de sugerir novas características ou atributos relevantes para a qualidade de supervisórios e indicaram o grau de importância de cada característica e atributo selecionado pelo especialista avaliador.

Tabela 6: Atributos de qualidade de um supervisório.

P1 - Dinâmica do processo	P12 – Documentação
P2 - Segurança de acesso	P13 - Quantidade e prioridade de alarmes
P3 - Armazenamento de dados	P14 - Visualização e intervenção de alarmes
P4 – Configuração dos parâmetros	P15 – Comunicação com o CLP
P5 – Apreensibilidade	P16 - Configuração de tags e drivers de comunicação
P6 - Apresentabilidade quanto à estética das telas e animações	P17 – Quantidade de variáveis
P7 - Apresentabilidade quanto à quantidade de informações na tela	P18 – Organização de *tags*
P8 - Apresentabilidade quanto à hierarquia das telas	P19 – Quantidade de scripts
P9 - Apresentabilidade quanto aos desenhos dos elementos do sistema	P20 - Erros de scripts
P10 – Status das variáveis	P21 – Configuração de Telas
P11 – Recursos gráficos	P22 - Tempo de desenvolvimento

4.3 Segunda etapa de avaliação

A segunda etapa do processo de avaliação da qualidade da norma ISO/IEC 14598-1 está ilustrada na figura 29.

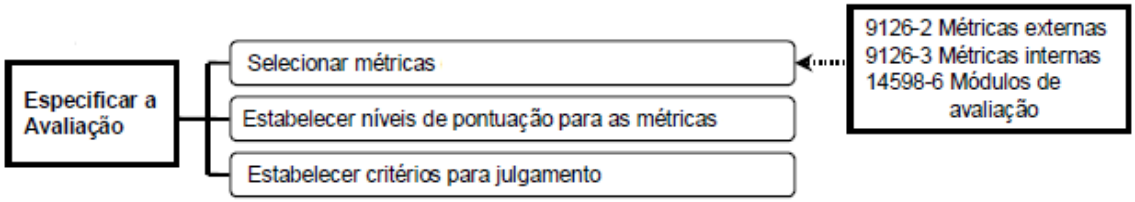


Figura 29 - Segunda etapa do processo de avaliação

Fonte: ISO/IEC 14598, 1998

Na especificação da avaliação é necessária a seleção de métricas que se correlacionem às características do *software*, pois a forma pela qual as características de qualidade têm sido definidas não permite sua medição direta. Muitas medições de *software*

podem ser feitas convenientemente com diversas métricas (ISO/IEC 14598, 1998). Nesta pesquisa foram utilizadas métricas objetivas e subjetivas descritas a seguir.

As medidas baseadas nas opiniões dos especialistas, chamadas de métricas subjetivas, são obtidas através de termos linguísticos como irrelevante, pouco presente, atende moderadamente, facilita totalmente, entre outros. Estes termos possuem associação com níveis de pontuação, que, por sua vez, são relacionados a números *fuzzy* triangulares como ilustrado na tabela 7.

Tabela 7 - Associação entre termos linguísticos e números *fuzzy*

Fonte: Adaptado de Belchior, 1997

Nota	**Números *Fuzzy***	**Termo Linguístico**	**Interpretação**
0	Ñ1 = (0,0; 0,0; 1,0)	Sem Relevância	Indica de maneira absoluta que a existência de certo atributo de qualidade não tem importância.
1	Ñ2 = (0,0; 1,0; 2,0)	Pouco Relevante	Indica de maneira absoluta que a existência de certo atributo de qualidade tem pouca importância.
2	Ñ3 = (1,0; 2,0; 3,0)	Moderadamente Relevante	Indica de maneira absoluta que a existência de certo atributo de qualidade tem moderada importância.
3	Ñ4 = (2,0; 3,0; 4,0)	Relevante	Indica de maneira absoluta que a existência de certo atributo de qualidade é bem importante.
4	Ñ5 = (3,0; 4,0; 4,0)	Muito Relevante	Indica de maneira absoluta que a existência de certo atributo de qualidade é muito importante.

Cada um dos atributos pode ser mais ou menos relevante, bem como pode estar mais ou menos presente numa solução desenvolvida por um programador. Assim, para efeito de avaliação, faz-se necessário quantificar o grau presença dos vários atributos numa solução particular, bem como o grau de relevância de cada um deles.

O grau de presença dos atributos do supervisório, numa escala de 0 a 4, é determinado através da aplicação de um questionário respondido pelos especialistas que, no caso do experimento aqui relatado exerce a função de cliente e avaliador (num cenário real, consultores externos, clientes ou mesmo outros programadores da empresa podem ser os avaliadores). A partir desses valores, números *fuzzy* triangulares, que representam a **presença**

de atributos, são formados para que seja possível determinar quais os atributos deficientes e os eficientes em um supervisório desenvolvido. Logo, em seguida, esses números foram agregados através do método descrito por Belchior (1997).

De modo análogo, o padrão de qualidade, que está relacionado ao **grau de relevância dos atributos** de um supervisório, foi definido através da agregação da opinião de quinze especialistas, após responderem a um questionário para identificar o grau de relevância, numa escala de 0 a 4, de cada atributo. A partir desses valores, foram formados números *fuzzy* triangulares, que foram agregados para formar um padrão de qualidade que pode ser usado para avaliar qualquer supervisório.

Essa relação dos níveis de pontuação com números *fuzzy* chama-se de fuzzificação. A fuzzificação é uma operação em que valores linguísticos, descrições vagas ou qualitativas, são transformados em funções de pertinência (SIMÕES e SHAW, 2007).

A utilização de números *fuzzy* triangulares na fuzzificação deve-se à preferência por um baixo custo computacional na análise dos dados. Outros termos linguísticos também são utilizados para a medição da presença dos requisitos de qualidade de um supervisório. As tabelas 8, 9 e 10 ilustram a associação dos números *fuzzy* com estes termos e suas interpretações.

Tabela 8 - Associação entre termos linguísticos de presença e números *fuzzy*

Nota	Números *Fuzzy*	Termo Linguístico	Interpretação
0	Ñ1=(0,0; 0,0; 1,0)	Total Ausência	Indica de maneira absoluta que o atributo está totalmente ausente no supervisório.
1	Ñ2=(0,0; 1,0; 2,0)	Baixa Presença	Indica de maneira absoluta que o atributo está pouco presente no supervisório.
2	Ñ3=(1,0; 2,0; 3,0)	Moderada Presença	Indica de maneira absoluta que o atributo está moderadamente presente no supervisório.
3	Ñ4=(2,0; 3,0; 4,0)	Alta Presença	Indica de maneira absoluta que o atributo está muito presente no supervisório.
4	Ñ5=(3,0; 4,0; 4,0)	Total Presença	Indica de maneira absoluta que o atributo está totalmente presente no supervisório.

Tabela 9 - Associação entre termos linguísticos de atendimento e números *fuzzy*

Nota	Números *Fuzzy*	Termo Linguístico	Interpretação
0	Ñ1=(0,0; 0,0; 1,0)	Não Atende	Indica de maneira absoluta que o atributo não atende aos requisitos solicitados.
1	Ñ2=(0,0; 1,0; 2,0)	Atende Pouco	Indica de maneira absoluta que o atributo atende pouco aos requisitos solicitados.
2	Ñ3=(1,0; 2,0; 3,0)	Atende Moderadamente	Indica de maneira absoluta que o atributo atende moderadamente aos requisitos solicitados.
3	Ñ4=(2,0; 3,0; 4,0)	Atende Muito	Indica de maneira absoluta que o atributo atende muito aos requisitos solicitados.
4	Ñ5=(3,0; 4,0; 4,0)	Atende Totalmente	Indica de maneira absoluta que o atributo atende totalmente aos requisitos solicitados.

Tabela 10 - Associação entre termos lingüísticos de facilidade e números *fuzzy*

Nota	Números *Fuzzy*	Termo Linguístico	Interpretação
0	Ñ1 = (0,0; 0,0; 1,0)	Não Facilita	Indica de maneira absoluta que o atributo não facilita o entendimento do operador.
1	Ñ2 = (0,0; 1,0; 2,0)	Facilita Pouco	Indica de maneira absoluta que o atributo facilita pouco o entendimento do operador.
2	Ñ3 = (1,0; 2,0; 3,0)	Facilita Moderadamente	Indica de maneira absoluta que o atributo facilita moderadamente o entendimento do operador.
3	Ñ4 = (2,0; 3,0; 4,0)	Facilita Muito	Indica de maneira absoluta que o atributo facilita muito o entendimento do operador.
4	Ñ5 = (3,0; 4,0; 4,0)	Facilita Totalmente	Indica de maneira absoluta que o atributo facilita totalmente o entendimento do operador.

A partir dos números fuzzy gerados da associação com os termos linguísticos foram construídas as funções de pertinência, que estão ilustradas nas figuras 30, 31, 32 e 33.

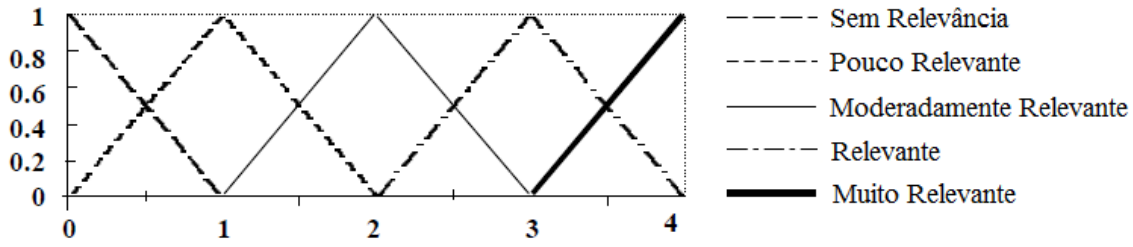


Figura 30 - Funções de Pertinência para termos linguísticos de relevância

Fonte: Adaptado de Belchior, 1997

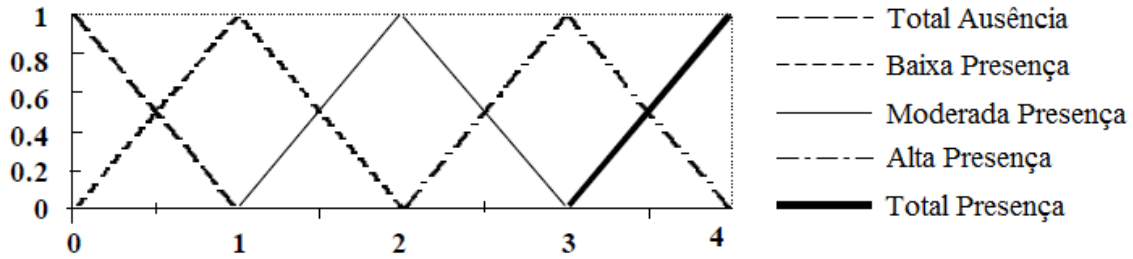


Figura 31 - Funções de Pertinência para termos linguísticos de presença

Fonte: Adaptado de Belchior, 1997

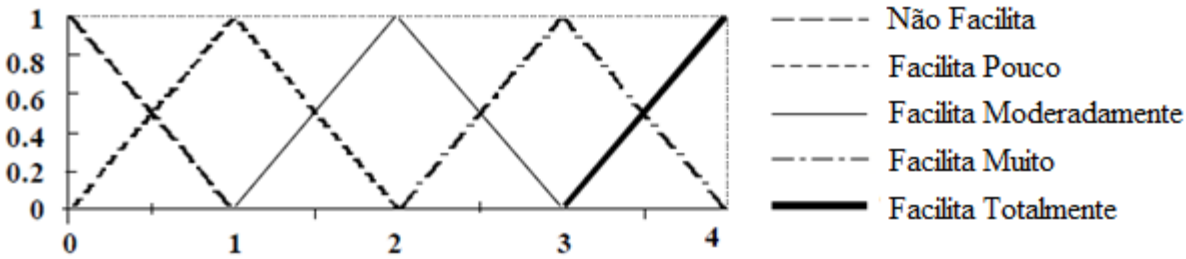


Figura 32 - Funções de Pertinência para termos linguísticos de facilidade

Fonte: Adaptado de Belchior, 1997

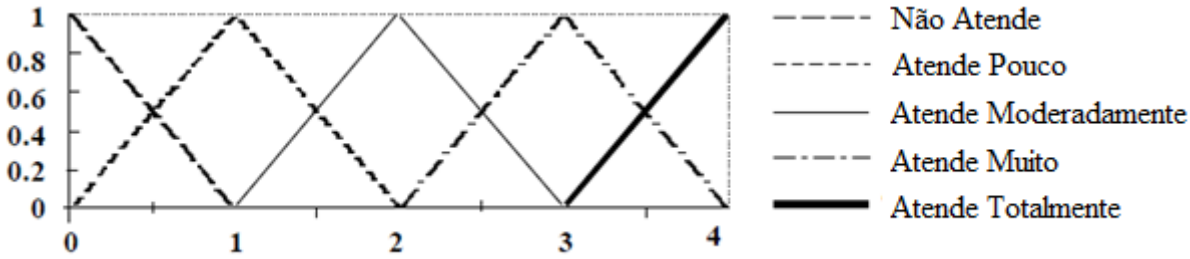


Figura 33 - Funções de Pertinência para termos linguísticos de atendimento

Fonte: Adaptado de Belchior, 1997

O estudo das normas ISO/IEC 9126-2 e 9126-3 permitiu a identificação de duas métricas objetivas que foram adaptadas para serem utilizadas no processo de avaliação. As métricas são descritas na tabela 11.

O estabelecimento de critérios de julgamento para o supervisório é baseado em um levantamento realizado pelo CenPRa (Centro de Tecnologia da Informação Renato Archer), que possui um banco de dados com aproximadamente 300 produtos de *software*. Esses critérios podem ser alterados conforme a neessidade da empresa. O critério adaptado de julgamento deste levantamento está descrito a seguir (MARTINEZ, 1999):

- Numa escala de 0 a 10, para que o supervisório tenha qualidade de **nível Alto**, ele deve atender pelo menos a 80% dos requisitos do padrão de qualidade, ou seja, a nota pode variar de 8,0 a 10,0;
- Para que o supervisório tenha qualidade de **nível Médio**, ele deve atender pelo menos a 50% dos requisitos do padrão de qualidade, ou seja, a nota pode variar de 5,0 a 7,9;
- Para um supervisório que tenha qualidade de **nível Baixo**, a nota pode variar de 0,0 a 4,9.

Tabela 11 - Métricas objetivas para a qualidade de supervisórios

Nome da Métrica	**Propósito da métrica**	**Medida e Fórmula**	**Interpretação**	**Tipo de Medida**
Configuração de *tags* e *drivers* de comunicação	Identificar o grau de acerto das configurações de *tags* e *drivers* de comunicação feitas pelo programador	X = 1 - (A / B) A = número de erros de configuração de *tags* ou *drivers* identificados B = número de *tags* + número de *drivers*	0 <= X <= 1 Quanto mais próximo de 1, melhor.	A=quantidade B=quantidade X=quantidade/ quantidade
Configuração de scripts	Identificar o grau de acerto das configurações de scripts feitas pelo programador	X = 1 - (A / B) A = número de operações incorretas dos scripts B = número total de operações dos scripts	0 <= X <= 1 Quanto mais próximo de 1, melhor.	A=quantidade B=quantidade X=quantidade/ quantidade

O supervisório está pronto para ser entregue quando está no nível Alto. O nível Médio indica que são necessários alguns ajustes para que o padrão de qualidade seja atendido. Já o nível Baixo aponta que os requisitos precisam ser revistos no supervisório e uma nova avaliação deve acontecer para que seja entregue ao cliente.

Para atributos que proporcionam segurança para todo o sistema recomenda-se que sejam aceitos somente com, pelo menos, 95% dos requisitos do padrão de qualidade atendidos. Esta prática é necessária para que se possa manter a segurança de um sistema.

Os critérios de aceitação são apresentados na figura 34.

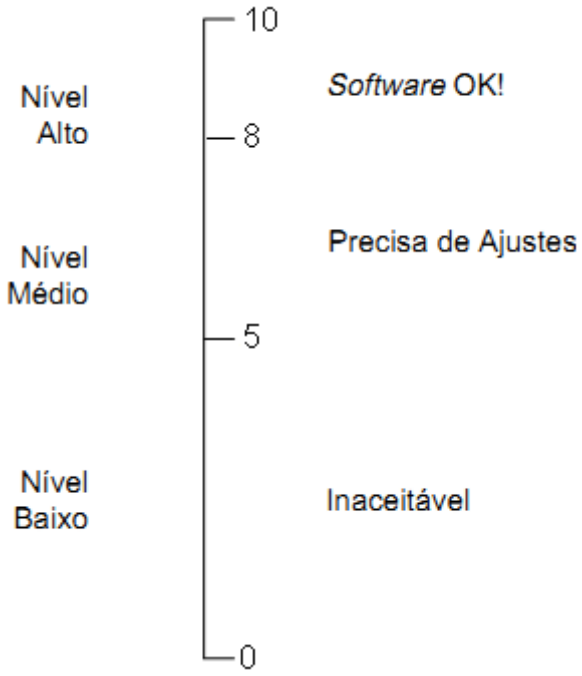


Figura 34 - Critério de aceitação

4.4 Terceira etapa de avaliação

A terceira etapa do processo de avaliação da qualidade da norma ISO/IEC 14598-1 está ilustrada na figura 35.

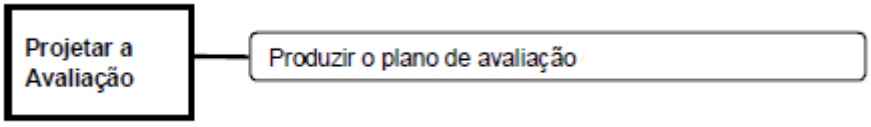


Figura 35 - Terceira etapa do processo de avaliação

Fonte: ISO/IEC 14598, 1998

O projeto de avaliação deve documentar os procedimentos a serem usados pelo especialista avaliador para realizar as medições contidas na especificação de avaliação. O

avaliador deve aplicar um plano de avaliação que descreva os recursos necessários para realizar a avaliação especificada (ISO/IEC 14598, 1998).

O plano de avaliação de um supervisório deve ser composto das seguintes atividades:

- Verificar a disponibilidade dos recursos necessários para a avaliação;
- Documentar os métodos de avaliação e produzir um plano preliminar;
- Programar as ações de avaliação.

Um dos principais recursos desse plano é a mão de obra que fará a avaliação. Na metodologia de avaliação proposta neste trabalho é necessário que se forme um grupo de especialistas que desenvolvem supervisórios na própria empresa para opinarem sobre a presença dos atributos de qualidade no aplicativo desenvolvido. A empresa pode, na ausência de um grupo, adaptar o método de avaliação de supervisórios para a sua realidade ou contratar consultores que possam fazer a avaliação.

Toda a documentação do supervisório desenvolvido necessária para a avaliação deve ser disponibilizada aos especialistas com o objetivo de facilitar o julgamento dos atributos de qualidade de um supervisório. Documentos como um manual de operação do sistema, manual descritivo do processo, as tabelas de variáveis de entrada, saída e de alarme são essenciais para a avaliação.

O plano preliminar de avaliação, que é composto pela especificação dos métodos de avaliação, foi elaborado e está descrito em conjunto com a execução da avaliação. Com vistas a facilitar o entendimento do plano preliminar de avaliação, um estudo de caso ilustrará a aplicação do plano na seção 4.6.

A partir desse ponto podem-se marcar as ações planejadas de avaliação do supervisório desenvolvido tendo em vista a disponibilidade dos recursos, os prazos de entrega do aplicativo aos clientes e considerando o tempo disponível dos avaliadores para esta atividade.

4.5 Quarta etapa de avaliação

A quarta etapa do processo de avaliação da qualidade da norma ISO/IEC 14598-1 está ilustrada na figura 36.

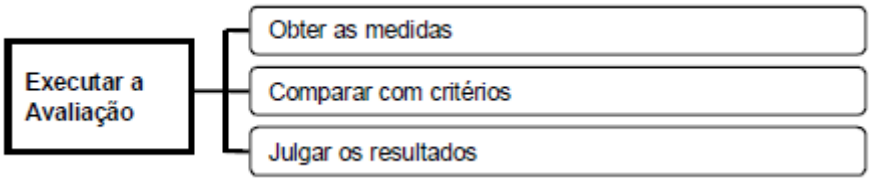


Figura 36 - Quarta etapa do processo de avaliação

Fonte: ISO/IEC 14598-1, 1998

A execução da avaliação é ilustrada na figura 37, onde podemos ter uma visão geral do tratamento dos dados.

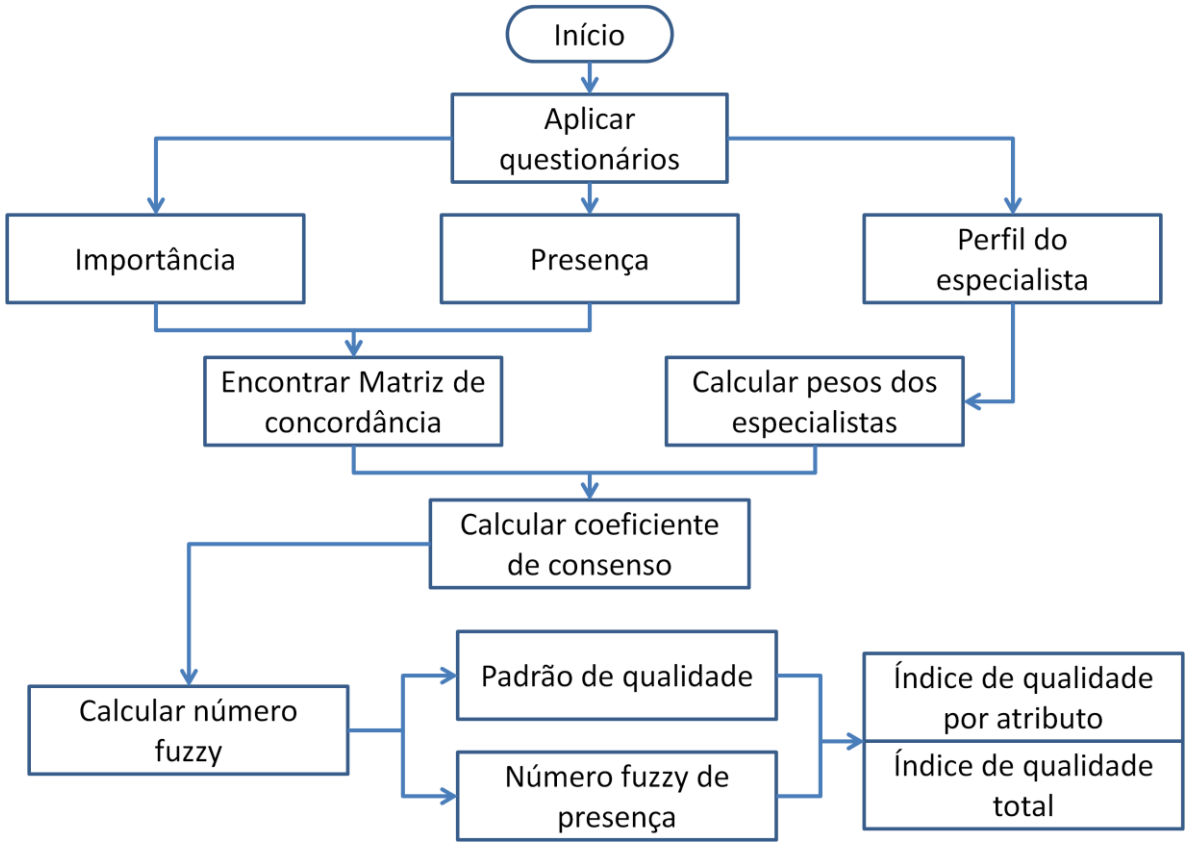


Figura 37 – Visão geral do processo de avaliação

A avaliação começa a ser executada com a aplicação de três questionários listados a seguir:

- Questionário de delimitação do perfil do especialista (QDPE);
- Questionário do Grau de presença de características de qualidade de um supervisório (QPCQS);
- Questionário do Grau de relevância de características de qualidade de um supervisório (QRCQS).

