UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ISMAEL SCHEEREN

COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTOS E DE ENGENHARIA DE SISTEMAS BASEADA EM MODELOS

Porto Alegre
2013

ISMAEL SCHEEREN

COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTOS E DE ENGENHARIA DE SISTEMAS BASEADA EM MODELOS

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica, da Universidade Federal do Rio Grande do Sul, como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.

Área de concentração: Controle e Automação

ORIENTADOR: Prof. Dr. Carlos Eduardo Pereira

Porto Alegre

2013

ISMAEL SCHEEREN

COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTOS E DE ENGENHARIA DE SISTEMAS BASEADA EM MODELOS

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia Elétrica e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____

Prof. Dr. Carlos Eduardo Pereira, UFRGS

Doutor pela TU Stuttgart – Stuttgart, Alemanha

Banca Examinadora:

Prof. Dr. Leandro Buss Becker, UFSC

Doutor pela UFRGS – Porto Alegre, Brasil

Prof. Dr. Walter Fetter Lages, PPGEE/UFRGS

Doutor pelo ITA – São José dos Campos, Brasil

Prof. Dr. Valner João Brusamarello, PPGEE/UFRGS

Doutor pela UFSC – Florianópolis, Brasil

Prof. Dr. Marcelo Pimenta, PPGC/UFRGS

Doutor pela Universite Toulouse – Toulouse, França

Coordenador do PPGEE: _____

Prof. Dr. Arturo Suman Bretas

Porto Alegre, dezembro de 2013.

DEDICATÓRIA

Dedico este trabalho à evolução silenciosa do universo.

AGRADECIMENTOS

À UFRGS e ao Programa de Pós-Graduação em Engenharia Elétrica, PPGEE, pela oportunidade de realização de trabalhos em minha área de pesquisa.

Ao Prof. Dr. Carlos Eduardo Pereira pela paciência, dedicação e pela atuação majestosa no ensino de valores como liberdade e independência.

À *General Electric Inspection Technologies GmbH* por ter disponibilizado as informações necessárias para a realização desse trabalho e à Siemens® do Brasil por disponibilizar a licença acadêmica do COMOS® para testes.

À Catarine Markus pelo apoio incondicional e pelo encorajamento durante as fases difíceis do projeto.

Aos familiares e amigos pela compreensão e paciência durante minha ausência e de forma especial ao amigo Bernardo Eltz pelas valiosas recomendações e correções no texto.

RESUMO

Na busca de maior flexibilidade, agilidade, reuso e consequente redução de custos, esforços tem sido direcionados no sentido de desenvolver métodos e ferramentas de engenharia de sistemas baseados em modelos. Apesar dos avanços recentes, as tecnologias disponíveis ainda despertam dúvidas em relação à sua aplicação prática e seus benefícios. Os maiores obstáculos estão contidos na dificuldade da integração entre ferramentas e troca de informações entre artefatos de diferentes disciplinas. Com o objetivo de comparar a Engenharia Centrado em Documentação (ferramentas CAx) com a Engenharia de Sistemas Baseada em Modelos (MBSE), esse trabalho utilizou um domínio industrial real para extrair, analisar e comparar dados quantitativos e qualitativos do projeto de engenharia. Foi desenvolvido um método de engenharia baseada em modelos com o uso da ferramenta Eclipse para a comparação com o método de engenharia vigente. A linguagem ModelicaML foi utilizada para criar os modelos abstratos enquanto que a ferramenta COMOS® da Siemens® foi utilizada para a realização dos artefatos técnicos multidisciplinares do domínio em estudo. O *software* OpenModelica foi utilizado para simular o comportamento do sistema a partir da transformação do modelo abstrato para código Modelica com o uso de *software* escrito em Java. Os dados de engenharia e de gerenciamento do projeto do Sistema de Circulação de Água foram disponibilizados pela *General Electric Inspection Technologies GmbH* e foram utilizados para a comparação entre os dois métodos analisados. Os testes demonstraram que as ferramentas MBSE necessitam de refinamento, principalmente quando conectam os modelos abstratos às plataformas de execução de projetos. Em contrapartida, MBSE se mostrou uma excelente ferramenta na comunicação entre equipes multidisciplinares, pois proporciona uma linguagem de representação de sistemas abstrata e abrangente. A interligação dos modelos abstratos desenvolvidos em ModelicaML com a plataforma de simulação usando linguagem Modelica foi fundamental na análise e melhor compreensão dos fenômenos envolvidos no processo técnico propiciando um importante avanço na antecipação da detecção de erros em projetos de sistemas de automação.

Palavras-chave: Engenharia de Sistemas. Engenharia baseada em Modelos. COMOS. Modelica.

ABSTRACT

Achieving more flexibility, agility, reuse and consequently cost reduction in scope of Systems Engineering is an industrial need. In that sense, efforts have been driven to develop Model-Based Systems Engineering tools and methods. Despite of recent progress, there are still doubts in terms of the practical use and benefits. The main issues are related to tool integration and exchange of information between multidisciplinary artifacts. This project is intended to compare Document-Based Engineering (CAx tools) and Model-Based Systems Engineering (MBSE) in scope of Industrial Automation using a real domain. Therefore, a MBSE methodology was developed centered on the Eclipse tool. The ModelicaML language was used to perform abstract modeling while COMOS® from Siemens® was used to develop the multidisciplinary artifacts necessary for the domain under investigation. Furthermore, the OpenModelica environment was used to simulate system and component behavior using object codes generated by a Java tool from the abstract models. The engineering and project management data of the Water Circulation System were made available by General Electric Inspection Technologies GmbH, which served as the case study for this comparison. Tests have shown that the tools involved on this investigation still need further development concerning maturity and exchange of information from abstract models down to domain models. On the other hand, MBSE has proven to be an important tool to match different team approaches and concerns helping on communication using conceptual-wide and abstract symbols. Connecting abstract models from ModelicaML to a simulation environment using Modelica language have been proven to be an important approach to better understand systems behavior and provided an analysis environment for early detection of errors and failures.

Keywords: Systems Engineering. Model-based Engineering. COMOS. Modelica.

SUMÁRIO

1 INTRODUÇÃO 14

1.1 OBJETIVOS DO TRABALHO 16

2 FUNDAMENTAÇÃO TEÓRICA 18

2.1 PARADIGMA DE ORIENTAÇÃO A OBJETOS EM SISTEMAS DE ENGENHARIA 20

2.2 REUSO DE COMPONENTES 21

2.3 INTEGRAÇÃO ENTRE OBJETOS MULTIDISCIPLINARES 23

2.4 ENGENHARIA DE REQUISITOS 24

2.5 ENGENHARIA CENTRADA EM DOCUMENTOS 25

2.6 ENGENHARIA DE SISTEMAS BASEADA EM MODELOS 27

2.7 ENGENHARIA DE DOMÍNIO 29

2.8 SISTEMAS FÍSICO-CIBERNÉTICOS 35

3 ESTADO DA ARTE 37

3.1 MBSE COM UML/SYSML 37

3.2 ENGENHARIA DE DOMÍNIO COM COMOS® 41

3.3 SISTEMAS FÍSICO-CIBERNÉTICOS COM MODELICA 46

3.3.1 ModelicaML 53

3.4 QUADRO COMPARATIVO 55

4 PROPOSTA 57

4.1 PROJETO CENTRADO EM DOCUMENTOS DA GEIT 57

4.2 PROPOSTA DE MÉTODO DE ENGENHARIA DE SISTEMAS 59

4.2.1 Fase 1: Análise de Requisitos 62

4.2.2 Fase 2: Arquitetura da solução 62
4.2.3 Fase 3: Detalhamento de componentes 63
4.2.4 Fase 4: Atualização do Repositório de Domínio 63
4.2.5 Fase 5: Sincronização dos objetos ModelicaML e artefatos COMOS® 64
4.2.6 Fase 6: Simulação e validação 64
4.2.7 Fase 7: Relatórios de produção 65
4.3 COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTAÇÃO E MÉTODO BASEADO EM MODELOS 67
5 VALIDAÇÃO 69
5.1 DESCRIÇÃO DO OBJETO DE ESTUDO DE CASO 69
5.2 VALIDAÇÃO DO MÉTODO PROPOSTO 73
5.2.1 Fase 1: Análise de Requisitos 73
5.2.2 Fase 2: Arquitetura da solução 74
5.2.3 Fase 3: Detalhamento de componentes 75
5.2.4 Fase 4: Atualização do Repositório de Domínio 76
5.2.5 Fase 5: Sincronização dos objetos 77
5.2.6 Fase 6: Simulação e validação 78
5.2.7 Fase 7: Relatórios de produção 80
5.3 COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTAÇÃO E MÉTODO BASEADO EM MODELOS 80
5.3.1 Mudança de tamanho da bomba principal 81
6 CONCLUSÃO 86
6.1 TRABALHOS FUTUROS 90