No primeiro questionário, a experiência do especialista é estimada através de perguntas a respeito da quantidade de supervisórios que já desenvolveu ou nos quais fez manutenção, entre outras. Desta forma, os dados são coletados e o perfil do especialista é formado através de pontuações dadas para cada resposta que, somadas, indicam numericamente a experiência dos especialistas.

A opinião de todos os especialistas consultados deve ser ponderada pela experiência que o mesmo tem em desenvolvimento de supervisórios para que seja dado um peso maior aos mais experientes. Por esse motivo, o **peso da experiência de cada especialista** é calculado somando-se as pontuações dos especialistas (tQDPE), obtidas pelo questionário, e dividindo a soma dos pontos de cada especialista pela soma dos pontos de todos os j-ésimos especialistas. A equação 2 foi utilizada para realização deste cálculo (BELCHIOR, 1997).

$$PE_i = \frac{tQDPE_i}{\sum_{j=1}^{n} tQDPE_j} \tag{2}$$

4.6 Matriz de concordância

A existência de um consenso entre os especialistas foi examinada através da elaboração de uma **matriz de concordância para cada atributo** de qualidade de supervisórios, composta pelos cálculos dos graus de concordância entre dois especialistas quaisquer.

Através do cálculo de alguns índices de concordância entre especialistas, aplica-se o conceito de similaridade (usado em tomada de decisão), nas avaliações da qualidade de *software* e na apuração das estimativas dos especialistas. Assim sendo, o resultado final da avaliação tende para onde houve maior grau de consenso, não gerando um resultado de tendência central. Quando, por exemplo, um especialista divergir totalmente dos outros especialistas, isto é, seu estado de concordância em relação a todos os outros for nulo, seu julgamento é automaticamente desprezado, pelo próprio modelo (BELCHIOR, 1997).

Calculou-se o grau de concordância $C(\tilde{N}_i,\tilde{N}_j)$ entre dois especialistas, combinando-se os julgamentos dos especialistas E_i e E_j, através da razão entre a área de interseção de suas funções de pertinência e a área total. A equação 3 foi utilizada para o cálculo do grau de concordância (BELCHIOR, 1997). As equações seguintes foram aplicadas

aos dados de importância e presença. $\mu_{\tilde{N}_i}(x)$ *e* $\mu_{\tilde{N}_j}(x)$ são as funções de pertinência do i-ésimo especialista e do j-ésimo especialista, respectivamente.

$$C(\tilde{N}_i, \tilde{N}_j) = \frac{\int \left(min\left\{\mu_{\tilde{N}_i}(x), \mu_{\tilde{N}_j}(x)\right\}\right) dx}{\int \left(max\left\{\mu_{\tilde{N}_i}(x), \mu_{\tilde{N}_j}(x)\right\}\right) dx} \quad (3)$$

A matriz de concordância foi construída com a alocação dos valores dos graus de concordância entre dois especialistas, fornecendo as indicações de aquiescência entre eles, onde $C_{ij} = C(\tilde{N}_i, \tilde{N}_j)$, se $i \neq j$ e $C_{ij} = 1$, se $i = j$ (BELCHIOR, 1997). Quando não houve interseção entre o i-ésimo e o j-ésimo especialista, ou seja, $C_{ij} = 0$, foi necessário saber se eles entenderam bem o questionamento e se podiam entrar num consenso.

$$MC = \begin{bmatrix} 1 & C_{12} & \cdots & C_{1j} & \cdots & C_{1n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{i1} & C_{i2} & \cdots & C_{ij} & \cdots & C_{in} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{n1} & C_{n2} & \cdots & C_{nj} & \cdots & 1 \end{bmatrix}$$

4.7 Coeficiente de consenso dos especialistas

Para encontrar o **coeficiente de consenso** entre os especialistas, presente na figura 37 deve-se encontrar a concordância e o grau de concordância relativa. O cálculo da **concordância relativa (CR_i)** do i-ésimo especialista, pode indicar os maiores índices de consenso entre o i-ésimo e os j-ésimos especialistas consultados. Portanto, o procedimento de cálculo da média quadrática do grau de concordância entre os especialistas foi realizado através da equação 4.

$$CR_i = \sqrt{\frac{1}{n-1}\sum_{j=1}^{n} C_{ij}^2} \quad (4)$$

O **grau de concordância relativa (GCR_i)** de um especialista em relação a todos os outros (j-ésimos especialistas), foi obtido pela média ponderada de CR_i de cada especialista, implementada através da equação 5.

$$GCR_i = \frac{CR_i}{\sum_{j=1}^{n}(CR_j)} \quad (5)$$

O **coeficiente de consenso dos especialistas** pode indicar de forma geral a porcentagem de consenso obtida na avaliação. O coeficiente de consenso, obtido para o i-

ésimo especialista (CCE_i) considera tanto o GCR_i, quanto o peso, PE_i, de cada um deles (BELCHIOR, 1997). O procedimento foi realizado através da equação 6.

$$CCE_i = \frac{GCR_i \bullet PE_i}{\sum_{j=1}^{n}(GCR_j \bullet PE_j)} \quad (6)$$

4.8 Agregação dos números *fuzzy* e defuzzificação

Os números *fuzzy* foram agregados através da utilização da equação 7. Desta forma, considerou-se a opinião, por atributo, de cada especialista como "$\tilde{N}_i$" ponderada pelo coeficiente de consenso de cada especialista "CCE" para gerar os números *fuzzy* agregados ($\tilde{N}$). Na equação 7, para cada atributo, calculou-se o número *fuzzy* agregado com a agregação das opiniões dos especialistas (BELCHIOR, 1997).

$$\tilde{N} = \sum_{i=1}^{n}(CCE_i \bullet \tilde{N}_i) \quad (7)$$

Os elementos do número *fuzzy* $\tilde{N}$ = (a, m, b) são encontrados separadamente da seguinte forma:

$$a = (CCE_1 \,.\, a_1) + (CCE_2 \,.\, a_2) + \ldots + (CCE_{15} \,.\, a_{15})$$

$$m = (CCE_1 \,.\, m_1) + (CCE_2 \,.\, m_2) + \ldots + (CCE_{15} \,.\, m_{15})$$

$$b = (CCE_1 \,.\, b_1) + (CCE_2 \,.\, b_2) + \ldots + (CCE_{15} \,.\, b_{15})$$

Os elementos a_1, m_1 e b_1 são pertencentes aos números *fuzzy* gerados pela opinião de cada especialista.

Os números *fuzzy* de relevância agregados formam o **padrão de qualidade** (PQ), que serve de elemento comparativo para determinar a qualidade de qualquer supervisório.

Para cada atributo foi necessário obter um valor numérico real que melhor representasse um número *fuzzy* triangular. Este processo de conversão chama-se de defuzzificação e pode ser feito através de diversos métodos. Conforme Simões e Shaw (2007), a escolha do método de defuzzificação em aplicações de suporte à decisão depende do contexto da decisão. Em decisões quantitativas, como alocação de recursos, pode-se usar o método centro do máximo e em decisões qualitativas, como qualidade de *software*, é recomendada a utilização do método média dos máximos.

Neste trabalho, o procedimento de defuzzificação foi realizado através do método média dos máximos. Como os números *fuzzy* triangulares (a, m, b) só possuem um valor

máximo, o valor *crisp* (número natural) será o termo central do triângulo *fuzzy*, ou seja, será o termo ‘m’. A equação 8 foi utilizada para o cálculo do “valor *crisp*”.

$$V_{crisp} = m \tag{8}$$

Para a realização do processo de **normalização dos valores *crisp* de relevância** (importância) do i-ésimo especialista é necessário que cada valor *crisp* do i-ésimo especialista seja dividido pela soma de todos os valores *crisp* dos 22 especialistas, ou seja, o valor normalizado é calculado de acordo com a equação 9 e é mostrado nas tabelas 18 e 19 na coluna de nome “Valor normal”.

$$V_{norm_i} = \frac{V_{crisp_i}}{\sum_{j=1}^{22} V_{crisp_j}} \tag{9}$$

A partir do processo de normalização pode-se examinar a porcentagem de importância de um determinado atributo para o desenvolvimento de supervisórios com qualidade. Esta porcentagem indica o quanto o atributo contribui para a qualidade total do supervisório, mas não significa que a ausência do mesmo diminui a qualidade somente na porcentagem de importância do atributo. Todos os atributos são essenciais para a entrega do supervisório.

4.9 Cálculo do índice de qualidade por atributo

Pode-se avaliar a qualidade de cada atributo no supervisório através do cálculo do índice de qualidade por atributo, que vai indicar se o atributo está ou não dentro do padrão de qualidade desejado, e em que porcentagem. A existência de um padrão de qualidade definido torna possível a comparação dos resultados de avaliação da qualidade de um supervisório com este padrão. Com isto, pode-se obter o índice de qualidade através dos seguintes procedimentos (BELCHIOR, 1997):

1. **Redefinição da função característica do atributo de qualidade**: redefinir o número *fuzzy* triangular de importância do tipo $\tilde{N}_i = (a_i, m_i, b_i)$, para o número *fuzzy* trapezoidal $Q_i = (a_i, m_i, m_i, b_i)$. O mapeamento de $\tilde{N}_i$ para Q_i resultará, então, na função $Q_i = (a_i, m_i, b_n, b_n)$, onde *n* é o limite superior do *conjunto referencial* já definido (valor 4).
2. **Cálculo do índice de qualidade:** o índice de qualidade, q_k, de cada atributo *k*, que está sendo avaliado, será formalizado pela equação 10. $\mu_{Q_k}(x)$ *e* $\mu_{\tilde{N}_k}(x)$ são as funções de pertinência do número *fuzzy* trapezoidal de importância do k-ésimo atributo e do número *fuzzy* triangular de presença do k-ésimo atributo, respectivamente. O índice é calculado pela interseção

das áreas dos números *fuzzy* de importância e presença dividido pela área do número *fuzzy* de presença.

$$q_k = \frac{\int \left(min\{\mu_{Q_k}(x), \mu_{\tilde{N}_k}(x)\}\right) dx}{\int \left(\mu_{\tilde{N}_k}(x)\right) dx} \quad (10)$$

- Uma vez que q ϵ [0, 1], quando q = 1, isto significa que o atributo avaliado atinge totalmente o padrão de qualidade; se q = 0, então o atributo avaliado está totalmente fora do padrão de qualidade; se $0 \leq q \leq 1$, o atributo está dentro do padrão de qualidade, na proporção do valor de q, isto é, o atributo avaliado é q% do padrão de qualidade.

Alguns exemplos de transformação do número *fuzzy* triangular, do padrão de qualidade, para um número *fuzzy* trapezoidal são mostrados na tabela 12.

Tabela 12 - Exemplo de transformação dos números *fuzzy*

Número *Fuzzy* Triangular	**Número *Fuzzy* Trapezoidal**
Ñ = (1,37 ; 2,37 ; 3,37)	Q = (1,37 ; 2,37 ; 4 ; 4)
Ñ = (2 ; 3 ; 4)	Q = (2 ; 3 ; 4 ; 4)
Ñ = (0 ; 1 ; 2)	Q = (0 ; 1 ; 4 ; 4)

O índice de qualidade está comparando os números *fuzzy* que constituem o padrão de qualidade com os números *fuzzy* que representam o grau de presença dos atributos. Qualquer valor que esteja acima do padrão de qualidade é também considerado muito bom para a qualidade do *software*.

Pode-se verificar esta interpretação através do seguinte exemplo: Considerando o o número *fuzzy* (2; 2,5; 3) como padrão de qualidade e recebeu o número *fuzzy* (3; 3,5; 4) como grau de presença do atributo. Isto significa que o atributo é 55% Moderadamente relevante e 45% Relevante. Portanto o triângulo de importância (padrão de qualidade) será comparado com o triângulo de presença que é 45% Moderadamente presente e 55% Altamente Presente. Essa comparação com um triângulo de presença equivalente ao de relevância justifica-se pela interpretação de que um atributo deve ser tão presente quanto é relevante.

O exemplo está ilustrado na figura 38 e 39. Na figura 38, o triângulo de importância é o amarelo e o azul é o triângulo de presença. O atributo terá índice de qualidade zero quando o triângulo de presença não possuir interseção com o de relevância e estiver

localizado à esquerda dele. Entretanto, nota-se que este atributo está altamente presente no *software*, logo, é preciso ajustar a forma de calcular o índice.

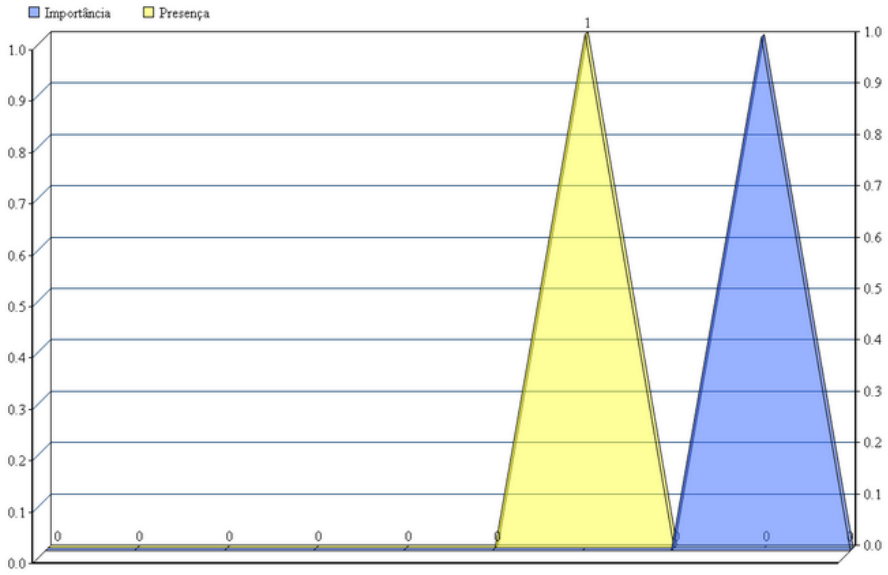


Figura 38 – Exemplo de número de presença maior que número de importância

A figura 39 mostra a transformação que o número *fuzzy* de importância sofre, de triangular para trapezoidal, para que o índice de qualidade expresse a real qualidade do atributo. Generalizando, verifica-se que quando o triângulo de presença não possuir interseção com o de relevância e estiver localizado à direita dele, pode-se dizer que o atributo terá índice de qualidade um, ou seja, 100%, pois isto significa que o atributo está menos "moderadamente presente" e mais "altamente presente" ou "totalmente presente".

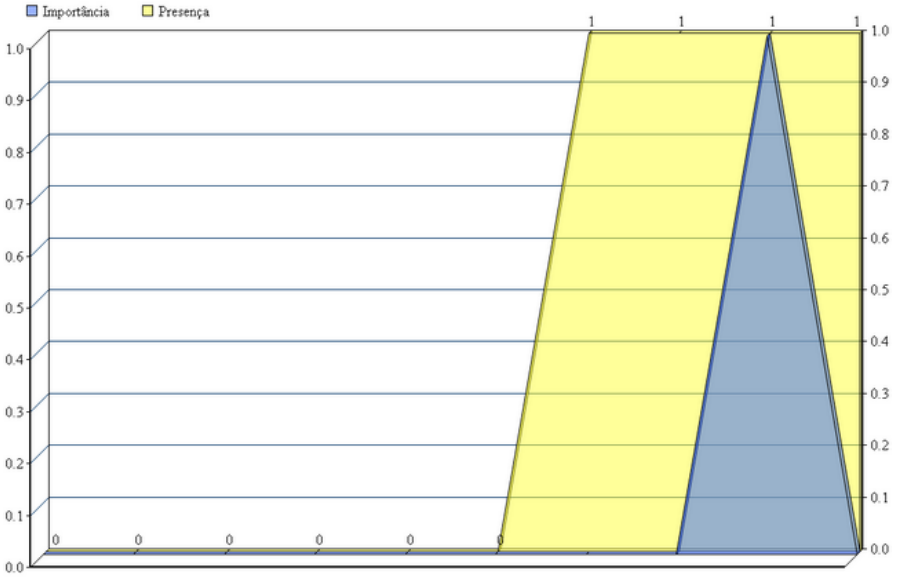


Figura 39 – Exemplo de transformação de número *fuzzy* triangular para trapezoidal

4.10 Cálculo do índice de qualidade total

A qualidade total de um supervisório pode ser definida, encontrada e avaliada através do cálculo de um índice de qualidade total. Inicialmente, foi pesquisado em Boente (2009) uma forma de encontrar um índice para qualidade de um *software*, porém verificou-se que o cálculo feito naquele trabalho praticamente era independente dos valores de relevância dos atributos. A equação 11 mostra a forma que Boente (2009) encontra o índice de qualidade, onde $V_{norm\ imp\ i}$ e $V_{norm\ pres\ i}$ são, respectivamente, os valores *crisp* de importância e de presença do i-ésimo atributo, normalizados pelo valor *crisp* máximo.

$$I_{Qual} = \frac{\sum_{i=1}^{22}\left(V_{norm\ imp_i} * V_{norm\ pres_i}\right)}{\sum_{i=1}^{22} V_{norm\ imp_i}} \tag{11}$$

Outra forma de atribuir um índice de qualidade a um *software* é proposta neste trabalho devido aos problemas encontrados na equação 11, em que os valores normalizados são utilizados para se calcular uma média ponderada pelos valores de relevância dos 22 atributos de qualidade. O índice de qualidade total é calculado através da equação 12. q_i é o valor do índice de qualidade do i-ésimo atributo e V_{norm_i} é o valor *crisp* de importância normalizado.

$$I_{Qualtotal} = \frac{\sum_{i=1}^{22}\left(V_{norm_i} * q_i\right)}{\sum_{i=1}^{22} V_{norm_i}} \tag{12}$$

A comparação entre o índice de qualidade total proposto e o de Boente (2009) foi realizada através de alguns experimentos. Inicialmente, os valores de presença de um supervisório desenvolvido foram mantidos constantes e os 22 valores de relevância receberam um valor único como mostrado na tabela 13. Os resultados estão ilustrados no gráfico 1.

Tabela 13: Números *fuzzy* de relevância aplicados no experimento

Número do gráfico		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Número *fuzzy* de relevância	**a**	2,9	2,8	2,7	2,6	2,5	2,2	2	1,5	1	0,5	0
	m	3,9	3,8	3,7	3,6	3,5	3,2	3	2,5	2	1,5	1
	b	4	4	4	4	4	4	4	3,5	3	2,5	2

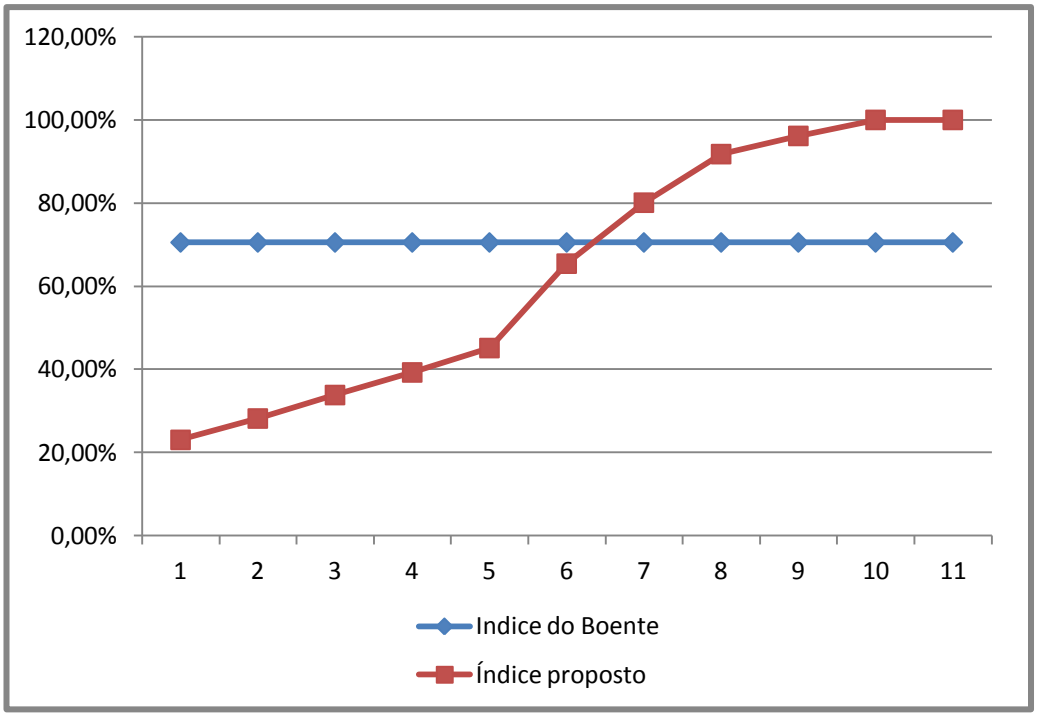


Gráfico 1 – Comparação entre índices quando a presença é constante.

Nota-se que o índice de Boente se mantém constante em 70,54% mesmo que os valores de relevância sejam alterados em toda sua escala. Este erro é sanado pela utilização do índice proposto neste trabalho, que é sensível às variações dos valores de relevância obtidos por um consenso entre especialistas.

Em outro experimento, os números *fuzzy* de relevância são mantidos constantes e os valores de presença são variados conforme mostra a tabela 14. Nota-se que os índices de Boente se mantêm constante em 100% mesmo que os valores de presença sejam alterados em toda sua escala. Já o cálculo proposto neste trabalho torna o índice sensível à mudança do número *fuzzy* de presença como visto no gráfico 2.

Tabela 14: Números *fuzzy* de presença aplicados no experimento.

Número do gráfico		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Número *fuzzy* de presença	**a**	2,9	2,8	2,7	2,6	2,5	2,2	2	1,5	1	0,5	0
	m	3,9	3,8	3,7	3,6	3,5	3,2	3	2,5	2	1,5	1
	b	4	4	4	4	4	4	4	3,5	3	2,5	2

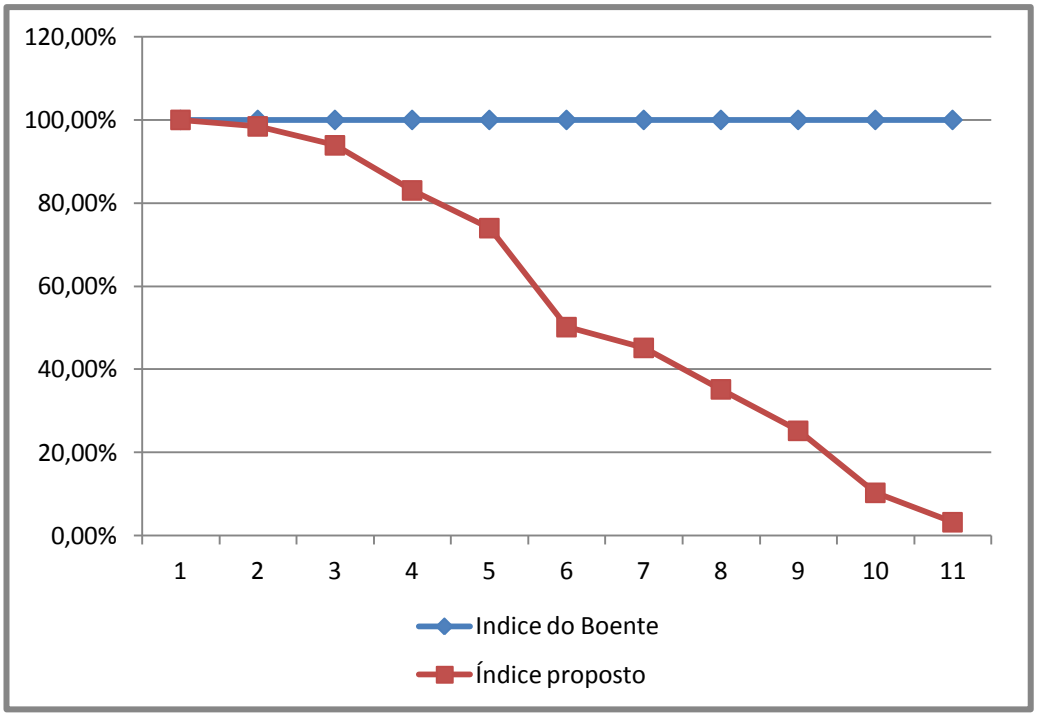


Gráfico 2 – Comparação entre índices quando a relevância é constante.

Através da avaliação do índice de qualidade por atributo e do índice de qualidade total pode-se inferir se o supervisório está atendendo aos requisitos básicos de qualidade. Porém, só pode-se autorizar a liberação do supervisório se todos os índices estiverem dentro do padrão de qualidade, ou seja, os índices devem, obrigatoriamente, ser utilizados em conjunto para a tomada de decisões.

Desta forma, está definido o tratamento dos dados para a avaliação da qualidade de supervisórios, sendo exemplificada cada uma das etapas. No capítulo seguinte, são descritos os resultados desta experiência, realizada com o objetivo de validá-lo.

5. RESULTADOS E DISCUSSÕES

5.1 Avaliação de um supervisório desenvolvido numa instituição de ensino

Com o objetivo de validar o método SuperviQuest foi desenvolvido um supervisório. Os especialistas foram consultados para opinarem a respeito do grau de presença dos atributos de qualidade nesse supervisório elaborado por programadores iniciantes (estudantes que estão no último semestre do curso de Mecatrônica Industrial do IFCE) que se basearam na metodologia ilustrada na figura 16, proposta neste trabalho.

Os estudantes desenvolveram um supervisório que simulava o processo de funcionamento de um reator químico. Foi utilizado o ambiente de desenvolvimento Elipse SCADA para elaborar o supervisório.

A escolha do ambiente de desenvolvimento do supervisório que foi avaliado nesta dissertação deve-se a todas as características descritas anteriormente e às seguintes vantagens:

- Existência de um modo demonstração de configuração que permite que estudantes desenvolvam aplicativos sem a necessidade de uma licença de programador;
- Possibilidade de tirar dúvidas com o suporte on-line da Elipse que responde os questionamentos com brevidade e disponibiliza diversos manuais para programadores;
- O *software* é nacional e é bastante utilizado na região Nordeste em empresas como a FAPIJA, Intervet, COGERH, Hotel Luzeiros, CEMEC, Dispa, Findal, FAE, SEMARH, COELCE, CAGECE, Petrobrás, Gerdau e, segundo Elipse (2012), está presente em países como a Argentina, Rússia, Alemanha, Portugal, Estados Unidos, entre outros;

O *software* teve uma limitação de 20 variáveis em virtude do ambiente de desenvolvimento. A aplicação foi desenvolvida em três meses, através dos *feedbacks* com o cliente (professor). O *feedback* consistiu de uma versão inicial, três versões intermediárias e uma versão final.

Foram consultados 15 especialistas em desenvolvimento de supervisórios que trabalham em 5 instituições listadas a seguir:

- Infitech Tecnologias Inovadoras (Empresa de Automação)
- Ímpar Tecnologias (Empresa de Automação)
- Nexus Tecnologias (Empresa de Automação)
- Senai (Instituição de ensino)
- IFCE (Instituição de ensino)

Todos os especialistas responderam aos questionários, e a tabela de identificação do peso dos especialistas foi elaborada através dos dados do QDPE e está ilustrada na tabela 15. As células P1 a P9 são as pontuações das perguntas feitas aos especialistas E1 a E15. tQDPE é o total de pontuação de cada especialista e P_{Ei} é o peso de cada especialista.

Tabela 15 - Identificação dos pesos dos especialistas

Coleta dos Resultados do QDPE											
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**tQDPE**	**P_{Ei}**
E1	4	4	4	1	3	0	4	2	1	23	0,06628
E2	0	3	3	0	3	0	3	2	1	15	0,04323
E3	1	3	2	0	1	0	2	2	1	12	0,03458
E4	4	4	4	2	4	2	4	3	2	29	0,08357
E5	4	4	4	1	2	4	0	2	3	24	0,06916
E6	4	4	4	2	3	3	4	3	2	29	0,08357
E7	3	3	4	2	4	2	2	3	2	25	0,07205
E8	4	4	4	2	3	1	4	2	2	26	0,07493
E9	4	4	4	4	4	1	4	2	1	28	0,08069
E10	4	4	4	3	4	0	3	3	1	26	0,07493
E11	4	4	4	1	4	0	3	3	2	25	0,07205
E12	3	4	3	2	2	0	3	2	1	20	0,05764
E13	2	2	1	2	2	0	0	2	1	12	0,03458
E14	4	4	4	4	3	3	1	3	2	28	0,08069
E15	4	4	4	2	4	0	2	3	2	25	0,07205
										Total	**Total**
										347	1,00

O gráfico 3 foi elaborado com o objetivo de mostrar a diferença entre os pesos atribuídos aos especialistas consultados.

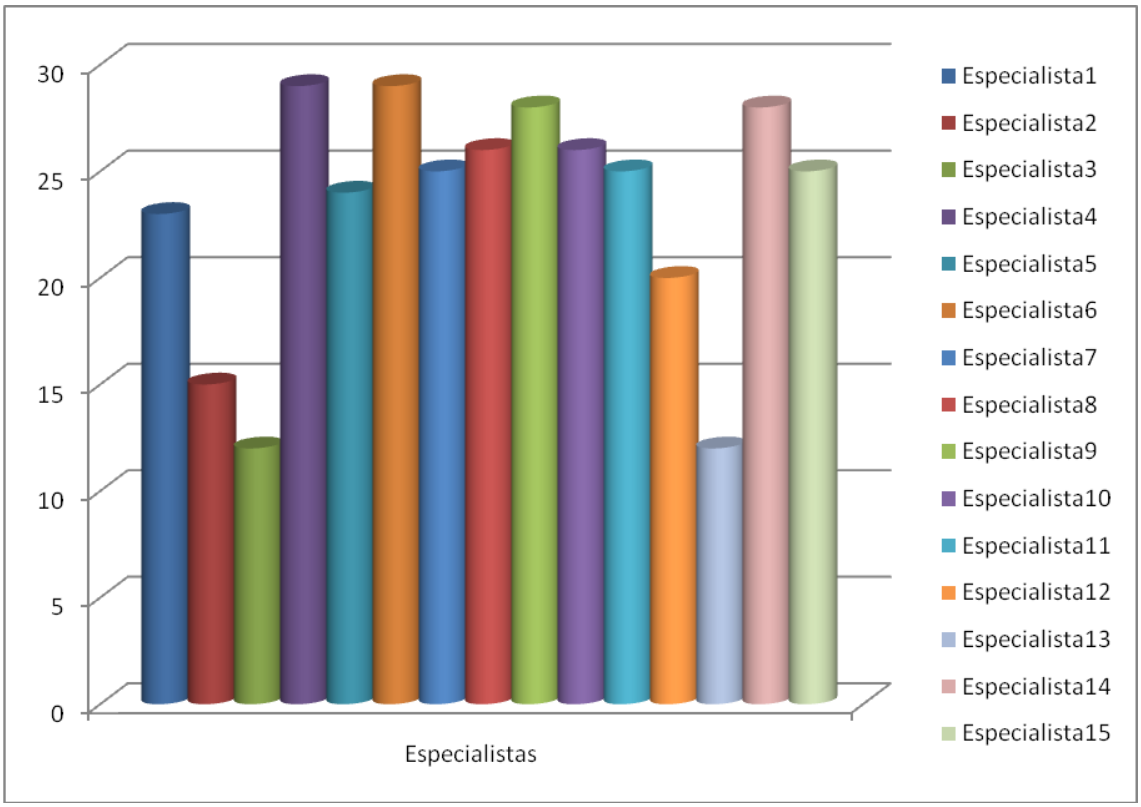


Gráfico 3 - Comparação dos Pesos dos Especialistas

A consulta a respeito da relevância dos atributos de qualidade de supervisórios foi realizada através da aplicação do questionário QRCQS (apêndice B) e os dados foram organizados na tabela 16, de forma que os valores que estão na tabela correspondem aos valores da escala para a alternativa que foi marcada pelo especialista. Os especialistas também opinaram a respeito do supervisório desenvolvido através do questionário QPCQS (apêndice C). Os dados foram coletados e estão na tabela 17.

Tabela 16 - Grau de relevância dos atributos de qualidade de um supervisório

Coleta dos Dados do QRCQS															
Especialistas															
	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15
P1	4	3	3	3	4	4	3	3	3	3	3	3	3	3	3
P2	3	3	3	4	3	3	3	4	3	3	3	3	3	3	3
P3	3	3	3	4	4	4	3	4	3	3	3	3	3	3	3
P4	3	3	3	3	3	3	3	1	3	1	2	3	3	3	3
P5	3	3	3	4	4	3	3	4	3	3	3	3	3	3	3
P6	3	3	3	3	3	3	4	4	3	3	3	3	3	3	3
P7	3	3	3	3	4	4	4	3	3	3	3	3	3	3	3
P8	4	3	4	3	3	3	3	3	3	3	3	3	3	3	3
P9	2	2	2	2	2	3	3	3	3	2	2	3	4	3	2
P10	3	3	3	3	3	3	3	2	3	3	3	3	4	3	4
P11	3	3	3	4	2	3	3	3	3	3	3	3	3	3	2
P12	3	3	3	4	1	3	3	3	3	3	3	3	4	3	2
P13	3	3	3	3	4	4	3	3	3	3	2	3	3	2	2
P14	3	3	3	4	2	3	3	3	3	3	3	2	3	3	2
P15	3	3	3	3	4	4	4	3	3	3	3	3	3	3	3
P16	3	3	3	3	4	4	4	3	3	3	3	3	3	3	3
P17	3	2	3	3	3	3	4	3	3	3	3	3	3	2	3
P18	2	3	3	3	3	2	4	2	4	3	3	2	3	2	3
P19	3	3	3	3	3	3	4	1	3	3	2	3	3	3	2
P20	4	3	2	4	3	4	3	3	3	3	3	3	3	2	3
P21	4	3	3	3	3	3	3	3	4	3	3	4	3	3	3
P22	3	2	3	3	4	3	3	3	3	3	3	3	3	2	2

Tabela 17 - Grau de presença dos atributos de qualidade de um supervisório

Coleta dos Dados do QPCQS															
Especialistas															
	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15
P1	3	3	3	3	3	1	3	3	4	3	3	3	3	3	3
P2	4	3	4	3	3	2	3	3	3	4	3	2	3	3	3

P3	3	4	3	3	3	1	3	3	3	3	3	3	4	3	3
P4	4	3	3	3	3	2	3	2	2	3	3	3	2	3	2
P5	3	3	3	3	3	2	2	3	3	3	3	3	3	3	3
P6	3	4	3	3	1	3	2	2	2	3	3	2	3	3	3
P7	3	3	2	2	1	2	2	3	3	3	3	3	2	4	3
P8	4	3	3	2	1	2	2	2	2	2	2	2	3	3	2
P9	3	3	3	3	3	2	3	3	3	4	2	3	4	3	3
P10	4	3	4	3	3	3	3	3	3	3	3	3	3	3	3
P11	4	3	3	3	3	2	2	4	3	2	2	3	3	4	3
P12	2	4	2	3	2	2	3	3	2	1	1	3	2	3	2
P13	3	3	3	3	3	2	2	2	2	2	3	2	3	3	3
P14	3	3	2	2	2	2	2	2	2	2	2	2	3	3	3
P15	0	1	0	1	0	0	0	1	1	1	1	0	1	0	1
P16	1	1	1	1	0	0	0	1	0	1	1	0	1	1	1
P17	4	3	3	3	3	2	3	3	2	3	4	3	3	3	3
P18	4	3	3	3	3	3	2	3	2	1	3	2	3	4	3
P19	3	2	2	3	2	2	2	2	2	3	2	3	3	3	2
P20	2	3	4	3	4	4	3	3	4	4	4	4	2	4	4
P21	3	3	3	3	3	2	3	2	3	2	2	2	3	3	3
P22	3	3	3	3	2	3	3	2	2	2	3	3	2	3	3

5.2 Grau de importância de cada atributo

Uma pesquisa de campo para a avaliação da qualidade de supervisório foi realizada e os resultados foram apurados, compondo-se de duas fases:

- Determinação do padrão de qualidade para supervisórios; e
- Avaliação de um supervisório real através da comparação do padrão de qualidade com os dados do QPCQS (Questionário de presença).

Na determinação do padrão de qualidade, obtiveram-se os números *fuzzy* agregados, através dos cálculos presente nas seções 4.5 à 4.8, que foram defuzzificados, pelo método “média dos máximos”, e normalizados pelo somatório dos termos para serem

avaliados. As tabelas 18 e 19 mostram os números *fuzzy* agregados, os valores "*crisp*" encontrados através da defuzzificação e os valores "*crisp*" normalizados para cada atributo.

Tabela 18 - Agregação de números *fuzzy* de relevância dos atributos

Agregação dos números *fuzzy* de relevância									
Item	E1	.....	E15	Número *Fuzzy* Agregado			Valor *Crisp*	Valor Normal	%
P1	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,131	3,131	4	3,1306	0,0475	4,75%
P2	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,076	3,076	4	3,0764	0,0467	4,67%
P3	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,222	3,222	4	3,2218	0,0489	4,89%
P4	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,83	2,83	3,83	2,8305	0,0429	4,29%
P5	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,136	3,136	4	3,1364	0,0476	4,76%
P6	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,07	3,07	4	3,0703	0,0466	4,66%
P7	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,134	3,134	4	3,1345	0,0475	4,75%
P8	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,047	3,047	4	3,0469	0,0462	4,62%
P9	(1,00; 2,00; 3,00)	.....	(1,00; 2,00; 3,00)	1,455	2,455	3,441	2,4553	0,0372	3,72%
P10	(2,00; 3,00; 4,00)	.....	(3,00; 4,00; 4,00)	2,027	3,027	3,973	3,0273	0,0459	4,59%
P11	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,964	2,964	3,93	2,9636	0,0449	4,49%
P12	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,987	2,987	3,822	2,9874	0,0453	4,53%
P13	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,937	2,937	3,847	2,9369	0,0445	4,45%
P14	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,91	2,91	3,875	2,9100	0,0441	4,41%
P15	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,134	3,134	4	3,1345	0,0475	4,75%
P16	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,134	3,134	4	3,1345	0,0475	4,75%
P17	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,968	2,968	3,94	2,9682	0,0450	4,50%
P18	(1,00; 2,00; 3,00)	.....	(2,00; 3,00; 4,00)	1,758	2,758	3,662	2,7576	0,0418	4,18%
P19	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,897	2,897	3,866	2,8971	0,0439	4,39%
P20	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,097	3,097	3,936	3,0970	0,0470	4,70%
P21	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,121	3,121	4	3,1211	0,0473	4,73%
P22	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,893	2,893	3,877	2,8934	0,0439	4,39%
							Total	**Total**	**Total**
							65,9313	1,0000	100,00%

Observa-se que a média dos valores normalizados é 4,55% e o desvio padrão é de 0,26%. Logo, podemos evidenciar que os atributos têm importâncias bastante similares e não

podem ser escolhidos exclusivamente ou descartados no momento do desenvolvimento do supervisório.

Tabela 19 - Agregação de números *fuzzy* de presença dos atributos

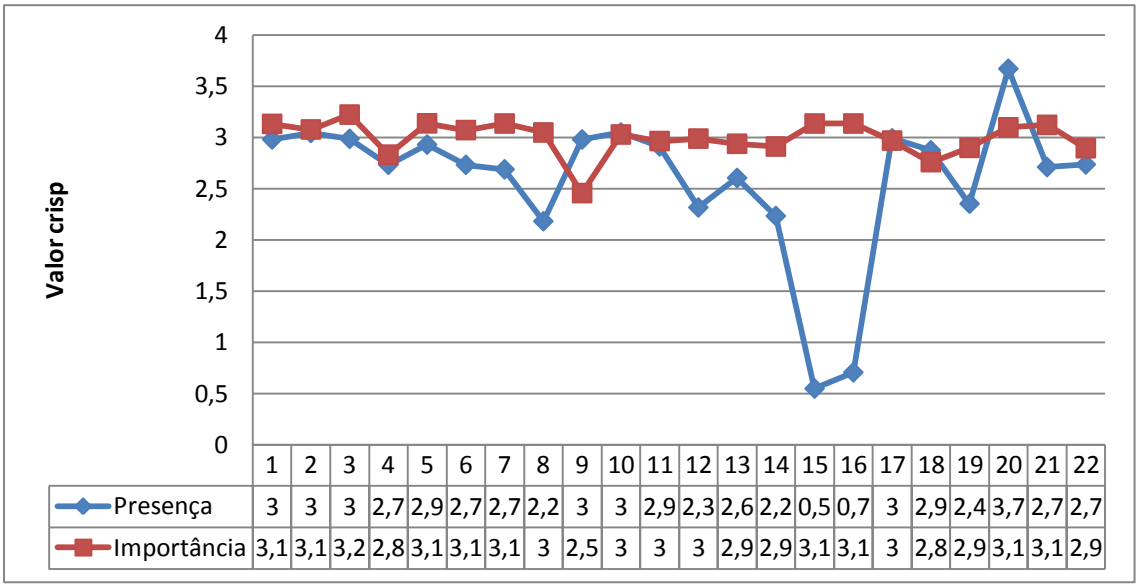
Agregação dos números *fuzzy de presença*							
Item	E1	.....	E15	Número Fuzzy Agregado			Valor Crisp
P1	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,979	2,979	3,948	2,9789
P2	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,042	3,042	3,923	3,0424
P3	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,986	2,986	3,946	2,9857
P4	(3,00; 4,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,731	2,731	3,702	2,7314
P5	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,929	2,929	3,929	2,9288
P6	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,732	2,732	3,712	2,7317
P7	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,688	2,688	3,649	2,6875
P8	(3,00; 4,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,181	2,181	3,153	2,1808
P9	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,979	2,979	3,919	2,9788
P10	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	2,047	3,047	4	3,0469
P11	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,911	2,911	3,741	2,9108
P12	(1,00; 2,00; 3,00)	.....	(1,00; 2,00; 3,00)	1,315	2,315	3,295	2,3155
P13	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,604	2,604	3,604	2,6036
P14	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,232	2,232	3,232	2,2324
P15	(0,00; 0,00; 1,00)	.....	(0,00; 1,00; 2,00)	0	0,548	1,548	0,5485
P16	(0,00; 1,00; 2,00)	.....	(0,00; 1,00; 2,00)	0	0,707	1,707	0,7068
P17	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,99	2,99	3,912	2,9898
P18	(3,00; 4,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,873	2,873	3,779	2,8729
P19	(2,00; 3,00; 4,00)	.....	(1,00; 2,00; 3,00)	1,352	2,352	3,352	2,3521
P20	(1,00; 2,00; 3,00)	.....	(3,00; 4,00; 4,00)	2,669	3,669	3,944	3,6689
P21	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,71	2,71	3,71	2,7097
P22	(2,00; 3,00; 4,00)	.....	(2,00; 3,00; 4,00)	1,735	2,735	3,735	2,7352

Pode-se avaliar o valor m dos triângulos *fuzzy* Ñ = (a, m, b) do **padrão de qualidade** para interpretar a inserção destes atributos nos grupos “sem relevância”, “pouco relevante”, “moderadamente relevante”, “relevante” e “muito relevante”. A tabela 20 mostra essa interpretação.