LISTA DE ILUSTRAÇÕES

Figura 1 – Processo de desenvolvimento de sistemas baseado em MBSE (EIGNER;GILZ e ZAFIROV, 2012). 29

Figura 2 – Visões de um artefato multidisciplinar (MAGA e JAZDI, 2009). 31

Figura 3 – Representação das etapas da Engenharia de Domínio (MAGA e JAZDI, 2009). 32

Figura 4 – Camadas de um Repositório de Domínio (MAGA e JAZDI, 2009). 33

Figura 5 – relação semântica entre diagramas SysML. 39

Figura 6 – Exemplo de esquema elétrico usado para fabricação de Sistemas de Automação. 40

Figura 7 – Propriedades de um objeto COMOS®. 43

Figura 8 – Conectores em COMOS®. 44

Figura 9 – Componentes multidisciplinares de Modelica (MODELICA ASSOCIATION, 2012). 47

Figura 10 – Comparação entre ModelicaML *Internal Class Definition* e Modelica *Connection Diagram* (POP;BĂLUȚĂ e FRITZSON, 2007). 48

Figura 11 – Sistemas com múltiplas disciplinas modelados com Modelica (MODELICA ASSOCIATION, 2013). 49

Figura 12 – Estrutura de diagramas ModelicaML (POP;BĂLUȚĂ e FRITZSON, 2007). 54

Figura 13 – Processo de engenharia centrado em documento. 58

Figura 14 – Troca de informações entre distintas representações de objetos nos modelos. 61

Figura 15 – Método de projeto proposto – MoDElS.iDEAS. 62

Figura 16 – Validação de requisitos de entrada. 65

Figura 17 – Interligação entre diferentes camadas de abstração. 66

Figura 18 – Proposta de comparação entre métodos. 68
Figura 19 – Estrutura do sistema de Ensaio Não Destrutivo por ultrassom. 70
Figura 20 – Componentes constituintes do WCS. 71
Figura 21 – Representação do WCS em objetos do OpenModelica. 72
Figura 22 – Exemplo de Diagrama de Requisitos do ModelicaML. 73
Figura 23 – Arquitetura física do WCS representada no OpenModelica. 74
Figura 24 – Estrutura de componentes do sistema WCS. 75
Figura 25 – Descrição do comportamento de objetos. 76
Figura 26 – Estrutura de componentes do Domínio. 77
Figura 27 – Integração dos parâmetros simulados na ferramenta COMOS®. 78
Figura 28 – Exemplo de avaliação de requisitos do domínio WCS. 79
Figura 29 – Desenho tridimensional do WCS. 80
Figura 30 – Requisitos do sistema relacionados à bomba principal. 81
Figura 31 - Curva característica da bomba Grundfos NB32-160.1/139 (GRUNDFOS, 2013). 82
Figura 32 – Curva característica da bomba principal representada em código Modelica. 82
Figura 33 – Simulação dos níveis dos tanques do sistema antes da modificação da bomba. 83
Figura 34 – Simulação dos níveis dos tanques do sistema após a modificação da bomba. 83
Figura 35 – Detalhe da integração de parâmetros do modelo abstrato no COMOS®. 84

LISTA DE TABELAS

Tabela 1. Comparação entre diferentes perspectivas de desenvolvimento de Sistemas.....56

Tabela 2. Comparação entre diferentes ferramentas que suportam o desenvolvimento de Sistemas. 56