Tabela 20 - Interpretação dos grupos de atributos de qualidade

Número do Atributo	Atributo	Interpretação
P1	Dinâmica do processo	87% Relevante e 13% Muito relevante
P2	Segurança de acesso	92% Relevante e 8% Muito relevante
P3	Armazenamento de dados	78% Relevante e 22% Muito relevante
P4	Configuração dos parâmetros	17% Moderadamente relevante e 83% Relevante
P5	Apreensibilidade	86% Relevante e 14% Muito relevante
P6	Apresentabilidade quanto à estética das telas e animações	93% Relevante e 7% Muito relevante
P7	Apresentabilidade quanto à quantidade de informações na tela	87% Relevante e 13% Muito relevante
P8	Apresentabilidade quanto à hierarquia das telas	95% Relevante e 5% Muito relevante
P9	Apresentabilidade quanto aos desenhos dos elementos do sistema	55% Moderadamente relevante e 45% Relevante
P10	Status das variáveis	97% Relevante e 3% Muito relevante
P11	Recursos gráficos	4% Moderadamente relevante e 96% Relevante
P12	Documentação	2% Moderadamente relevante e 98% Relevante
P13	Quantidade e prioridade de alarmes	7% Moderadamente relevante e 93% Relevante
P14	Visualização e intervenção de alarmes	9% Moderadamente relevante e 91% Relevante
P15	Comunicação com o CLP	87% Relevante e 13% Muito relevante
P16	Configuração de tags e drivers de comunicação	87% Relevante e 13% Muito relevante
P17	Quantidade de variáveis	4% Moderadamente relevante e 96% Relevante
P18	Organização de *tags*	25% Moderadamente relevante e 75% Relevante
P19	Quantidade de scripts	11% Moderadamente relevante e 89% Relevante
P20	Erros de scripts	90% Relevante e 10% Muito relevante
P21	Configuração de Telas	88% Relevante e 12% Muito relevante
P22	Tempo de desenvolvimento	11% Moderadamente relevante e 89% Relevante

O gráfico de valores *crisp* de presença e de importância dos atributos (antes da normalização) pode ser visualizado no gráfico 4. Pode-se estabelecer visualmente a comparação entre a presença dos atributos e o padrão de qualidade (importância dos atributos).



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Presença	3	3	3	2,7	2,9	2,7	2,7	2,2	3	3	2,9	2,3	2,6	2,2	0,5	0,7	3	2,9	2,4	3,7	2,7	2,7
Importância	3,1	3,1	3,2	2,8	3,1	3,1	3,1	3	2,5	3	3	3	2,9	2,9	3,1	3,1	3	2,8	2,9	3,1	3,1	2,9

Gráfico 4 - Valores *crisp* de presença e importância dos atributos

5.3 Índice de Qualidade de cada atributo

O cálculo do índice de qualidade de cada atributo foi realizado através da redefinição das funções características de importância dos atributos de qualidade. Lembrando que esta transformação tem o objetivo adequar o cálculo do índice quando os números *fuzzy* de presença forem maiores que os de importância. As funções foram redefinidas conforme a tabela 21.

Tabela 21 - Redefinição dos números *fuzzy* de importância

Número do Atributo	**Número *fuzzy* triangular**	**Número *fuzzy* trapezoidal**
P1	Ñ = (2,13 ; 3,13 ; 4)	Q = (2,13 ; 3,13 ; 4 ; 4)
P2	Ñ = (2,07 ; 3,07 ; 4)	Q = (2,07 ; 3,07 ; 4 ; 4)
P3	Ñ = (2,22 ; 3,22 ; 4)	Q = (2,22 ; 3,22 ; 4 ; 4)
P4	Ñ = (1,83 ; 2,83 ; 3,83)	Q = (1,83 ; 2,83 ; 4 ; 4)
P5	Ñ = (2,13 ; 3,13 ; 4)	Q = (2,13 ; 3,13 ; 4 ; 4)
P6	Ñ = (2,07 ; 3,07 ; 4)	Q = (2,07 ; 3,07 ; 4 ; 4)
P7	Ñ = (2,13 ; 3,13 ; 4)	Q = (2,13 ; 3,13 ; 4 ; 4)
P8	Ñ = (2,04 ; 3,04 ; 4)	Q = (2,04 ; 3,04 ; 4 ; 4)
P9	Ñ = (1,45 ; 2,45 ; 3,44)	Q = (1,45 ; 2,45 ; 4 ; 4)
P10	Ñ = (2,02 ; 3,02 ; 3,97)	Q = (2,02 ; 3,02 ; 4 ; 4)
P11	Ñ = (1,96 ; 2,96 ; 3,92)	Q = (1,96 ; 2,96 ; 4 ; 4)
P12	Ñ = (1,98 ; 2,98 ; 3,82)	Q = (1,98 ; 2,98 ; 4 ; 4)
P13	Ñ = (1,93 ; 2,93 ; 3,84)	Q = (1,93 ; 2,93 ; 4 ; 4)
P14	Ñ = (1,91 ; 2,91 ; 3,87)	Q = (1,91 ; 2,91 ; 4 ; 4)
P15	Ñ = (2,13 ; 3,13 ; 4)	Q = (2,13 ; 3,13 ; 4 ; 4)
P16	Ñ = (2,13 ; 3,13; 4)	Q = (2,13 ; 3,13; 4 ; 4)
P17	Ñ = (1,96 ; 2,96 ; 3,93)	Q = (1,96 ; 2,96 ; 4 ; 4)
P18	Ñ = (1,75 ; 2,75 ; 3,66)	Q = (1,75 ; 2,75 ; 4 ; 4)
P19	Ñ = (1,89 ; 2,89 ; 3,86)	Q = (1,89 ; 2,89 ; 4 ; 4)
P20	Ñ = (2,09 ; 3,09 ; 3,93)	Q = (2,09 ; 3,09 ; 4 ; 4)
P21	Ñ = (2,12 ; 3,12 ; 3,37)	Q = (2,12 ; 3,12 ; 4 ; 4)
P22	Ñ = (1,89 ; 2,89 ; 3,37)	Q = (1,89 ; 2,89 ; 4 ; 4)

Com a perspectiva de avaliar se o supervisório desenvolvido contém os elementos constituintes de um *software* de qualidade, o número *fuzzy* trapezoidal do padrão de qualidade

(PQ) pode ser comparado com o número *fuzzy* triangular de presença dos atributos através da equação 10. Os índices de qualidade podem ser visualizados e analisados na tabela 22.

Tabela 22 - Índices de qualidade por atributo

Atributos	**PQ para Supervisório**	**Avaliação do Supervisório**	**Índice de Qualidade**
P1 - Dinâmica do processo	Ñ = (2,13 ; 3,13 ; 4)	Ñ = (1,97; 2,97; 3,94)	0,8517
P2 - Segurança de acesso	Ñ = (2,07 ; 3,07 ; 4)	Ñ = (2,04; 3,04; 3,92)	0,9641
P3 - Armazenamento de dados	Ñ = (2,22 ; 3,22 ; 4)	Ñ = (1,98; 2,98; 3,94)	0,7736
P4 – Configuração dos parâmetros	Ñ = (1,83; 2,83; 3,83)	Ñ = (1,73; 2,73; 3,70)	0,9019
P5 – Apreensibilidade	Ñ = (2,13 ; 3,13 ; 4)	Ñ = (1,92; 2,92; 3,92)	0,8031
P6 - Apresentabilidade quanto à estética das telas e animações	Ñ = (2,07 ; 3,07 ; 4)	Ñ = (1,73; 2,73; 3,71)	0,6871
P7 - Apresentabilidade quanto à quantidade de informações na tela	Ñ = (2,13 ; 3,13 ; 4)	Ñ = (1,68; 2,68; 3,64)	0,5962
P8 - Apresentabilidade quanto à hierarquia das telas	Ñ = (2,04 ; 3,04 ; 4)	Ñ = (1,18; 2,18; 3,15)	0,3145
P9 - Apresentabilidade quanto aos desenhos dos elementos do sistema	Ñ = (1,45; 2,45; 3,44)	Ñ = (1,97; 2,97; 3,91)	1
P10 – Status das variáveis	Ñ = (2,02; 3,02; 3,97)	Ñ = (2,04; 3,04; 4)	1
P11 – Recursos gráficos	Ñ = (1,96; 2,96; 3,92)	Ñ = (1,91; 2,91; 3,74)	0,9431
P12 – Documentação	Ñ = (1,98; 2,98; 3,82)	Ñ = (1,31; 2,31; 3,29)	0,4362
P13 - Quantidade e prioridade de alarmes	Ñ = (1,93; 2,93; 3,84)	Ñ = (1,60; 2,60; 3,60)	0,6944
P14 - Visualização e intervenção de alarmes	Ñ = (1,91; 2,91; 3,87)	Ñ = (1,23; 2,23; 3,23)	0,4372
P15 – Comunicação com o CLP	Ñ = (2,13 ; 3,13 ; 4)	Ñ = (0 ; 0,54 ; 1,54)	0
P16 - Configuração de tags e drivers de comunicação	Ñ = (2,13 ; 3,13; 4)	Ñ = (0 ; 0,70 ; 1,70)	0
P17 – Quantidade de variáveis	Ñ = (1,96; 2,96; 3,93)	Ñ = (1,99; 2,99; 3,91)	1
P18 – Organização de *tags*	Ñ = (1,75; 2,75; 3,66)	Ñ = (1,87; 2,87; 3,77)	1
P19 – Quantidade de scripts	Ñ = (1,89; 2,89; 3,86)	Ñ = (1,35; 2,35; 3,35)	0,5292
P20 - Erros de scripts	Ñ = (2,09; 3,09; 3,93)	Ñ = (2,66; 3,66; 3,94)	1
P21 – Configuração de Telas	Ñ = (2,12; 3,12; 3,37)	Ñ = (1,71; 2,71; 3,71)	0,6309
P22 - Tempo de desenvolvimento	Ñ = (1,89; 2,89; 3,37)	Ñ = (1,73; 2,73; 3,73)	0,8480

Os resultados da tabela 22 devem ser interpretados sob a perspectiva dos critérios de julgamento expostos no capítulo 4, ou seja, quando o índice de qualidade estiver abaixo de 80% é necessário melhorar os aspectos do *software* que se referem àquele atributo. Portanto, nota-se que os atributos Dinâmica do processo (P1), Segurança de acesso (P2), Configuração dos parâmetros (P4), Apreensibilidade (P5), Recursos gráficos (P11) e Tempo de desenvolvimento (P22) estão acima do nível 80%, então, eles estão atendendo ao padrão de qualidade (PQ) para supervisórios proposto neste trabalho.

Estes resultados mostram que o modelo de desenvolvimento de supervisórios proposto contribuiu bastante para o programador compreender e implementar a dinâmica do processo industrial. Pode-se inferir também que o tempo de desenvolvimento, cerca de três meses, foi adequado para o projeto de acordo com os especialistas. Verifica-se ainda que a apreensibilidade teve uma boa avaliação, ou seja, os avaliadores sentiram facilidade em operar o supervisório.

As orientações de cadastramento de usuários com diferentes níveis de acesso, que constam em um tópico completo do modelo de desenvolvimento de supervisórios facilitaram o cadastramento e a configuração das permissões para cada usuário.

Os recursos gráficos receberam uma boa avaliação e atenderam completamente ao padrão de qualidade principalmente devido ao tópico de elaboração dos gráficos de tendência do modelo de desenvolvimento que serviu de apoio ao programador. Os gráficos foram elaborados numa tela única e organizada.

Alguns atributos chegaram ao grau de excelência em qualidade com índice de qualidade de valor 100%, são eles: Apresentabilidade quanto aos desenhos dos elementos do sistema (P9), Status das variáveis (P10), Quantidade de variáveis (P17), Organização de tags (P18) e Erros de scripts (P20). Estes atributos não devem ser alterados em nenhum ponto.

Com este resultado, pode-se verificar que a ideia de fazer o planejamento e a organização das variáveis através de uma tabela que organize os dados de entrada e saída (proposta do modelo de desenvolvimento de supervisórios) é eficiente. Isto foi evidenciado através dos bons resultados que a simulação, a organização e o status das variáveis proporcionaram à qualidade do supervisório.

A organização de variáveis é um dos atributos mais difíceis de ser aprendido no desenvolvimento de supervisórios, segundo especialistas. Portanto, o modelo forneceu uma boa ferramenta para aprendizagem de como organizar variáveis. As tabelas 23 e 24 mostram a organização das variáveis de entrada e saída, respectivamente, realizada pelo programador.

Tabela 23 - Organização das variáveis de entrada do supervisório

Endereçamento no CLP	Endereçamento das tags	Tipo de sensor	Funcionalidade
I:0.0	Boia_01	Sensor óptico de presença	Sensor digital que detecta o nível baixo do reator químico
I:0.1	Boia_02	Sensor óptico de presença	Sensor digital que detecta o nível médio do reator químico
I:0.2	Boia_03	Sensor óptico de presença	Sensor digital que detecta o nível alto do reator químico
I:0.3	SN1	Sensor óptico de presença	Sensor digital que indica se o tanque está na posição correta de envasilhamento
I:0.4	SN2	Sensor óptico de presença	Sensor digital que indica se o tanque pode ser retirado e pode ser usado para contagem dos tanques cheios
IW:1.0	Termômetro máquina de vapor	Sensor de temperatura	Sensor analógico que mede a temperatura da máquina de vapor
IW:2.0	Termômetro reator	Sensor de temperatura	Sensor analógico que mede a temperatura do reator

Tabela 24 - Organização das variáveis de saída do supervisório

Endereçamento no CLP	Endereçamento das tags	Tipo de sensor	Funcionalidade
O:1.0	Esteira	Motor da esteira	Atuador responsável pela movimentação da esteira
O:1.1	Misturador	Motor do misturador	Atuador responsável pela rotação do misturador
O:1.2	Válvula 1	Eletroválvula	Atuador que permite a passagem de líquido do reservatório 2 para o reator
O:1.3	Válvula 2	Eletroválvula	Atuador que permite a passagem de líquido do reservatório 1 para o reator
O:1.4	Válvula 3	Eletroválvula	Atuador que permite a passagem de vapor da máquina de vapor para o reator
O:1.5	Dreno	Eletroválvula	Atuador que permite a passagem do produto da reação para o tanque

Os *scripts* também atenderam integralmente ao padrão de qualidade. Os atributos que tiveram índices de qualidade em nível médio (50% a 79%), comparando com os critérios de aceitação ou julgamento mostrados na figura 34, foram: Armazenamento de dados (P3), Apresentabilidade quanto à estética das telas e animações (P6), Apresentabilidade quanto à quantidade de informações na tela (P7), Quantidade e prioridade de alarmes (P13), Quantidade de Scripts (P19), Configuração de Telas (P21). Nota-se que estes atributos precisam de alguns ajustes no supervisório para atingir o padrão de qualidade.

Este resultado do armazenamento de dados deve-se também às orientações do tópico de planejamento de históricos, relatórios e banco de dados. O programador utilizou o modelo para elaborar as telas de históricos, relatórios e guardar os dados em arquivos próprios do ambiente de desenvolvimento.

O modelo de desenvolvimento de supervisórios também disponibilizou muitos exemplos de telas para o programador. Estes exemplos ajudaram-no a desenvolver mais rapidamente as telas e a interação entre elas. Porém, é necessário continuar consultando os especialistas sobre qual é a forma de se obter excelência para este atributo.

O supervisório também teve um bom desempenho em mostrar os elementos gráficos (desenhos, animações, *layout* das telas) que representam a planta. Entretanto, é preciso enfatizar mais ao programador a necessidade de não ter excesso nem ausência de informações na tela.

Nota-se também que a tabela de quantidade e prioridade de alarmes proposta no modelo e elaborada pelo programador facilitou bastante a criação dos alarmes e de suas prioridades. As orientações de como descobrir quais são os alarmes mais importantes do processo também auxiliaram o programador a construir a tabela. A tabela 25, criada pelo programador, mostra como os alarmes foram organizados no supervisório.

O atributo “quantidade de scripts” teve um resultado mediano principalmente devido ao rigor que a maioria dos especialistas experientes (conhecedores de diversas intruções do ambiente de desenvolvimento) tem quando vão fazer um supervisório. Eles utilizam bem mais scripts que um programador iniciante.

Tabela 25 - Organização dos alarmes do supervisório

Alarme	Prioridade	Reconhecimento	Solução
Perigo: Temperatura elevada na máquina de vapor	100%	Imediato	Fechar válvula de entrada de vapor, verificar imediatamente se o sensor está medindo corretamente e acionar equipe de manutenção.
Superaquecimento do reator químico	100%	Imediato	Fechar válvula de entrada de vapor, verificar se o sensor está medindo corretamente e acionar equipe de manutenção.
Válvulas com defeito	50%	Diário	Verificar qual é o defeito e, se necessário, trocar as válvulas.
Posição de tanque incorreta	40%	Diário	Verificar se a medição do sensor SN1 está correta e reposicionar o tanque.
Problemas com a esteira ou motor	40%	Diário	Verificar qual é o problema com a esteira e/ou o motor.

Os atributos seguintes foram problemáticos no supervisório. São eles: Apresentabilidade quanto à hierarquia das telas (P8), Documentação (P12), Visualização e intervenção de alarmes (P14). Estes atributos realmente devem ser revistos e melhorados no supervisório desenvolvido. Estes resultados são importantíssimos para o programador verificar as deficiências do supervisório e poder melhorá-lo mais rapidamente.

O resultado negativo para o atributo "Documentação" deve-se a entrega incompleta dos documentos que descrevem informações do supervisório e processo. Os estudantes que desenvolveram o supervisório terminaram o curso e abandonaram a elaboração da documentação do usuário. Logo, somente alguns documentos forem entregues para a avaliação dos especialistas.

Houve uma má interpretação, por parte dos especialistas, do atributo "Apresentabilidade quanto à hierarquia das telas". A grande maioria dos especialistas entendeu o atributo hierarquia de telas como um recurso presente em alguns ambientes de desenvolvimento de supervisórios. No software desenvolvido no Elipse Scada não existe este recurso, então a maioria avaliou negativamente a presença do atributo.

Alguns atributos foram avaliados como totalmente ausentes no supervisório. São eles: Comunicação com o CLP (P15) e Configuração de *tags* e *drivers* de comunicação (P16).

Este resultado já estava sendo esperado, pois o desenvolvimento teve algumas limitações como:

- Ausência de dispositivo físico (controlador lógico programável) para fazer a integração com o supervisório;
- Ausência da planta do processo industrial para fornecer os valores reais das variáveis.

O método de avaliação atestou a ausência desses atributos (vide gráfico 5) e consolidou sua qualidade na determinação de deficiências e eficiências em um supervisório desenvolvido. Pode-se dizer ainda que 50% dos atributos atenderam ao padrão de qualidade, porém a tomada de decisões tem que levar em consideração o índice de cada atributo, pois uma falha pode ocasionar sérios problemas futuramente.

O gráfico 5 mostra os resultados dos índices de qualidade calculados para cada atributo para que se possa ter uma visão mais ampla do atendimento aos requisitos do padrão de qualidade.

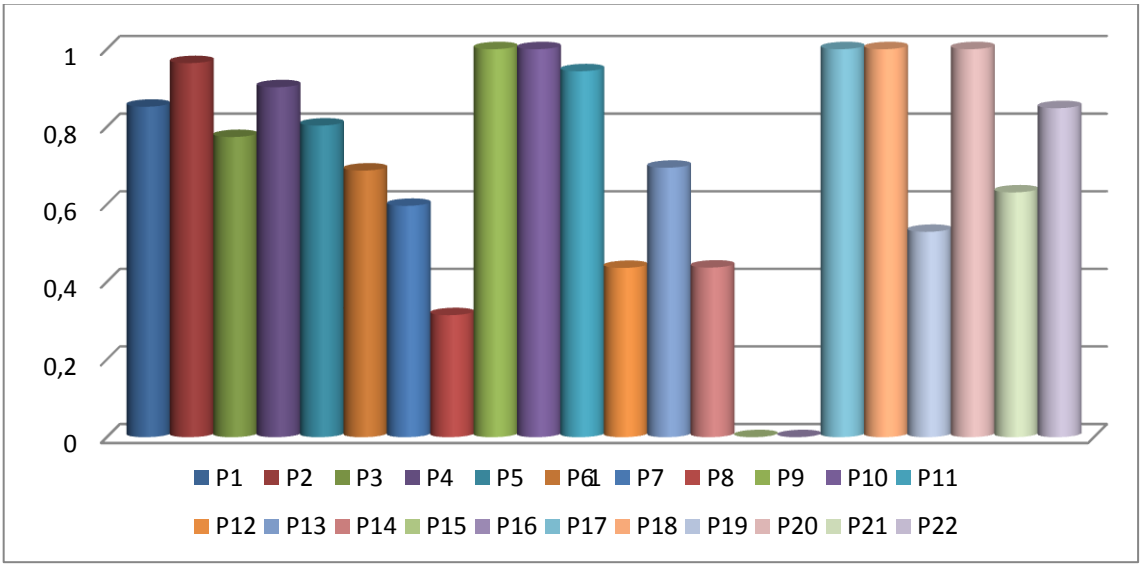


Gráfico 5 - Índices de qualidade dos atributos

5.4 Índice de Qualidade total do supervisório

Existe uma medida que auxilia na tomada de decisões, principalmente em determinar o momento em que o supervisório será entregue. Este valor é o índice de qualidade total do supervisório.

Este índice foi calculado utilizando os índices de qualidade para cada atributo, ponderados pelos valores normalizados de importância de cada atributo. O valor do índice de qualidade total, calculado pela equação 12, é mostrado a seguir.

$$\mathbf{I_{Qual} = 69,48\%}$$

Portanto, verifica-se que o supervisório atende em 69,48% o padrão de qualidade proposto neste trabalho. Logo, presume-se que grande parte dos critérios de qualidade está presente no *software* desenvolvido. Contudo, fazendo uma comparação com os critérios de julgamento percebe-se que o supervisório não pode ainda ser entregue ao cliente.

Se a maioria dos atributos não fosse modificada no supervisório e os atributos Comunicação com o CLP (P15) e Configuração de *tags* e *drivers* de comunicação (P16), que proporcionam segurança ao sistema, atendessem a 95% do padrão de qualidade (o que acontece na grande maioria dos projetos), o índice de qualidade total seria de 79%, ou seja, esta seria a porcentagem de qualidade do supervisório se não houvesse as limitações desse projeto.

Observa-se que, mesmo se o índice de qualidade total fosse maior ou igual a 80%, o supervisório não poderia ser entregue porque existem índices de atributos menores que 80%. Lembrando que os índices devem, obrigatoriamente, ser usados em conjunto.

6. CONCLUSÕES

A área de Qualidade de *Software* é essencial para o desenvolvimento da supervisão em processos industriais e é um grande desafio inserir os conceitos e as metodologias de qualidade nesta seara industrial.

O trabalho propõe uma solução para o seguinte problema de pesquisa: Como definir um padrão de qualidade através da identificação de quais são os atributos e as características fundamentais de um *software* de supervisão? Esta solução foi baseada no estudo na norma ISO/IEC 9126 que estabelece algumas características e atributos de qualidade de um *software*.

A norma ISO/IEC 9126 propiciou a identificação das principais características e atributos de qualidade de um supervisório e a norma ISO/IEC 14598 serviu de base para a elaboração de uma metodologia de avaliação da qualidade desses *softwares*, que mostra de forma qualitativa e quantitativa o estado atual do supervisório.

Três características foram selecionadas através da norma (funcionalidade, usabilidade e confiabilidade) e mais uma foi proposta nesta pesquisa (configuração) pela necessidade de avaliar a configuração de parâmetros. Vinte e dois atributos foram escolhidos pelo coordenador da pesquisa e pelos especialistas participantes, que opinaram a respeito da inserção de novos atributos e também a respeito do grau de relevância que cada atributo possui.

A pesquisa de campo foi realizada através de questionários, que constituíram uma boa contribuição para detectar os especialistas que possuem mais experiência e para identificar o grau de relevância e de presença dos atributos e características de qualidade de um supervisório. Diante disso, conclui-se que um padrão de qualidade de supervisórios foi elaborado e validado com êxito.

Os modelos de desenvolvimento de *software* como os tradicionais e o ágil proporcionaram o embasamento teórico para a elaboração de um modelo de desenvolvimento de supervisórios que contivesse os atributos considerados fundamentais para a qualidade do *software*, solucionando outro problema de pesquisa.

O modelo foi utilizado pelos programadores para desenvolver um supervisório com o objetivo de servir como estudo de caso para verificar os benefícios do modelo. A avaliação do supervisório mostrou que a maioria dos atributos apresentou uma boa qualidade.

O protótipo forneceu uma boa visão inicial do supervisório. O *feedback* do cliente nos momentos de apresentação das versões intermediárias facilitou bastante o preenchimento dos requisitos do projeto. O planejamento prévio do projeto proporcionou mais organização e eficiência no desenvolvimento. Logo, pode-se concluir que o modelo atendeu às expectativas de guiar um programador no desenvolvimento de um supervisório.

A proposta de desenvolver uma metodologia de avaliação da qualidade de um supervisório foi realizada através do arcabouço teórico da lógica *fuzzy*. A abordagem *fuzzy* forneceu uma ferramenta capaz de interpretar resultados de medição de *software*, e auferiu uma base matemática capaz de realizar a análise de opiniões subjetivas sujeitas a incertezas, coletadas a partir da opinião de 15 especialistas no desenvolvimento de supervisórios. Alguns trabalhos serviram de exemplos para a elaboração da metodologia e novas contribuições foram realizadas como a criação de um índice de qualidade total baseado na média dos índices de qualidade ponderada pelos graus de relevância dos atributos.

A avaliação apontou as falhas, as correções que devem ser feitas e os atributos que atenderam ou não ao padrão de qualidade. Os índices serviram como boas ferramentas no momento de avaliar qual era o grau de qualidade dos atributos e do *software* completo. Em geral, o supervisório atingiu um bom nível de qualidade (aproximadamente 69% do padrão de qualidade) e os requisitos e atributos que chegaram a um nível alto de qualidade podem servir de exemplo para os próximos projetos.

A avaliação identificou, mesmo com os bons índices alcançados, atributos que estavam ausentes no supervisório, o que permitiu validar a ferramenta, já que aqueles atributos realmente não foram inseridos no *software* devido a limitações de componentes físicos do processo.

Alguns atributos tiveram resultados negativos por motivos de falta de comunicação entre especialistas e inexperiência do programador. Deve-se ter mais atenção a esses problemas no momento do desenvolvimento e da avaliação.

Observa-se que a avaliação e o desenvolvimento têm melhores resultados quando os avaliadores podem colocar observações a respeito dos atributos que estão com falhas e quando eles têm conhecimento do padrão de qualidade de cada atributo antes de avaliar. As observações facilitam a correção dos erros de cada atributo e a consciência do padrão de qualidade facilita a decisão do avaliador em liberar ou não a entrega daquele atributo. A ausência desses procedimentos é um ponto fraco da pesquisa.

O procedimento padrão adotado por empresas de automação nos dias de hoje consiste em um simples *checklist* de requisitos; o presente trabalho, portanto, agrega utilidade para os profissionais e para a indústria, mostrando-se uma contribuição significativa.

Atualmente, os estudantes do curso superior de Mecatrônica Industrial do IFCE, Campus Limoeiro do Norte, estão aprendendo a desenvolver supervisórios através do modelo proposto neste trabalho e nota-se que os índices de qualidade dos *softwares* desenvolvidos com o método proposto são melhores quando comparados aos métodos da engenharia de *software*. A avaliação dos supervisórios desenvolvidos na disciplina de sistemas de supervisão, realizada pelos professores, também se tornou mais prática, porque são avaliados requisitos bem definidos e os questionários são preenchidos rapidamente.

6.1 Trabalhos futuros

A utilização de definições, elementos e metodologias, da área de Qualidade de *Software*, em aplicações industriais ainda é incipiente no Brasil. Muitos trabalhos podem ser feitos para minimizar os problemas desta área. Entre esses trabalhos pode-se citar:

- Elaboração de um *software* que automatize a metodologia de avaliação da qualidade de supervisórios desenvolvida de forma a tornar mais rápida a geração de resultados.
- Utilização do padrão de qualidade e da metodologia de avaliação em supervisórios que serão instalados em processos industriais reais.
- Utilização do modelo de desenvolvimento de supervisórios em projetos na indústria e em disciplinas como redes industriais, sistemas de supervisão, sistemas de controle distribuído e sistemas SCADA.
- Desenvolvimento de métodos e testes de avaliação de processo para sistemas supervisórios.
- Aplicação de testes de *software* em supervisórios para detectar os níveis de qualidade da característica usabilidade.
- Desenvolvimento de um ambiente de avaliação, através da utilização de redes de computadores e Internet, que forneça suporte para os especialistas avaliadores.
- Selecionar ou elaborar novas métricas que possam gerar novos resultados a respeito da qualidade do supervisório.
- Escrever livros sobre sistemas SCADA e *softwares* supervisórios.

- Avaliar o aprendizado de estudantes de Engenharia através do método de avaliação de supervisórios.
- Avaliar entre os estudantes a facilidade de uso do modelo de desenvolvimento de supervisórios.
- Aplicar outros modelos de desenvolvimento de *software* no ciclo de vida de um supervisório para estabelecer comparações entre modelos.

REFERÊNCIAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Guia para Utilização das Normas sobre Avaliação de Qualidade de Produto de *Software***: ISO/IEC 9126 e ISO/IEC 14598. 1999.

ALBUQUERQUE, Pedro Urbano B. de; ALEXANDRIA, Auzuir Ripardo de. **Redes Industriais:** Aplicações em Sistemas Digitais de Controle Distribuído. 2ª edição. São Paulo: Ensino Profissional, 2009.

ALMEIDA, Ricardo Augusto de. **Desenvolvimento de um Sistema de Controle Remoto para Acionamento e monitoramento de Processos Químicos - CRAPQ**. 2010. 114 f. Dissertação (Mestrado em Engenharia Química) - Faculdade de Engenharia Química, Universidade de Campinas, Campinas, 2010.

AIHARA, C. K. et al. Desenvolvimento de Aplicativos para Monitoramento de variáveis de controle de processos industriais. **1ª Escola Brasileira de Aplicações em Dinâmica e Controle**, São Carlos, 2001.

ARRUDA, T. A.; PEREIRA, G. A. S. Arquitetura de *Hardware* e *Software* para Supervisão e Controle de um Carro Autônomo. **Congresso Brasileiro de Automática**, Campina Grande, 2012.

BELCHIOR, A. D. **Um modelo *fuzzy* para Avaliação da Qualidade de *Software***. 1997. 208 f. Tese (Doutorado em Engenharia de Sistemas e Computação) - Departamento de Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 1997.

BOENTE, A. N. P. **Um modelo *fuzzy* para avaliação da qualidade de produtos de *software* e da satisfação dos gerentes de projetos numa Fundação Pública Estadual**. 2009. 211 f. Dissertação (Mestrado em Administração e Desenvolvimento Empresarial) - Universidade Estácio de Sá, Rio de Janeiro, 2009.

CÁRDENAS, E. H. **Geração Genética Multiobjetivo de Sistemas *Fuzzy* Usando a Abordagem Iterativa**. 2011. 132 f. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de São Carlos, São Paulo, 2011.

CARULO, M. F. **Desenvolvimento e Implementação de Controlador *fuzzy* aplicado à produção de biodiesel**. 2008. 196 f. Tese (Doutorado em Engenharia Quimica) - Universidade Estadual de Campinas, Campinas, São Paulo, 2008.

CAPABILITY MATURITY MODEL INTEGRATION (CMMI). ***Version 1.1 - Continuous Representation***. *Technical Report CMU/SEI-2002-TR-011, Software Engineering Institute*, Carnegie Mellon University, Pittsburgh/PA - USA, 2002.

COSTA, B. S. J. **Ambiente para desenvolvimento de aplicações *fuzzy* industriais**. 2010. 91 f. Dissertação (Mestrado em Engenharia Elétrica e de Computação) – Universidade Federal do Rio Grande do Norte, Natal, 2010.

COUGHANOWR, D. R.; LEBLANC, S. E. ***Process Systems Analysis and Control***. *Third edition*. McGraw-Hill's, 2009.

CURZEL, J. L. **Síntese e Implementação de Controle Supervisório em uma Célula Flexível de Manufatura Didática**. 2008. 106 f. Dissertação (Mestrado em Engenharia Elétrica) – Universidade do Estado de Santa Catarina, Joinville, 2008.

ELIPSE. **Demonstração do Elipse SCADA**. Disponível em: http://www.elipse.com.br. Acesso em: 14 out. 2012.

ENGELBRECHT, A. P. ***Computational Intelligence****: An Introduction. Second Edition. South Africa*: WILEY, 2007.

FERREIRA JÚNIOR, P. A. **Eficiência Energética em Ambientes Prediais utilizando rede sem fio zigbee e Controle Fuzzy**. 2009. 111 f. Dissertação (Mestrado em Engenharia Elétrica) - Faculdade de Tecnologia da Universidade de Brasília, Distrito Federal, 2009.

FERNANDES, J. H. C. Qual a prática do desenvolvimento de software? **Ciência e Cultura**, Vol. 55, n. 2, jun. 2003.

GIANI, A. et al. *A testbed for secure and robust SCADA systems.* ***ACM SIGBED Review***, v.5, nº.2, 2008.

GROOVER, M. P. **Automação Industrial e Sistemas de Manufatura**, 3ª edição, São Paulo: Pearson, 2010.

IEEE. **IEEE 1061:** *Standard for a Software Quality Metrics Methodology*. *IEEE Standards Dept*, 1998.

ISO/IEC 14598. ***Information technology****: software product evaluation*, 1998.

ISO 9000-3, *Quality Management and Quality Assurance Standards - Part 3: Guidelines for the Application of ISO9001 to the Development, Supply and Maintenance of Software*, ISO, September, 1990.

ISO/IEC 9126. ***Information technology****: software product quality*, 1998.

ISO/IEC 12207. ***Information Technology:*** *Software Life Cycle Processes*, 2008.

ISO/IEC 15939. ***Software Engineering****: Software Measurement Process*, 2002.

KAN, S. H. ***Metrics and Models in Software Quality Engineering.*** *Second Edition.* Addison-Wesley, 2003.

KOSCIANSKY, A.; SOARES, M. S. Qualidade de Software. 2ª edição. São Paulo: Novatec, 2006.

MALVEZZI, W. R. **Uma ferramenta baseada em teoria** *fuzzy* **para o acompanhamento de alunos aplicado ao modelo de educação presencial mediado por tecnologia**. 2010. 125 f. Dissertação (Mestrado em Engenharia Elétrica) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2010.

MARCOS, I. P. M. **Metodologia para Monitoramento Inteligente de Condição de Máquina:** Uma abordagem usando funções de pertinência *fuzzy*. 2012. 175 f. Dissertação (Mestrado em Engenharia Mecânica) – Faculdade de Tecnologia da Universidade de Brasília, Brasília, 2012.

MARTINEZ, M. R. M. et al. *The Software Product Evaluation Data Base – Supporting MEDE-PROS.* ***ISESS - International Software Engineering Standards Symposium****: Best Software Practices for the Internet age*, Curitiba. Anais, 1999. Curitiba, mai 1999.

MARTINS, G. M. **Princípios de Automação Industrial**. Universidade Federal de Santa Catarina. Departamento de Eletromecânica e Sistemas de Potência, Santa Catarina, 2012.

MARTINS, J. C. C. Técnicas para Gerenciamento de Projetos de Software. Rio de Janeiro: Brasport, 2007.

MAZZOLA, V. B. **Engenharia de Software e Sistema da Informação**. 3ª edição. Florianópolis, 2008.

MECENAS, I. ; OLIVEIRA, V. **Qualidade de Software**: Uma Metodologia para Homologação de Sistemas. Rio de Janeiro: Alta Books, 2005.

MENEGOTTO, J. **Aplicação da Teoria dos Conjuntos Fuzzy em Modelos Farmacocinéticos Multicompartimentais**. 2011. 140 f. Dissertação (Mestrado em Matemática Aplicada) – Universidade Estadual de Campinas, 2011.

MESQUITA, L. B. M. de. **Qteste – Ferramenta de Avaliação da Qualidade de Software**. 2013. 79 f. Monografia (Graduação em Engenharia de Teleinformática) – Universidade Federal do Ceará, 2013.

MORAES, C. C. de; CASTRUCCI, P. de L. **Engenharia de Automação Industrial**. 2ª edição. Rio de Janeiro: LTC, 2007.

MORÉ, J. D. **Aplicação da Lógica Fuzzy na avaliação da confiabilidade humana nos ensaios não destrutivos por ultra-som**. 2004. 218 f. Tese (Doutorado em Engenharia Metalúrgica e dos Materiais) – Universidade Federal do Rio de Janeiro, 2004.

MUNAKATA, T. *Fundamentals of the new artificial intelligence – Neural, Evolutionary, Fuzzy and more. Second Edition*. SPRINGER, 2008.

NASCIMENTO JUNIOR, C. L.; YONEYAMA, T. **Inteligência Artificial em Controle e Automação**. Edgard Blücher, 2000.

NBR ISO 9241-10. **Requisitos ergonômicos para trabalho de escritório com computadores**: Parte 10 – Princípios de diálogo. 2000.

NOSCHESE, A. S. **Redes de Automação**. SENAI-SP, 2006.

OLIVEIRA, K. R. **AdeQuas: Ferramenta fuzzy para avaliação da qualidade de software**. 2002. 128 f. Dissertação (Mestrado em Informática Aplicada) – Universidade de Fortaleza. Fortaleza, 2002

PEDRYCZ, W.; GOMIDE, F. *An introduction to fuzzy sets: analysis and design*. London, Bradford Book, 1998.

PEROZZO, R. F. **Criação de um framework para construção de sistemas supervisórios em dispositivos móveis**. 2007. 108 f. Dissertação (Mestrado em Ciências da Computação) –

Universidade Federal do Rio Grande do Sul, Porto Alegre, 2007.

PFLEEGER, S. L. **Engenharia de Software**: Teoria e Prática, 2ª Edição, Prentice Hall, 2004.

PINTO, F. da C. **Sistemas de Automação e Controle**. SENAI, Espírito Santo, 2005.

PIRES, P. S. M.; OLIVEIRA, L. A. H. G. de; BARROS, D. N. Aspectos de segurança em sistemas SCADA – Uma visão geral. **4º Congresso Internacional de Automação, Sistemas e Instrumentação**. Mai. 2005.

POLITO-BRAGA et. al. Aspectos Metodológicos para o Ensino de Projetos de Controle e Automação. **Congresso Brasileiro de Automática**, Campina Grande, 2012.

PRESSMAN, R. S. **Engenharia de Software**. 6ª edição. São Paulo: McGraw-Hill, 2006.

PROJETO ISO. **Projeto de equivalência à norma ISO 9241-210**: Ergonomia da Interação humano-sistema – Parte 210: Projeto Centrado no ser humano para sistemas interativos, 2011.

PURIFICAÇÃO, A. B. A da. **Uso da teoria dos conjuntos fuzzy para análise do risco de desenvolvimento de software**. Universidade Estácio de Sá. Rio de Janeiro, 2008.

REYNARD, S. et al. ***Flexible manufacturing cell SCADA system for educational purposes***. *Computer Applications in Engineering Education*, 2008.

RIBEIRO, M. A. **Automação Industrial**. Tek Treinamento & Consultoria Ltda. 4ª edição. Salvador, 1999.

RIBEIRO de SÁ, I. I. **Identificação das ações empresariais a partir da percepção do consumidor**: Uso da Teoria fuzzy. 2007. 119 f. Dissertação (Mestrado em Administração e Desenvolvimento empresarial) – Universidade Estácio de Sá, Rio de Janeiro, 2007.

ROCHA, A. R. C. **Um Modelo para Avaliação da Qualidade de Especificações**. 1983. Tese (Doutorado em Informática) - PUC, Rio de Janeiro, 1983.

ROCKWELL. **Demonstração do FactoryTalk**. Disponível em: www.rockwellautomation.com/pt/rockwellsoftware/**factorytalk**. Acesso em: 23 set. 2013.

RODRIGUES, L. G. M. **Um Modelo de Avaliação de Requisitos no Processo de Desenvolvimento de Software**. 2006. 84 f. Dissertação (Mestrado em Computação) – Universidade Estadual de Campinas, Campinas, 2006.

ROCHA, A. R. C., et.al. **Qualidade de Software**: Teoria e Prática. São Paulo: Prentice Hall, 2001.

ROSÁRIO, J. M. **Princípios de Mecatrônica**. 1ª edição. São Paulo: Pearson, 2005.

SANTOS, R. C. dos. **Desenvolvimento de uma metodologia para avaliação de usabilidade de sistemas utilizando a lógica *fuzzy* baseado na ISO**. 2007. 115 f. Dissertação (Mestrado em Economia e Finanças) – Faculdade de Economia e Finanças IBMEC, Rio de Janeiro, 2007a.

SANTOS, H. G. **Desenvolvimento de um Supervisório Modular para uma Célula Flexível de Manufatura**. 2007. 186 f. Dissertação (Mestrado em Engenharia Mecânica) – Universidade Federal de Santa Catarina, Florianópolis, 2007b.

SEIXAS, C. **Arquiteturas de sistemas de automação**: uma introdução. Disponível em: http://www.cpdee.ufmg.br/~seixas/PaginaII/Download/IIDownload.htm. Acesso em: 16 fev. 2004.

SILVA, R. E. F. **Implementação de um Módulo de Supervisão para um Sistema de Detecção de Vazamentos em dutos de Petróleo**. 2009. 68 f. Dissertação (Mestrado em Ciências) – Universidade Federal do Rio Grande do Norte, Natal, 2009.

SILVEIRA, G. P. **Métodos numéricos integrados à lógica fuzzy e método estocástico para a solução de EDP's**: uma aplicação à dengue. 2011. 181 f. Tese (Doutorado em Matemática Aplicada) – Universidade de Campinas. Campinas, 2011.

SIMÕES, M. G.; SHAW, I. S. **Controle e Modelagem Fuzzy**. 2ª edição. São Paulo: Blucher, 2007.

SOARES, L. C. **Sistema Supervisório para Poços de Petróleo Baseados no Método de Elevação Artificial Plunger Lif**. 2010. 65 f. Dissertação (Mestrado em Ciência e Engenharia de Petróleo), Universidade Federal do Rio Grande do Norte, 2010.

SOMMERVILLE, I. **Engenharia de Software**. 9ª edição. São Paulo: Pearson, 2011.

SOUZA, R. B. **Uma arquitetura para sistemas supervisórios industriais e sua aplicação em processos de elevação artificial de petróleo**. 2005. 71 f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Rio Grande do Norte, Natal, 2005.

THOMAS, M.; KUMAR, P.; CHANDNA, V. *Design, development and commissioning of a supervisory control and data acquisition* (SCADA) *laboratory for research & training*. ***Power Engineering Society General Meeting*** **IEEE**, Vol.1, 2004.

TORRES FILHO, F.; VIEIRA, M. F. Q. Abordagem Ontológica para Modelagem da IHM de Subestações Elétricas. **Congresso Brasileiro de Automática**. Campina Grande, 2012.

VIANNA, W. da S. **Sistema Scada Supervisório**. Campos dos Goytacazes. Rio de Janeiro, 2008.

ZADEH, L. A. ***Fuzzy Sets, Information and Control*** **8**, 1965.

ZADEH, L. A. *The birth and evolution of fuzzy logic*. ***I.B. Turksen, Ed., Proceeding of NAFIP'90****. June,* 1990.

GLOSSÁRIO

Avaliação da qualidade: Exame sistemático do quanto uma entidade é capaz de atender aos requisitos especificados.

Atributo / Constructo: Propriedade mensurável, física ou abstrata, de uma entidade. São elementos do software que podem ser medidos para cada característica.

Características de software: requisitos de qualidade de software definidos na norma NBR ISO/IEC 9126-1.

Controle supervisório: termo usado para designar o controle realizado por um aplicativo de supervisão baseado numa plataforma SCADA.

Dinâmica do processo: conjunto de atividades que compõem o processo produtivo.

Documentação de usuário: Conjunto completo de documentos, disponível na forma impressa ou não, que é fornecido para a utilização de um produto, sendo também uma parte integrante do produto.

Medida (substantivo): Número ou categoria atribuído a um atributo de uma entidade através de uma medição.

Medição: Uso de uma métrica para atribuir um valor (o qual pode ser um número ou categoria), obtido a partir de uma escala, a um atributo de uma entidade.

Métrica: Método e escala de medição definidos. Uma escala quantitativa e um método que pode ser usado para medição.

Modelo de qualidade: Conjunto de características e os relacionamentos entre elas, que fornecem a base para a especificação dos requisitos de qualidade e para a avaliação da qualidade.

Monitoração: aquisição de dados para verificar as condições de funcionamento do processo (sinalização de valores limites, ocorrências de falhas, alarmes e relatórios).