LISTA DE ABREVIATURAS

CAD – Computer-Aided Design

CAE – Computer-Aided Engineering

CAM – Computer-Aided Manufactoring

CIF – Compositional Interchange Format

COTS - Commercial Off-The-Shelf Components

CPS – Cyber-Physical Systems

DB – Data base

DE – Domain Engineering

DRE – Domain Repository Engine

E-CAD – Electronic and Electric Computer-Aided Design

FB – Function Block

FMI – Functional Mockup Interface

FMU – Functional Mockup Units

GEIT – General Electric Inspection Technologies GmbH

GLP – Gás Liquefeito do Petróleo

MBSE – Model-Based Systems Engineering

ModelicaML – Modelica Modeling Language

PLC – Programmable Logic Controller

PLM – Plant Life Cycle Management

SysML – Systems Modeling Language

UML – Unified Modeling Language

WCS – Water Circulation System

XMI – XML Metadata Interface

1 INTRODUÇÃO

Um sistema é a combinação de elementos que interagem entre si para atingir um ou mais propósitos determinados. Sistema de Interesse é o sistema cujo ciclo de vida se considera (HASKINS e FORSBERG, 2011). No presente trabalho, Sistema de Interesse é limitado a projetos de automação industrial.

Numa visão de longo prazo, (INCOSE, 1997) descreveu a missão da Engenharia de Sistemas como sendo a de garantir o desenvolvimento e execução integrados de produtos de automação que atendam às expectativas das partes interessadas em relação aos custos, prazos e requisitos de risco de falha do projeto. A publicação vai além, listando os seguintes objetivos para o paradigma:

- fornecer um processo estruturado para integrar e ligar requisitos, cronograma e verificação;
- possibilitar à equipe de projeto trabalhar com um conjunto integrado de requisitos e processos;
- possibilitar a integração dos requisitos e etapas de engenharia antes que *hardware* e *software* estejam disponíveis, e;
- reduzir tarefas de reengenharia desnecessárias e onerosas decorrentes de dificuldades de integração ou omissões.

Passados mais de 15 anos da publicação desse documento, que serviu como referência para o desenvolvimento desta área de conhecimento e apesar de toda a evolução feita no sentido de resolver problemas no âmbito de automação industrial, algumas perguntas permanecem sem respostas. Problemas como a maturidade das ferramentas disponíveis (MAGA;JAZDI e GÖHNER, 2011), a falta de integração dos modelos envolvidos em projetos de engenharia de sistemas e as dificuldades de simulação virtual (KERNSCHMIDT *et al.*, 2013) permanecem em aberto.

Na busca de melhores resultados no que diz respeito à Engenharia de Sistemas de Automação, novos métodos são testados para a concepção, o desenvolvimento e a execução de projetos. Segundo uma visão simplificada, apresentada por PROJECT MANAGEMENT INSTITUTE (2013), as três principais variáveis que impactam no resultado de um projeto são custo, prazo e escopo. Portanto, o aumento do retorno de desenvolvedores e fabricantes de Sistemas de Automação está ligado com a redução do custo, a correta definição das necessidades do cliente e a redução da duração do ciclo de desenvolvimento até o lançamento do produto no mercado.

Com o objetivo de melhorar os resultados alcançados com os métodos tradicionais de engenharia, algumas tecnologias foram importadas da Engenharia de *Software* e aplicadas no contexto de Sistemas de Automação. Dentre as novas propostas, se destaca o amplo uso da tecnologia de orientação a objetos, que traz nos mecanismos de encapsulamento, polimorfismo e herança, os fundamentos para o reuso dos artefatos técnicos em diferentes projetos.

Existe uma curva de aprendizado que se faz necessária para adoção de novos métodos de trabalho. Há resistência, no meio industrial, à adoção de novos métodos que demandem incremento inicial de tempo de projeto e execução. A curva de aprendizagem necessária para mudar o método e o conhecimento já armazenado em ferramentas antigas requer que novas propostas sejam testadas e sua eficácia comprovada através de aplicações práticas antes de serem inseridas de forma ampla no mercado.

A Engenharia de Sistemas Baseada em Modelos (MBSE – *Model-Based Systems Engineering*) procura estabelecer uma plataforma que proporcione uma visão abstrata da concepção de um Sistema de Automação (LONG e SCOTT, 2012). Essa característica tem se mostrado útil ao facilitar o desenvolvimento de sistemas de maior complexidade, que combinam tecnologias oriundas de diferentes disciplinas e executam troca de informações entre seus objetos e com outros sistemas. Além disso, há melhor interação entre as diferentes equipes

disciplinares, uma vez que o modelo faz com que os times de desenvolvimento trabalhem em conjunto. Por outro lado, procedimentos baseados em MBSE ainda pecam em não conectar modelos abstratos à camada de realização, como acontece na Tecnologia de Informação com as técnicas de transformação de modelo para texto.