Nível de pontuação: Ponto de escala em uma escala ordinal, que é usado para categorizar uma escala de medição.

Qualidade: Totalidade de características de uma entidade que lhe confere a capacidade de satisfazer as necessidades explícitas e implícitas.

Satisfação: Ausência do desconforto e presença de atitudes positivas para com o uso de um produto.

Scripts: são sub-rotinas em que se podem definir linhas de códigos em linguagem de programação (Basic, VB ou C), que permitem maior flexibilidade para associar ações a eventos específicos.

Sinóptico: Relativo à sinopse; que permite ver de um só lance de vista as diversas partes de um conjunto; resumido, sintético: quadro sinóptico.

Sistema: Conjunto integrado que consiste em um ou mais processos, *hardware*, *software*, recursos e pessoas, capaz de satisfazer uma necessidade ou objetivo definido.

***Software*:** Conjunto completo ou apenas uma parte, dos programas, procedimentos, regras e documentação associada de um sistema de processamento de informação.

***Software* de supervisão / Sistema supervisório / Aplicativo de supervisão:** algoritmos de alto nível, conjunto de telas, recursos, comunicações e configurações elaboradas pelo usuário responsável pela personalização do sistema SCADA. É no aplicativo que é definida toda a funcionalidade do sistema. Software destinado a realizar a supervisão do processo.

Supervisão: aquisição de dados para permitir a elaboração de uma estratégia de operação para maximizar o retorno financeiro (maior produção, qualidade e eficiência).

Tags: é o nome dado às variáveis utilizadas em um sistema SCADA.

Usuário: Indivíduo que usa o produto de *software* para executar uma função específica.

Validação (para "*software*"): Processo de avaliação de "*software*" a fim de assegurar que este atende aos requisitos especificados.

APÊNDICES

APÊNDICE A – Questionário de delimitação do perfil do especialista

Solicito sua colaboração para preenchimento do questionário a seguir marcando com um X conforme solicitado:

1- Marque sua experiência como programador de supervisório. (Marque uma opção)

( ) Menos de 1 ano ( ) De 1 a 2 anos ( ) De 2 a 3 anos

( ) De 3 a 4 anos ( ) Mais de 4 anos

2- Em quantos supervisórios você já teve participação direta ou indireta? (Marque uma opção)

( ) Nenhum ( ) 1 supervisório ( ) 2 supervisórios

( ) 3 ou 4 supervisórios ( ) Mais de 4 supervisórios

3- Em quantos projetos de automação você participou? (Marque uma opção)

( ) Nenhum ( ) 1 ou 2 projetos ( ) 3 ou 4 projetos

( ) 5 ou 6 projetos ( ) Mais de 6 projetos

4- Você já participou de quantos treinamentos de sistemas supervisórios? (Marque uma opção e considere disciplinas de supervisório cursadas como 1 treinamento)

( ) Nenhum ( ) 1 treinamento ( ) 2 treinamentos

( ) 3 treinamentos ( ) Mais de 3 treinamentos

5- Com quantos softwares diferentes de desenvolvimento de supervisórios você já trabalhou? (Marque uma opção)

( ) Nenhum ( ) 1 software ( ) 2 softwares

( ) 3 softwares ( ) Mais de 3 softwares

6- Você já ministrou algum curso ou treinamento de supervisório? (Marque uma opção e considere cada disciplina ministrada em cada semestre como 1)

( ) Nenhum ( ) 1 apenas ( ) 2 ou 3 apenas

( ) 4 apenas ( ) Mais de 4

7- Você já fez manutenção (correção de erros ou expansão) de quantos sistemas supervisórios feitos por terceiros? (Marque uma opção)

( ) Nenhum ( ) 1 supervisório ( ) 2 supervisórios

( ) 3 ou 4 supervisórios ( ) Mais de 4 supervisórios

8- Marque as atividades (cargos) que já exerceu ou exerce, na área de automação. **Pode marcar mais de uma opção**.

( ) Gerente de projeto ( ) Professor na área ( ) Analista de automação

( ) Programador ( ) Usuário

9- Marque a opção que melhor classifica seu grau de instrução. **Pode marcar mais de uma opção**.

( ) Técnico ( ) Graduado ( ) Especialista

( ) Mestre/Mestrando ( ) Doutor/Doutorando

APÊNDICE B – Questionário do grau de relevância de características de qualidade de um sistema supervisório

Solicito ao especialista o preenchimento do questionário a seguir, marcando com um X conforme solicitado:

Característica: Funcionalidade

Dinâmica do processo: Subcaracterística que verifica quanto a dinâmica do processo implementada no supervisório corresponde à especificação do projeto.

1- Quão relevante é a implementação, de acordo com a especificação do projeto, da dinâmica do processo para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Segurança de acesso: Subcaracterística que se refere à criação de usuários com diferentes tipos de acesso aos recursos.

2- A criação de usuários com diferentes tipos de acesso aos recursos e a possibilidade de cadastrá-los on-line é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Armazenamento de dados: Subcaracterística que verifica a existência da possibilidade de armazenamento de dados em históricos, banco de dados e a geração de relatórios em intervalos de tempo definidos.

3- Qual é o grau de relevância da possibilidade de armazenamento de dados em históricos, banco de dados e a geração de relatórios em intervalos de tempo definidos para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Configuração dos parâmetros : Subcaracterística que avalia se o supervisório possibilita a configuração dos parâmetros essenciais para o processo durante a execução do aplicativo.

4- A possibilidade de alteração do valor dos parâmetros essenciais para o processo é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Característica: Usabilidade

Apreensibilidade: Subcaracterística que verifica o nível de facilidade com que o usuário aprende a operar o supervisório com a sua utilização.

5- A facilidade com que o operador aprende a utilizar a aplicação é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Apresentabilidade: Subcaracterística que avalia a estética e organização das telas e animações criadas.

6- Indique o grau de relevância da estética das telas e animações criadas para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

7- A avaliação da quantidade de informações na tela (excesso ou ausência) é relevante para o desenvolvimento do supervisório?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

8- Avaliar a organização ou hierarquia das telas é relevante para o desenvolvimento do supervisório?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

9- É relevante avaliar o quanto os desenhos dos elementos do sistema estão representando bem o processo real para o desenvolvimento do supervisório?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Status das variáveis: Subcaracterística que avalia o quanto o status das variáveis envolvidas está presente no supervisório.

10- A presença do status das variáveis envolvidas é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Recursos gráficos: Subcaracterística que verifica o quanto a criação de gráficos está de conformidade com o projeto.

11- Verificar o quanto os gráficos atendem ao projeto é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Documentação: Subcaracterística que verifica qual é o grau em que a documentação do supervisório facilita a utilização do mesmo.

12- Verificar se a documentação do supervisório facilita a utilização do mesmo é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Característica: Confiabilidade

Alarmes: Subcaracterística que avalia o quanto a geração de prioridades, visualização e quantidade de alarmes atende as especificações do projeto.

13- Quão relevante é verificar se a geração de prioridades e quantidade de alarmes atende às especificações do projeto para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

14- A forma de visualização e intervenção dos alarmes é relevante para o desenvolvimento do supervisório.

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Comunicação com o CLP: Subcaracterística que avalia se a comunicação com o CLP está configurada corretamente e pode ser reestabelecida on-line caso ocorra falhas.

15- Avaliar se a comunicação com o CLP é confiável e pode ser reestabelecida on-line caso ocorra falhas é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

16- Realizar as configurações de tags e drivers de comunicação corretamente é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Característica: Configuração

Quantidade de variáveis: Subcaracterística que verifica o quanto o número de variáveis atende às especificações do projeto.

17- Verificar o quanto o número de variáveis atende as especificações do projeto é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Organização de tags: Subcaracterística que avalia a organização dos tags do supervisório.

18- A organização dos tags em relação ao nome e ao agrupamento é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Quantidade de Scripts: Subcaracterística que avalia o quanto a qualidade e quantidade de scripts do supervisório atende a necessidade do projeto.

19- Avaliar se a qualidade e quantidade de scripts do supervisório atende a necessidade do projeto é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Erros de Scripts: Subcaracterística que avalia o grau de acerto das configurações de scripts do supervisório.

20- Configurar os scripts do supervisório corretamente é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Configuração de Telas: Subcaracterística que avalia o quanto a quantidade de telas e interação entre elas está de conformidade com a necessidade do projeto.

21- Avaliar se a quantidade de telas e interação entre elas está de conformidade com a necessidade do projeto é relevante para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

Tempo de desenvolvimento: Subcaracterística que avalia quanto o tempo de desenvolvimento está de conformidade com a necessidade do projeto.

22- É relevante avaliar se o tempo de desenvolvimento está de conformidade com a necessidade do projeto para o desenvolvimento do supervisório com qualidade?

( ) Sem relevância ( ) Pouco relevante ( ) Moderadamente Relevante
( ) Relevante ( ) Muito Relevante

APÊNDICE C – Questionário do grau de presença de características de qualidade de um sistema supervisório

Solicito ao especialista o preenchimento do questionário a seguir marcando com um X conforme solicitado:

Característica: Funcionalidade

Dinâmica do processo: Subcaracterística que verifica quanto a dinâmica do processo implementada no supervisório corresponde à especificação do projeto.

1- A implementação da dinâmica do processo está de acordo com a especificação do projeto?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

Segurança de acesso: Subcaracterística que se refere à criação de usuários com diferentes tipos de acesso aos recursos.

2- Qual é o grau de presença de usuários com diferentes tipos de acesso aos recursos, considerando a possibilidade de cadastrá-los durante a execução do aplicativo?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

Armazenamento de dados: Subcaracterística que verifica a existência da possibilidade de armazenamento de dados em históricos, banco de dados e a geração de relatórios em intervalos de tempo definidos.

3- Quão presente no supervisório é a possibilidade de armazenamento de dados em históricos, banco de dados e a geração de relatórios em intervalos de tempo definidos?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

Configuração dos parâmetros : Subcaracterística que avalia se o supervisório possibilita a configuração dos parâmetros essenciais para o processo durante a execução do aplicativo.

4- A possibilidade de alteração do valor dos parâmetros essenciais para o processo está presente no supervisório?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

Característica: Usabilidade

Apreensibilidade: Subcaracterística que verifica o nível de facilidade com que o usuário aprende a operar o supervisório com a sua utilização.

5- Qual é o nível de facilidade de operação do supervisório através da utilização do mesmo?

( ) Não facilita ( ) Facilita pouco ( ) Facilita moderadamente

( ) Facilita muito ( ) Facilita totalmente

Apresentabilidade: Subcaracterística que avalia a estética e organização das telas e animações criadas.

6- A estética das telas e animações criadas no supervisório atende à necessidade do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

7- A quantidade de informações nas telas atende à idéia de mostrar ao operador os dados essenciais e evitar excessos?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

8- Qual é o grau de presença da organização/hierarquia das telas no supervisório desenvolvido?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

9- Os desenhos dos elementos do sistema, presentes no supervisório, estão atendendo a representação do processo real?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

Status das variáveis: Subcaracterística que avalia o quanto o status das variáveis envolvidas está presente no supervisório.

10- Qual é o grau de presença do status das variáveis envolvidas no supervisório?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

Recursos gráficos: Subcaracterística que verifica o quanto a criação de gráficos está de conformidade com o projeto.

11- Os gráficos criados no supervisório e as suas características atendem à necessidade do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

Documentação: Subcaracterística que verifica qual o grau em que a documentação do supervisório facilita a utilização do mesmo.

12- Qual é o grau de presença de características da documentação do supervisório que facilita a utilização do mesmo?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

Característica: Confiabilidade

Alarmes: Subcaracterística que avalia o quanto a geração de prioridades, visualização e quantidade de alarmes atende às especificações do projeto.

13- Qual é o grau em que a quantidade e a geração de prioridades dos alarmes atende às especificações do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

14- A forma de visualização e intervenção dos alarmes facilita o reconhecimento por parte dos operadores?

( ) Não facilita ( ) Facilita pouco ( ) Facilita moderadamente

( ) Facilita muito ( ) Facilita totalmente

Comunicação com o CLP: Subcaracterística que avalia se a comunicação com o CLP está configurada corretamente e pode ser reestabelecida on-line caso ocorra falhas.

15- Qual é o grau de presença da confiabilidade da comunicação com o CLP e do restabelecimento on-line da mesma, caso ocorra falhas?

( ) Total ausência ( ) Baixa presença ( ) Moderada presença

( ) Alta presença ( ) Total presença

16- Qual é o grau de acerto das configurações de tags e drivers de comunicação?

Cálculo: GA = 1 – (A/B), onde A = nº de erros de configuração (tags ou drivers) e B = nº de tags + nº de drivers

( ) Menos de 0,75 ( ) Entre 0,75 e 0,80 ( ) Entre 0,81 e 0,85

( ) Entre 0,86 e 0,94 ( ) Mais de 0,94

Característica: Configuração

Quantidade de variáveis: Subcaracterística que verifica o quanto o número de variáveis atende às especificações do projeto.

17- O número de variáveis atende às especificações do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

Organização de tags: Subcaracterística que avalia a organização dos tags do supervisório.

18- A organização dos tags em relação ao nome e ao agrupamento facilita a compreensão da funcionalidade de cada tag?

( ) Não facilita ( ) Facilita pouco ( ) Facilita moderadamente

( ) Facilita muito ( ) Facilita totalmente

Quantidade de Scripts: Subcaracterística que avalia o quanto a quantidade de scripts do supervisório atende a necessidade do projeto.

19- Qual é o grau em que a quantidade de scripts do supervisório atende à necessidade do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

Erros de Scripts: Subcaracterística que avalia o grau de acerto das configurações de scripts do supervisório.

20- Qual é o grau de acerto das configurações de scripts do supervisório?

Cálculo: GA = 1 – (A/B), onde A = nº de operações incorretas dos scripts e B = nº total de operações dos scripts

( ) Menos de 0,75 ( ) Entre 0,75 e 0,80 ( ) Entre 0,81 e 0,85

( ) Entre 0,86 e 0,94 ( ) mais de 0,94

Configuração de Telas: Subcaracterística que avalia o quanto a quantidade de telas e interação entre elas está de conformidade com a necessidade do projeto.

21- A quantidade de telas e interação entre elas está de conformidade com a necessidade do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

Tempo de desenvolvimento: Subcaracterística que avalia quanto o tempo de desenvolvimento está de conformidade com a necessidade do projeto.

22- O tempo de desenvolvimento está de conformidade com a necessidade do projeto?

( ) Não atende ( ) Atende pouco ( ) Atende moderadamente

( ) Atende muito ( ) Atende totalmente

APÊNDICE D – Questionário de avaliação do cenário atual de desenvolvimento de supervisórios

Solicito ao especialista o preenchimento do questionário a seguir:

1 - Quais os problemas/dificuldades você enfrentou para aprender a desenvolver um supervisório?

Resposta 1: O maior problema a princípio foi o entendimento do processo, a segunda dificuldade é a organização da planilha de pontos interligando o processo físico com o CLP e o SCADA (driver de comunicação).

Resposta 2: O manuseio de muitos valores de endereços de variáveis, que foi difícil de organizar tudo sem se perder nos valores.

Resposta 3: A grande dificuldade foi com relação a falta de ligação direta entre a literatura formal e o software utilizado, ou seja, a literatura formal normalmente trata das tecnologias envolvidas, protocolos, características dos padrões, mas não aborda a fundo alguma implementação. Tive que focar apenas no "Help" do software utilizado.

2 - Como você aprendeu a desenvolver supervisórios?

Resposta 1: Visualizando projetos de outros profissionais.

Resposta 2: Em parte no curso Superior de mecatrônica do IFCE, em parte desenvolvendo na empresa em que trabalho.

Resposta 3: Com o tutorial de uma das plataformas de desenvolvimento.

3 - O projeto de desenvolvimento de supervisório possui alguma lista de características essenciais a seu ver?

Resposta 1: Sim, como todo projeto relacionado com software tem que ser organizado, bem estruturado, comentado, testado, layout de tela agradável etc.

Resposta 2: Sim. Basicamente histórico e alarmes são fundamentais.

Resposta 3: Para o desenvolvimento de supervisórios posso citar uma lógica inicial: 1°Especificação das tecnologias envolvidas. 2°Definição do padrão de comunicação a ser utilizado. 3° Montagem da arquitetura de rede, etc.

4 - Você se baseia em alguma metodologia de desenvolvimento de software ou norma para elaborar supervisórios?

Resposta 1: Até agora não segui nenhuma norma padronizada.

Resposta 2: Não, apenas no meu conhecimento tácito.

Resposta 3: Atualmente me baseio em engenharia de requisitos.

5 - No desenvolvimento de supervisório você sente ausência de alguma metodologia (passo a passo)? Por quê?

Resposta 1: Sim, da mesma forma que na vida sem um metodologia têm-se uma maior probabilidade de apresentar erros.

Resposta 2: Uma metodologia seria bem vinda de forma que o projeto possa ser entregue no prazo.

Resposta 3: Na Engenharia de automação como um todo é carente de metodologias. Então sim, sinto ausência.

6 - Você utiliza algum método de avaliação de supervisórios?

Resposta 1: Não.

Resposta 2: Não.

Resposta 3: Apenas aprovação de telas com os clientes.

7 - Quantos contatos com o cliente são feitos ao longo do projeto?

Resposta 1: Antes do início, na finalização e aprovação da tela e no startup.

Resposta 2: Se bem planejado com o cliente, não mais que três, muito embora, como em qualquer desenvolvimento de software, sempre aparece alguma alteração de última hora.

Resposta 3: Normalmente 2, um no inicio do projeto e outro no final. Apenas caso surjam dúvidas com relação ao processo, entra-se em contato com o mesmo.

8 - A elaboração do supervisório na sua empresa é baseada somente na experiência de cada programador ou existem padrões a serem seguidos?

Resposta 1: Conforme a experiência de cada programador e conforme o que um programador conseguiu aprender de outro mais experiente, ou aprender com os supervisórios antigos tomados como exemplo para didática.

Resposta 2: Sim existem padrões.

Resposta 3: Apenas na experiência do desenvolvedor.

9 - Existe alguma ferramenta de gestão de projetos utilizada pelo seu grupo? Se existe qual a importância desta ferramenta?

Resposta 1: MsProject, cronograma e acompanhamento das atividades.

Resposta 2: Não, muito embora uma ferramenta de gestão maximize a produtividade do responsável pelo desenvolvimento do projeto.

Resposta 3: Sim. Atualmente o gerenciamento de projetos na empresa é baseado integralmente na ferramenta.

ANEXOS

ANEXO A – Planejamento da avaliação de um sistema supervisório através de diagramas UML

A ferramenta Qteste desenvolvida na monografia de Lana Beatriz, no departamento de engenharia de teleinformática da Universidade Federal do Ceará, para avaliar softwares é divida em quatro etapas listadas abaixo, que se relacionam conforme o diagrama de atividades da figura A.1

1. Cadastramento de perfis
2. Cadastramento de tipo de *software*
3. Criação da *Base de Avaliação*
4. Avaliação

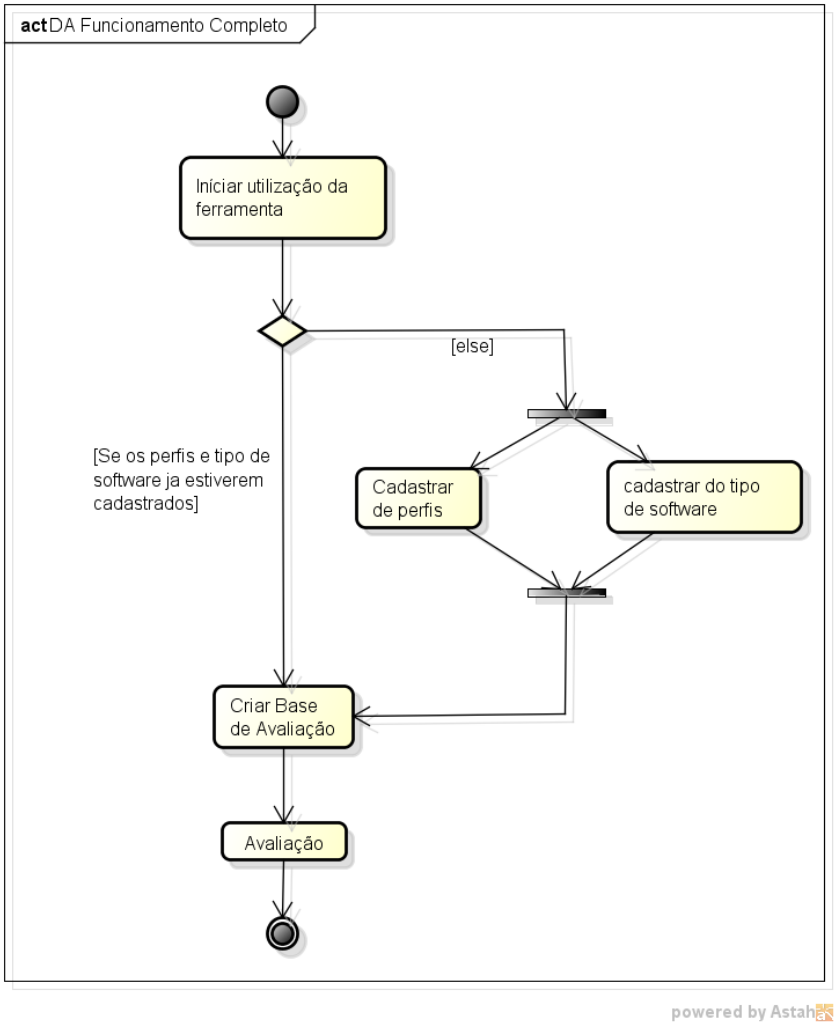


Figura A.1 - Diagrama de Atividades: Funcionamento Completo

A sequência temporal do diagrama mostra a dependência dos eventos para que ocorra a avaliação do *software*, que só acontece quando todas as outras atividades já tiverem sido realizadas.

Desde a inicialização, a realização do cadastro de perfis, o cadastro do tipo de *software* e a criação da *Base de Avaliação* podem ser realizados independentes da atividade *Avaliação*, isto é, é possível reutilizar perfis, tipos de *software* e *Bases de Avaliação* já cadastrados. A ferramenta Qteste foi projetada para criar uma base de dados reutilizável.

A exemplo disso, um perfil de especialista de *softwares* supervisórios que foi cadastrado pode ser utilizado como especialista em várias *Base de Avaliação*, que, por sua vez, pode ser usada na *Avaliação* de vários *softwares* supervisórios.