Já a Engenharia de Domínio (JAZDI *et al.*, 2010) busca reaplicar artefatos técnicos previamente desenvolvidos em projetos que possuam características semelhantes usando o mínimo esforço possível. Esse conceito concentra-se em proporcionar um ambiente que integre os componentes de várias disciplinas presentes em um sistema de automação de forma a realizá-los. A Engenharia de Domínio visa desenvolver uma plataforma condizente com PLM *(Product or Plant Life Cycle Management (STARK, 2011))*, que trata desde as especificações iniciais de sistemas até o seu descarte, passando por operação e manutenção. Por outro lado, a Engenharia de Domínio não possui a flexibilidade e a exatidão das notações que são oferecidas pela MBSE.

Embora tenham ocorridos avanços na última década no que se refere aos fundamentos para a engenharia baseada em modelos, o desenvolvimento de uma plataforma integrada que proporcione a redução de custo almejada por fabricantes de sistemas ainda carece de pesquisa. As ferramentas atuais são incapazes de lidar com diferentes plataformas, e a integração entre objetos abstratos e físicos ainda precisa ser melhor estudada, como destaca ACATECH (2011) no contexto de Sistemas Físico-Cibernéticos.

1.1 OBJETIVOS DO TRABALHO

Frente ao cenário exposto e levando-se em conta que os principais custos de engenharia de Sistemas de Automação estão relacionados ao número de horas de desenvolvimento e à incidência de erros de projeto, o presente trabalho avaliará o impacto da implantação de métodos de Engenharia de Sistemas baseados em Modelos nesses quesitos.

Para tanto, será desenvolvido um método para integrar os conceitos de MBSE, Engenharia de Domínio e Sistemas Físico-Cibernéticos no contexto de sistemas de automação com o objetivo de proporcionar uma plataforma integrada de desenvolvimento e verificação.

O Sistema de Circulação de Água utilizado pela empresa *General Electric Inspection Technologies GmbH* (GEIT) nos equipamentos de Ensaio Não Destrutivo por Ultrassom será usado para validar a proposta. Os resultados obtidos serão comparados com o método centrado em documentação, utilizada pela empresa, de forma a possibilitar a análise do tempo necessário para criar variantes de componentes no escopo do estudo de caso. A partir dos resultados será possível avaliar quais os benefícios que o atual nível de desenvolvimento da MBSE pode trazer para as empresas que produzem Sistemas de Automação. Também será possível traçar um rumo para que melhores resultados sejam alcançados e soluções para os problemas atuais sejam discutidas.

2 FUNDAMENTAÇÃO TEÓRICA

No âmbito de Automação Industrial, um sistema consiste de diversas unidades que trabalham integradas e formam uma rede cooperativa. Neste trabalho, sistema consiste em blocos funcionais criados por humanos e que possuem um objetivo. Blocos esses que podem ser *hardware*, *software*, dados, procedimento ou um indivíduo (WEILKIENS, 2011).

Um sistema deve ser projetado de forma consistente para atingir os objetivos ao longo de todas as suas fases: desenvolvimento, realização, utilização e descarte. Portanto, o processo de engenharia deve lidar com o fato de que um erro cometido na fase inicial do projeto muitas vezes só possa ser detectado nas fases finais de realização ou mesmo durante a sua utilização. Nos períodos mais tardios, não existem maneiras de influenciar as propriedades de um sistema de forma abrangente (WEILKIENS, 2011). Quanto melhores forem as ferramentas e métodos de projeto no sentido de detectar um erro, melhor o resultado esperado tanto em custo como em qualidade e atendimento dos requisitos.

A Engenharia de Sistemas é uma abordagem interdisciplinar com métodos estabelecidos para se atingir a realização bem sucedida de um sistema. Esse processo de engenharia é focado em determinar os requisitos funcionais na fase inicial do projeto baseando-se nas necessidades dos clientes. Após a documentação e análise dos requisitos, o processo segue com a engenharia e validação do sistema considerando o problema como um todo. Essa abordagem de gerenciamento de projetos deve considerar as necessidades técnicas e gerenciais com o objetivo de fornecer um produto de qualidade (HASKINS e FORSBERG, 2011).