A.1 Cadastro de perfis

A figura A.2 mostra o diagrama de casos de uso para a atividade Cadastro de Perfis. A seta vazada que liga os casos de uso representa a generalização. O fluxo da seta que parte do administrador, ator do caso de uso, indica que o administrador realiza/participa do caso de uso Cadastrar Perfil.

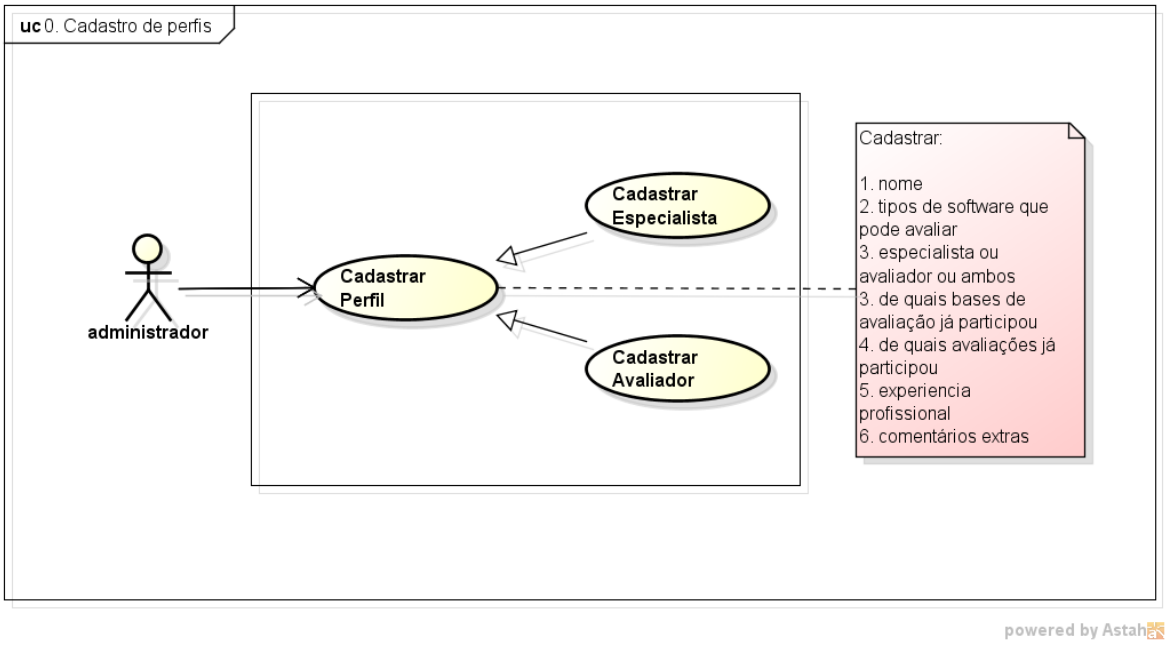


Figura A.2 - Diagrama de Casos de Uso: *Cadastro de Perfis*

As notas em diagramas, de cor rosa, contém observações. Nesse diagrama, a nota indica quais dados serão guardados com o cadastramento do perfil.

Em tópicos anteriores, aos poucos, os perfis *avaliador* e *especialista* foram introduzidos. Porém, agora se faz necessário uma definição mais exata:

- **Especialista**: é a pessoa que responderá o questionário da *Base de Avaliação*.
- **Avaliador**: é a pessoa que responderá o questionário da *Avaliação*.

Essa diferenciação é importante na escolha dos perfis, pois um perfil especialista desejado em um processo de avaliação não necessariamente será também um perfil avaliador desejado e vice-versa. A *Base de Avaliação* requer uma análise geral dos atributos com relação a um tipo de *software*, enquanto a *Avaliação* requer uma análise específica da presença dos atributos em um *software*. Porém isso não impede de ter um cadastro especialista e um avaliador para a mesma pessoa.

A.2 Cadastro do tipo de software

A figura A.3 mostra o diagrama de casos de uso para o Cadastro do Tipo de *Software*. O estereótipo <<*include*>> indica que o caso de uso Cadastrar Tipo de *Software* inclui todos os outros casos de uso.

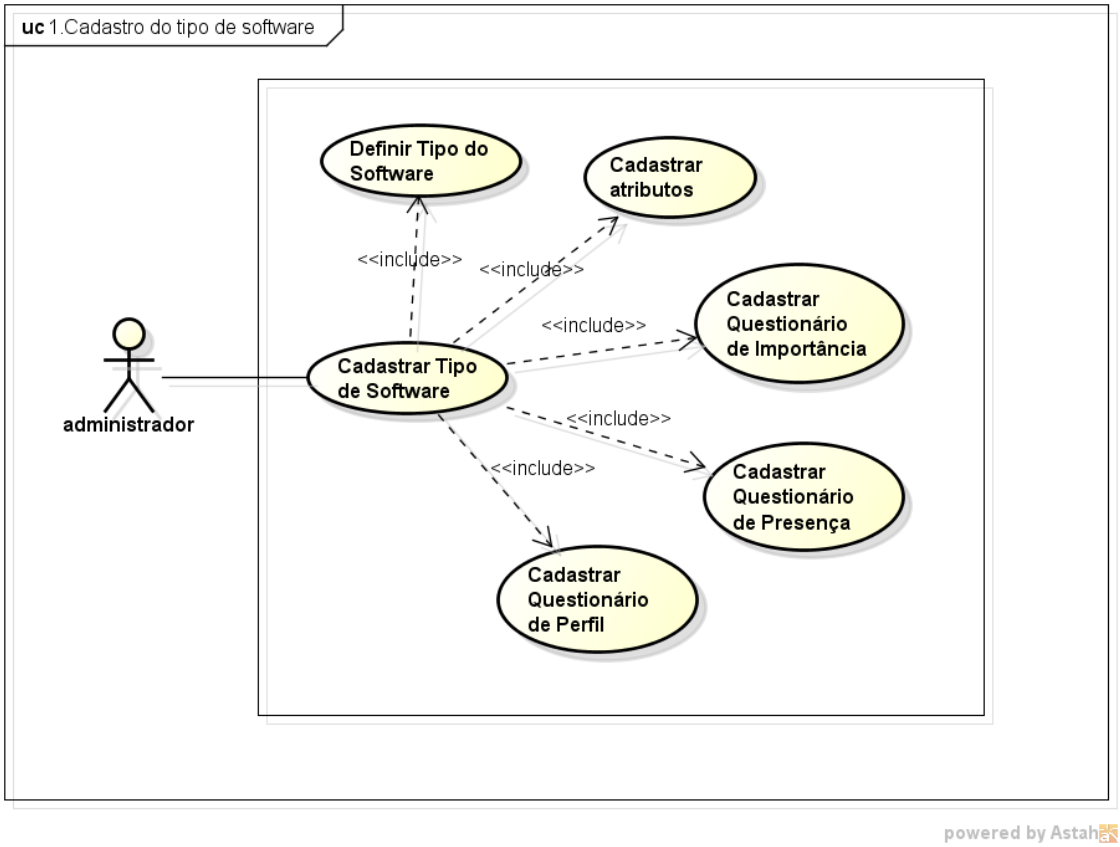


Figura A.3 - Diagrama de Casos de Uso - *Cadastro do Tipo de Software*

É importante perceber que definir o tipo de *software* significa definir os questionários de perfil, presença e importância. Esta etapa é realizada exclusivamente pelo

administrador, ou pelo grupo que o representa, que, de acordo com a metodologia de avaliação, determina os atributos e as questões para análise da qualidade.

O resultado dessa etapa é um conjunto de dados que reúne informações básicas para criação de uma *Base de Avaliação* e para a criação de uma *Avaliação*, podendo ser reutilizadas.

A.3 Base de Avaliação

Na *Base de Avaliação* cujo diagrama de casos de uso é mostrado na figura A.4, além do administrador, o especialista participa como ator e realiza os questionários de Importância e de Perfil.

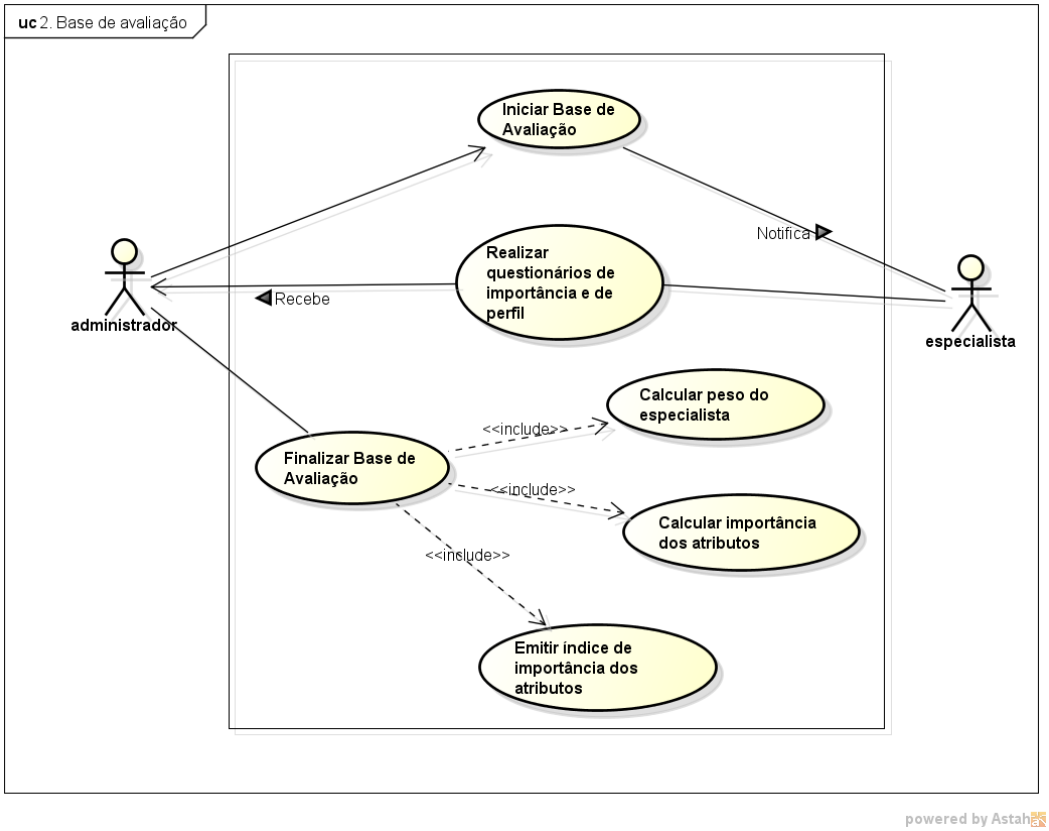


Figura A.4 - Diagrama de Casos de Uso: *Base de Avaliação*

Dois casos de uso apresentados no diagrama foram detalhados em diagramas de atividades, os quais estão organizados conforme uma sequência lógica adaptada para programação orientada a objetos, assim como todos os outros diagramas presentes.

O primeiro diagrama da figura A.5 representa o caso de uso *Iniciar Base de Avaliação*. O segundo diagrama da figura A.5 representa o caso de uso *Finalizar Base de*

Avaliação, onde é criado o objeto que representa o conjunto de dados da *Base de Avaliação*. Os retângulos sem os cantos arredondados nos diagramas de atividades representam objetos.

O estereótipo <<*caso de uso*>> nas atividades indica que estas atividades são casos de uso completos, isto é, possuem um nível de complexidade que pode ser detalhado em outros diagramas.

Essa etapa tem como resultado principal os seguintes dados:

- **Peso dos especialistas**
- **Importância dos atributos**

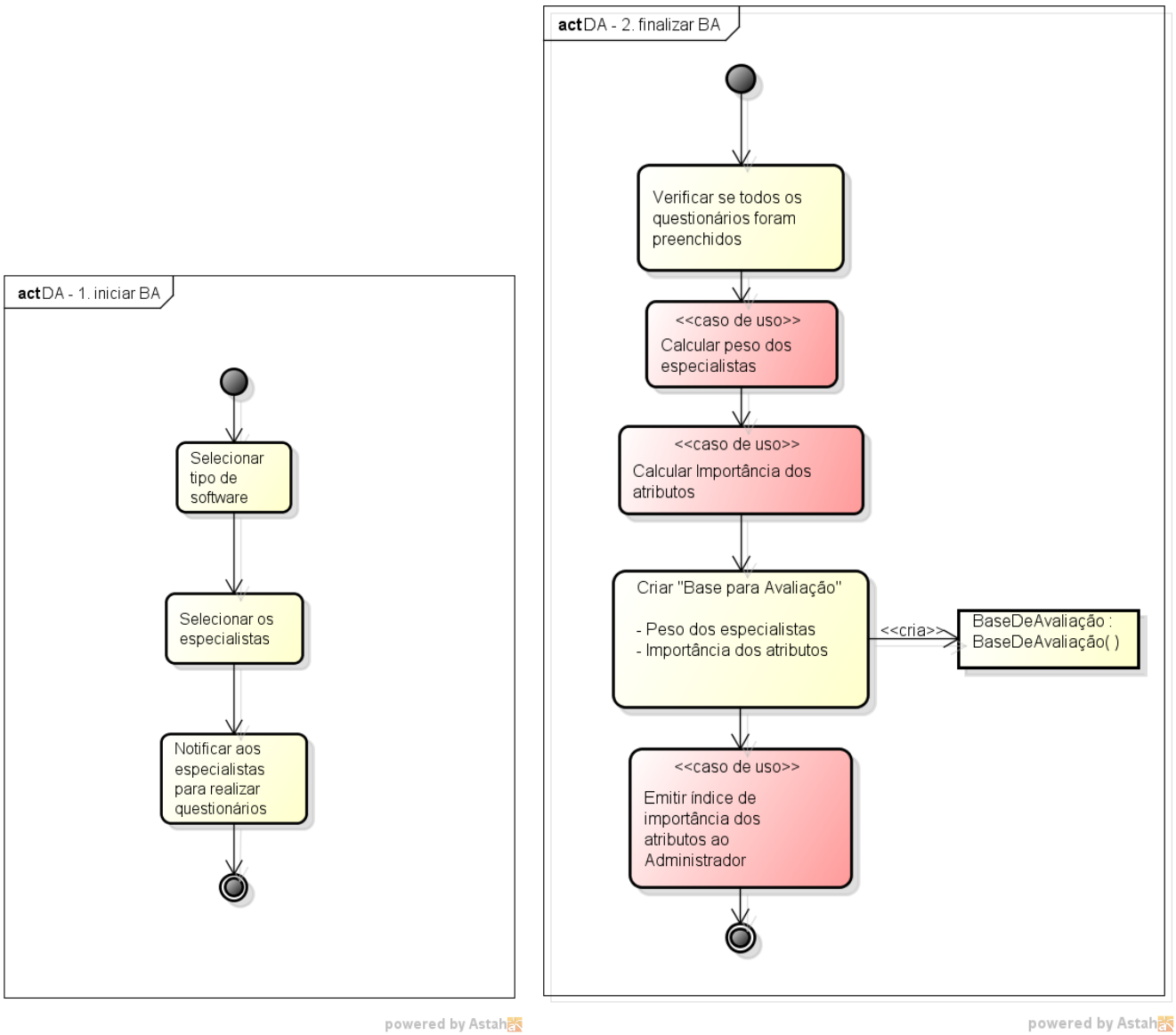


Figura A.5 - Diagramas de Atividades: *Base de Avaliação*

Ambos os resultados já podem fornecer informações importantes sobre os atributos de qualidade de um tipo de *software* ou sobre o nível de experiência dos

profissionais envolvidos. A análise desses dados pode fornecer uma noção sobre o ambiente de avaliação e influenciar no processo. Por exemplo, a constatação de um nível muito baixo da experiência dos especialistas pode frustrar a criação da *Base de Avaliação.*

A situação ideal é a criação de várias *Bases de Avaliação* em diferentes ambientes de desenvolvimento, que possam ser compartilhadas de alguma forma, a exemplo do que teríamos com um repositório internacional com padrões de avaliação.

A.4 Avaliação

A etapa de *Avaliação*, cujo diagrama de casos de uso é mostrado na figura A.6, possui a participação dos atores Administrador e Avaliador. Possui ainda semelhante inicialização com a etapa anterior, *Base de Avaliação*, mas sua finalização inclui outros cálculos e o resultado final de todo o processo de avaliação.

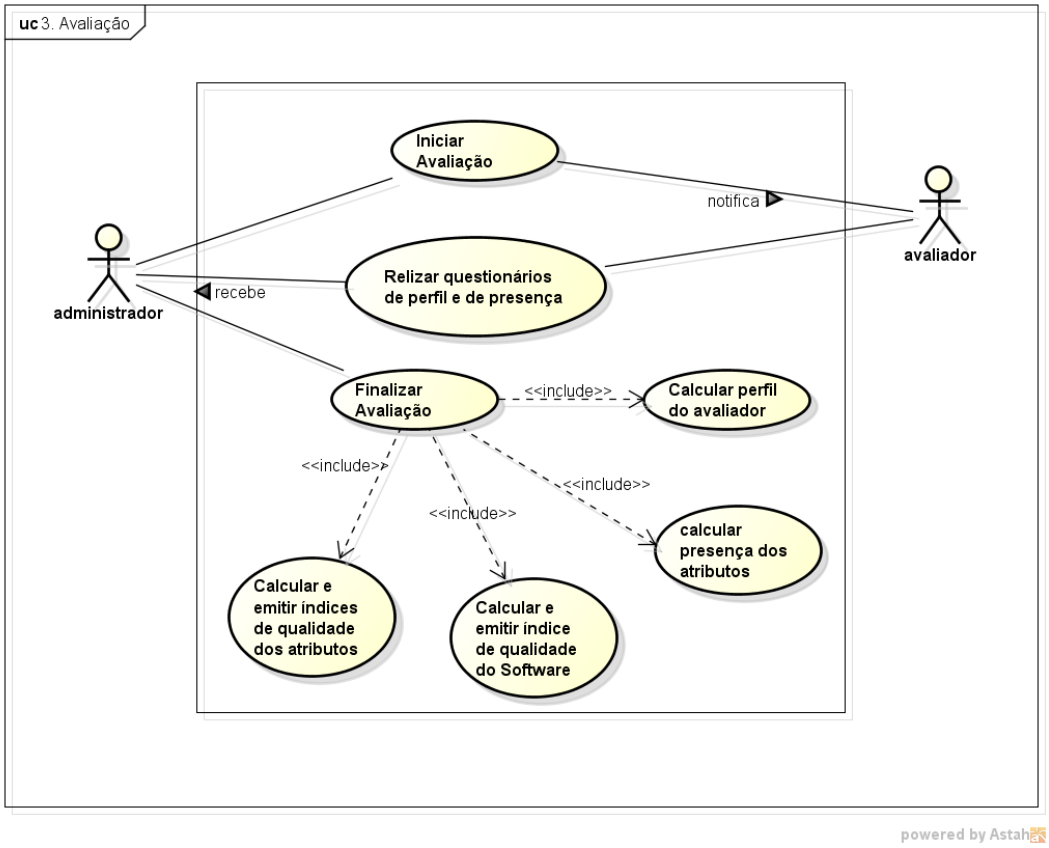


Figura A.6 - Diagrama de Casos de Uso: *Avaliação*

Para os casos de uso *Iniciar Avaliação* e *Finalizar Avaliação* foram criados os diagramas de atividades da figura A.7 e da figura A.8, respectivamente. Esta etapa necessita de alguns dados que foram coletados nas etapas anteriores. O primeiro diagrama mostra isso

com os pequenos quadrados antes ou depois das atividades, indicando que será utilizado ou gerado um objeto na atividade.

Somente nesta etapa o *software* a ser avaliado é selecionado, para que se proceda ao teste de avaliação para identificar seu grau de qualidade. Apesar desta definição tardia, um processo de avaliação de *software* pode já iniciar com o objetivo de avaliar um *software* específico.

Quando o objetivo do processo de avaliação é criar um padrão de avaliação, isto é, uma *Base de Avaliação*, para diversos *softwares* do mesmo tipo, os atributos tendem a ser mais gerais. Se o objetivo é avaliar um *software* específico os atributos tendem a detalhar mais critérios específicos deste *software*. Por isso, a metodologia de avaliação propõe que os objetivos sejam estabelecidos na primeira etapa do método de avaliação, ou seja, antes mesmo de começar a utilizar a ferramenta.

Ainda neste diagrama, as atividades de cálculo representam casos de uso complexos. O segundo diagrama finaliza todo o processo de avaliação de um determinado *software* na ferramenta Qteste, com os seguintes resultados:

- **Índices de qualidade dos atributos**
- **Índices de qualidade das características**
- **Índice de qualidade do *software* avaliado**

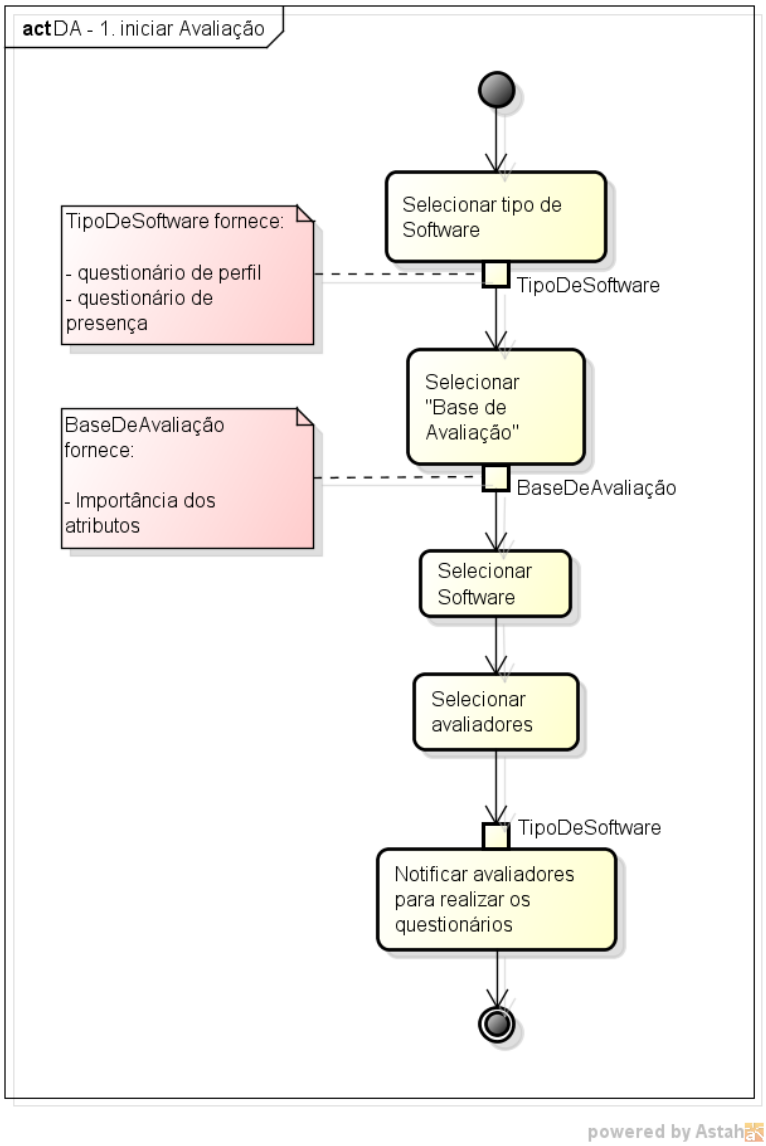


Figura A.7 - Diagrama de Atividades 1: *Avaliação*

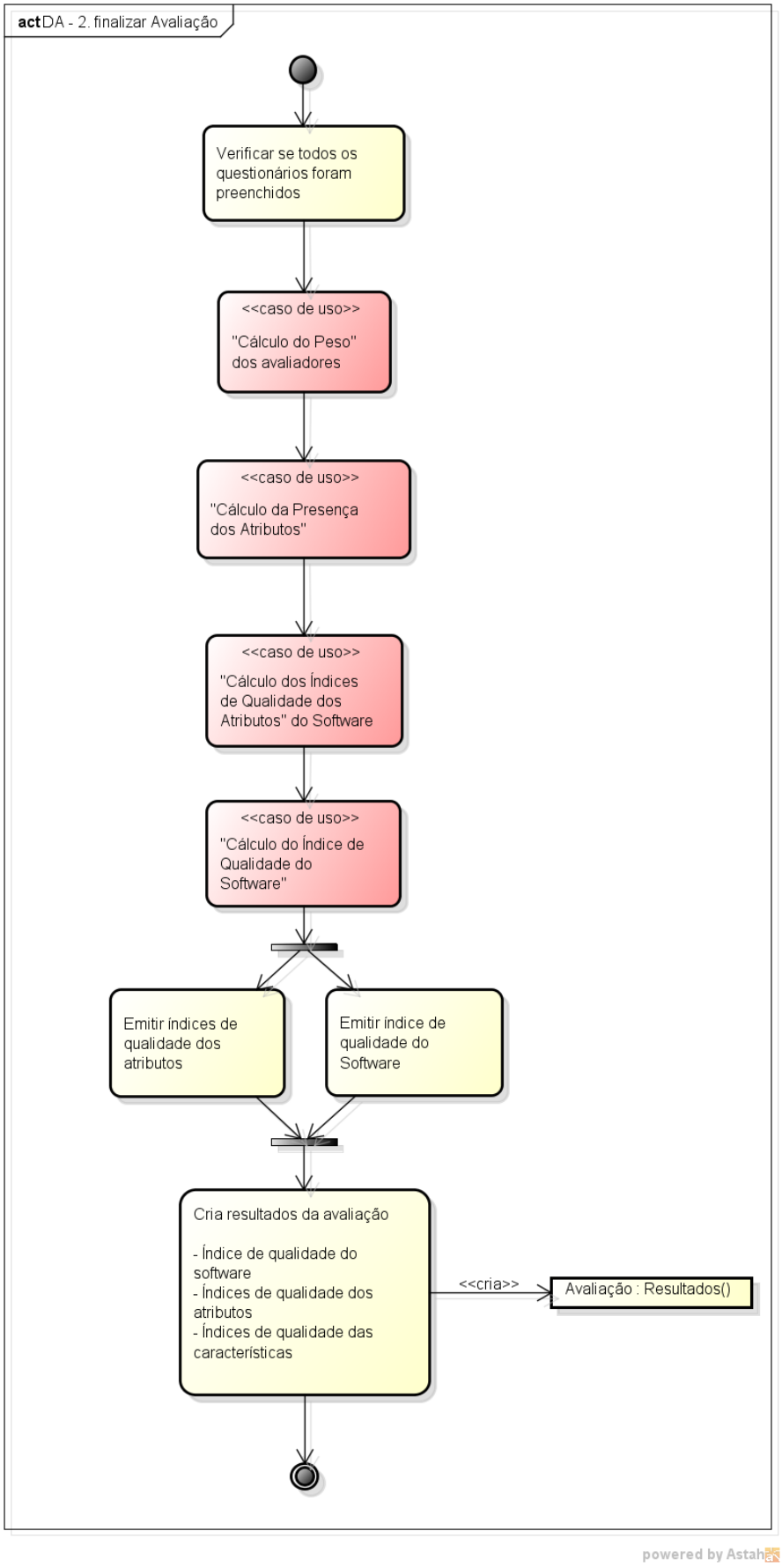


Figura A.8 - Diagrama de Atividades 2: *Avaliação*

A.5 Cálculos

Os casos de uso que envolviam cálculos foram detalhados em diagramas de atividades, passo a passo, estabelecendo uma ordem temporal para a construção da lógica da ferramenta. Nestes diagramas, foi definida uma nomenclatura para cada variável com a finalidade de facilitar o entendimento e a implementação da ferramenta.

É importante destacar que quando há uma atividade que envolve uma questão de algum dos questionários, significa que o mesmo processo deve ser realizado para todas as questões. Por exemplo, no cálculo do peso, a definição de um resultado numérico para uma questão pressupõe que o processo deva ser realizado para todas as questões.

A leitura dos diagramas é facilitada com um esquema de cores, as atividades verdes indicam que há entrada de dados e as azuis indicam que há criação de algum dado de saída. Os estereótipos também lidam com a entrada e saída de dados, a fim de tornar a leitura dos diagramas ainda mais fácil e intuitiva.

O diagrama da figura A.9, que representa o caso de uso Cálculo do Peso, possui um comportamento condicional em duas atividade para definição numérica da resposta do questionário de perfil: se é uma questão de única escolha, o resultado será de 0 a 4, de acordo com a resposta; se for múltipla escolha, a resposta é a soma unitária de cada item marcado. Os resultados dessa decisão serão somados na atividade seguinte.

No diagrama do cálculo do peso, assim como em outros cálculos, o *Tipo de Software* é considerado uma entrada com dados que identificam a quantidade de questões, os atributos que serão avaliados e as características a que estes atributos pertencem.

Os diagramas da figura A.10 e da figura A.11 representam respectivamente os cálculos da Importância e da Presença dos Atributos, os quais possuem um processamento inicial semelhante. O primeiro, porém, obtém na saída os resultados em porcentagem, adicionando mais algumas atividades. Esses diagramas incluem a *fuzzificação*, a obtenção dos números triangulares *fuzzy* no formado [a m b], e a *defuzzyficação*, a obtenção dos valores *Crisp*.

No diagrama do Cálculo dos Índices de Qualidade dos Atributos, figura A.12, o número *fuzzy* da importância, que determina um padrão de qualidade, é redefinido como um número *fuzzy* trapezoidal. Isso é possível porque se pode considerar que qualquer valor acima do padrão de qualidade (à direita da função característica) está, também, acima do grau de qualidade desejada e, portanto, é plenamente aceitável. Este diagrama utiliza a área de

intersecção entre números *fuzzy* de presença e importância, o que na prática representa o quanto os atributos presentes estão de acordo com o padrão de qualidade do *software*.

A.6 Cálculo do Peso

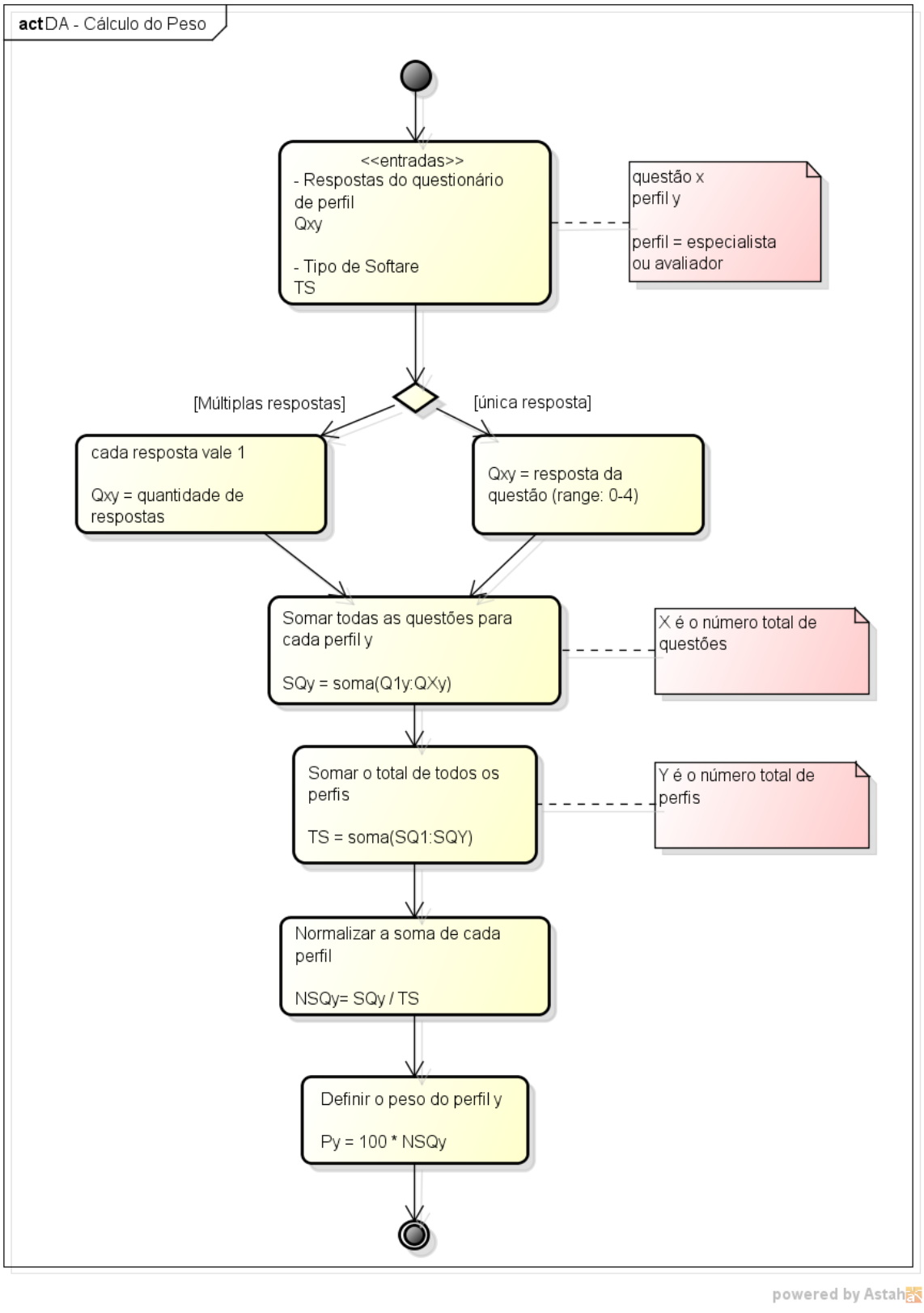


Figura A.9 - Cálculo do Peso

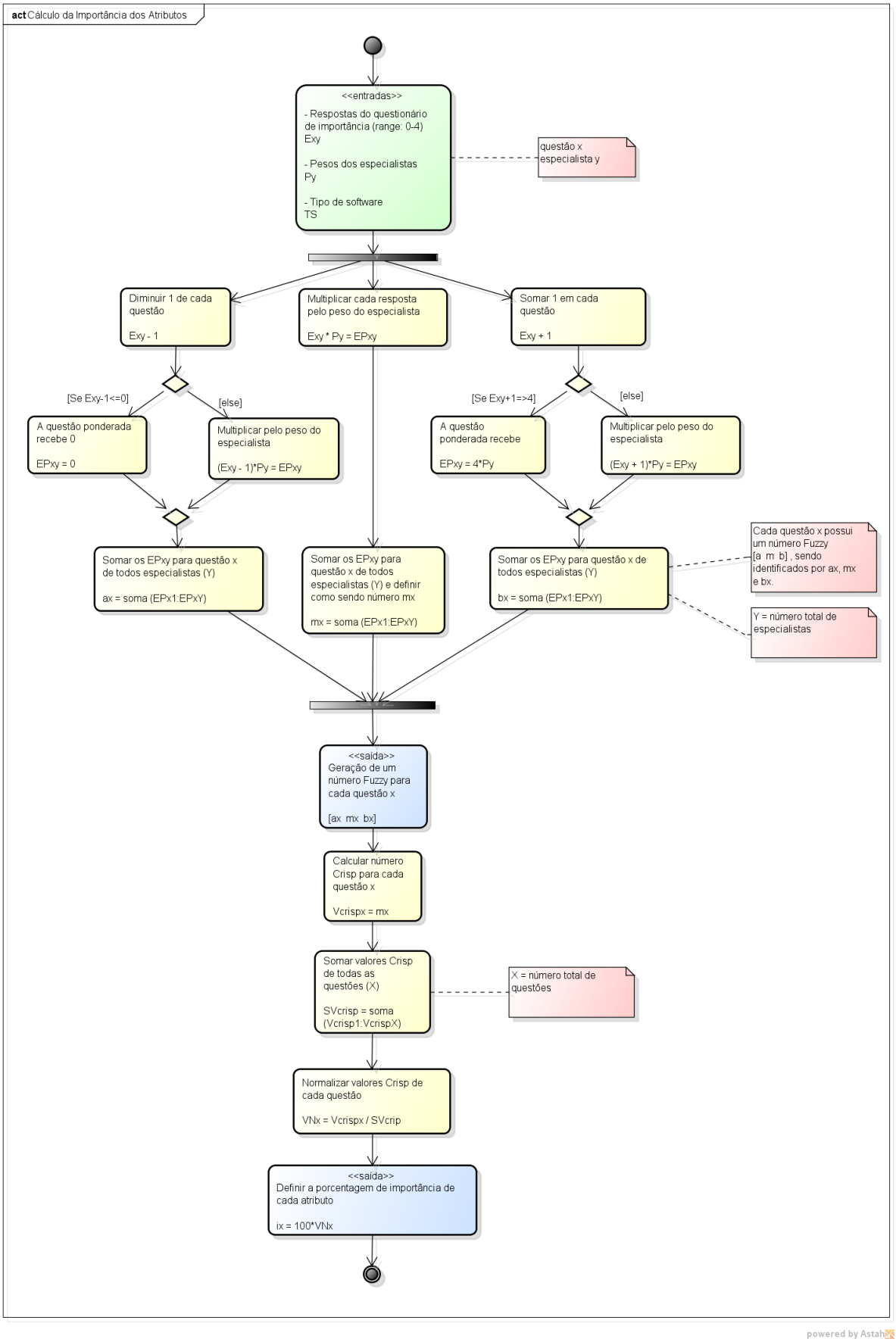


Figura A.10 - Cálculo da importância dos atributos

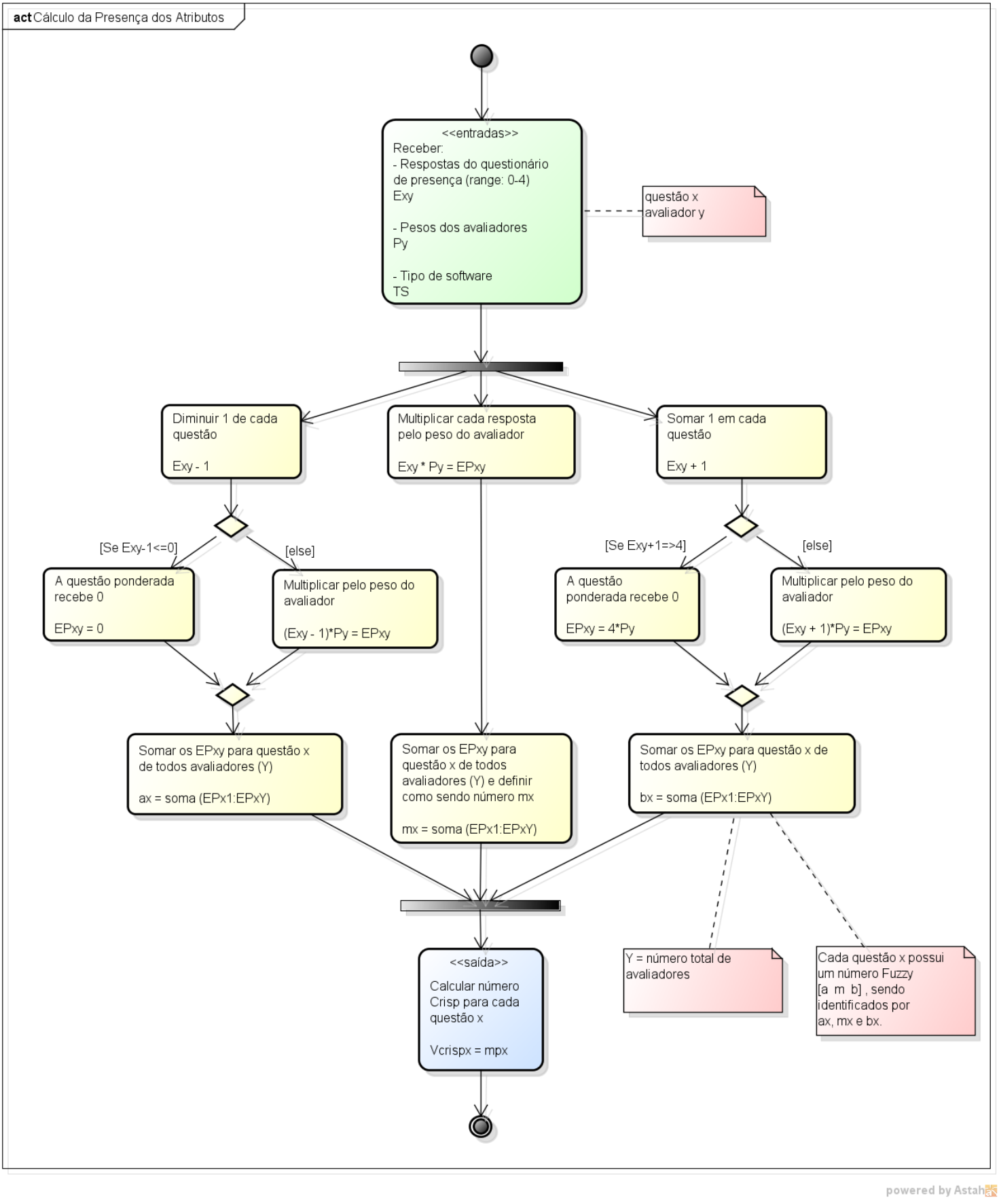


Figura A.11 - Cálculo da Presença dos Atributos

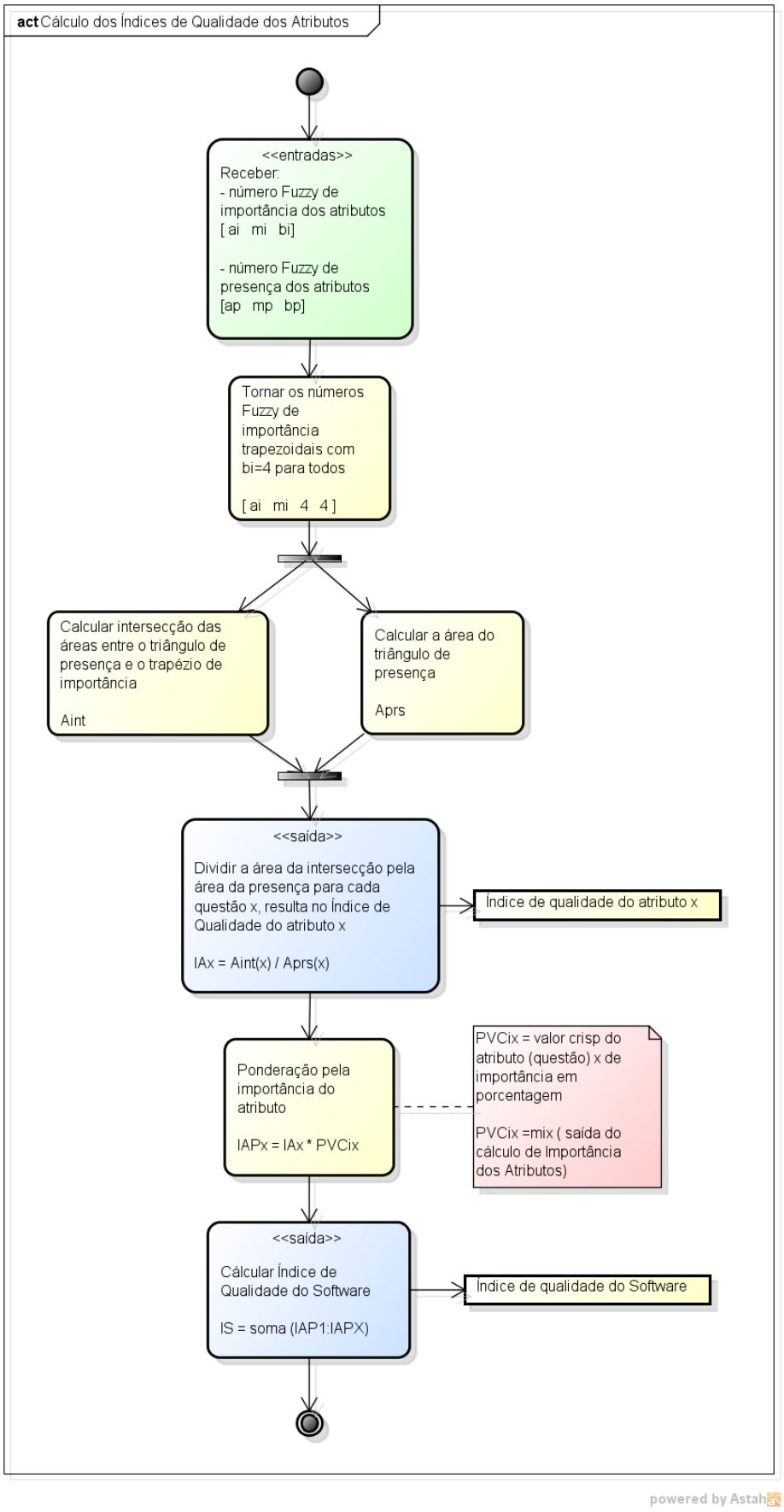


Figura A.12 - Cálculo dos Índices de Qualidade