Ao longo do processo de desenvolvimento de sistemas, a equipe deve questionar os objetivos do projeto, o que o sistema precisa fazer para atingir os objetivos e quais alternativas de solução melhor se aplicam para alcançar o objetivo proposto em termos de custo, prazo e qualidade. Também devem ser analisados os esforços de engenharia (restrições em termos de

custo, tempo, recursos) e natureza do desafio, como relata (BLANCHARD e FABRYCKY, 2010).

Vale ressaltar que essas questões são apropriadas tanto para a especificação de um artefato técnico simples (um disjuntor ou um mancal, por exemplo) como para o desenvolvimento de um sistema complexo composto de vários artefatos técnicos interligados. Contudo, com o avanço da complexidade dos sistemas de automação, as perguntas que embasavam o desenvolvimento de sistemas no passado não respondem às demandas de predição de erros, previsibilidade de comportamentos e confiabilidade dos projetos modernos.

É notório o incremento de complexidade ocorrido no âmbito desses sistemas. Segundo (CALVANO e JOHN, 2004), o incremento da complexidade originou-se na crescente integração de diferentes áreas do conhecimento causada pelo contínuo desenvolvimento das tecnologias da informação e comunicação. Esse trabalho também argumenta que o incremento de complexidade causa falhas que se tornam inevitáveis e imprevisíveis, atribuindo à disciplina de engenharia de sistemas a busca prioritária e urgente de meios de tratar tal complexidade e suas consequências.

A engenharia de sistemas é uma técnica multidisciplinar para desenvolver soluções balanceadas, em resposta a necessidades de clientes (FRIEDENTHAL;MOORE e STEINER, 2011). Após a revisão histórica da evolução das teorias de engenharia de sistemas e sistema de sistemas (*System of Systems*), (GOROD;SAUSER e BOARDMAN, 2008) constatou que não foi definida uma plataforma completa apta a lidar com a integração entre diferentes disciplinas, domínios e sistemas. (KERNSTINE, 2013) afirma que a Engenharia de Sistemas carece de plataformas de simulação que contemplem a previsibilidade de erros.

No contexto deste trabalho, disciplina entende-se por ramo do saber humano, por exemplo *software*, elétrica, mecânica, termodinâmica, hidráulica, biologia, controle, tempo real. Já as diferentes esferas de ação nas quais as diferentes disciplinas são empregadas pela

Engenharia de Sistemas são aqui chamadas de domínios, como aeronáutica, petróleo e gás, agricultura ou siderurgia.

No sentido de minimizar os erros de projeto ou falhas durante a operação causados por problemas durante o desenvolvimento de sistemas complexos, o presente trabalho argumenta que a conexão entre métodos de engenharia de sistemas baseados em modelos e métodos de engenharia de domínio devem ser estudados e a sua integração automatizada. Antes de avançar no detalhamento dessas teorias, importantes características de projetos de Engenharia de Sistemas precisam ser revisadas: Orientação à Objeto, Reuso e Integração entre componentes.

2.1 PARADIGMA DE ORIENTAÇÃO A OBJETOS EM SISTEMAS DE ENGENHARIA

Uma diferença básica entre a engenharia baseada em modelos oriunda da Engenharia de *Software* e a aplicada em Engenharia de Sistemas consiste na necessidade da última ter adicionalmente uma visão focada nos objetos físicos que executarão cada passo na obtenção do resultado final. Essa necessidade está fundamentada na relação próxima da Engenharia de Sistemas com disciplinas de componentes físicos, como é o caso de objetos mecânicos ou elétricos. Frequentemente fornecedores de máquinas ou plantas de automação usam componentes de prateleira (*commercial off-the-shelf (COTS) components*) para a fabricação de sistemas (BROWN e WALLNAN, 1996). Dessa forma, durante o desenvolvimento de um sistema industrial, é necessário realizar um processo de transição entre funções abstratas, que foram levantadas durante a fase de análise de requisitos e especificação, para uma abordagem que leva em conta os componentes físicos do sistema. Isso se faz necessário uma vez que serão os objetos físicos que executarão as funções abstratas para atingir o objetivo do sistema. As funções abstratas contidas nos objetos abstratos são utilizadas para simulação do projeto, que se torna mais condizente com a realidade quando as limitações dos objetos físicos são adicionadas no modelo (FRIEDENTHAL;MOORE e STEINER, 2011).

A linguagem unificada de modelagem ou UML (*Unified Modeling Language*) é uma linguagem visual para especificar, construir e documentar artefatos de sistemas (OBJECT MANAGEMENT GROUP, 2011b). Trata-se de uma linguagem orientada a objetos oriunda da Engenharia de *Software* para desenvolver aplicações e componentes. É usada em sistemas que requeiram interação com pessoas, setores da indústria, empresas e processos de negócios. UML é composta de diagramas que possuem propósito definido e um conjunto de símbolos específicos.

A UML é especificada por uma linguagem chamada MOF (*Meta Object Facility*), que a define de modo a possibilitar extensões de significado para elementos da linguagem. Os Estereótipos são empregados para modificar o significado original dos elementos enquanto que atributos (chamados de *Tagged Value*) são empregados para dar sentido às propriedades. Essas combinações de nome e valor podem ser utilizadas indefinidamente para atribuir propriedades a um elemento. A partir desse mecanismo, extensões são criadas para o UML com o propósito de atender uma área específica do conhecimento. Fala-se então de perfil UML (*UML profile*) (OBJECT MANAGEMENT GROUP, 2011a). O subconjunto UML criado para atender as demandas de engenharia de sistemas se chama SysML ou *Systems Modeling Language* (OBJECT MANAGEMENT GROUP, 2012).

2.2 REUSO DE COMPONENTES

A sistematização de processos de desenvolvimento de sistemas visa o reuso dos conhecimentos e dados gerados ao longo do tempo a partir de projetos anteriores. Tem-se tentado criar mecanismos que capturem informações oriundas de experiências passadas para que possam ser reutilizadas em projetos futuros. Nesse contexto, MBSE juntamente com a orientação a objetos fornecem uma plataforma adequada para o reaproveitamento de experiências passadas em comparação aos métodos centrados em documentação.

O reuso de conhecimentos (em forma de modelos, código, parâmetros) visa o estabelecimento de base sólida para a elaboração de um novo sistema. Dessa forma, o fabricante de sistemas consegue diminuir seus custos através de reaproveitamento de esforços e diminuição de erros, uma vez que reutiliza componentes que já foram testados e validados em outras aplicações.

Os métodos baseados em modelos abstratos podem ser úteis porque capturam o conhecimento gerado ao longo do processo de forma visual e intuitiva, facilitando o seu reaproveitamento em projetos futuros. Muito embora um projeto orientado à documentação possa ter mais detalhes no que diz respeito a sua realização física, o nível de especialização e detalhamento dos componentes nesse tipo de método prejudica o reaproveitamento em projetos semelhantes. Isso significa dizer que, o projeto de um acionamento elétrico por contatores, quando centralizado em documentos, somente poderá ser utilizado em projetos exatamente iguais ou terá de sofrer modificações manuais. Porém, caso o mesmo acionamento tenha sido desenvolvido numa plataforma flexível, orientada a modelos, o acionamento pode ser utilizado em outros projetos a partir da alteração de alguns parâmetros, como corrente e quantidade de ciclos de acionamento, passados diretamente de modelos abstratos para o modelo do componente.

Do exemplo acima, se percebe uma característica interessante dos componentes físicos. A capacidade de reutilização do seu modelo virtual se eleva à medida que é desenvolvido em termos de propriedades ou parâmetros. Ademais, modelos de componentes devem ser criados em função de um domínio específico. Um modelo de componentes hidráulicos apto a ser empregado em prensas hidráulicas possui características distintas de componentes hidráulicos usados em simples movimentação de materiais. O modelo do componente deve ser desenvolvido de acordo com o contexto no qual será empregado, pois guarda conhecimentos específicos daquele domínio. Um modelo estruturante pode ser refinado por modelos mais

específicos e sempre que um modelo estruturante é utilizado, outros modelos que o refinam também o são (PAREDIS *et al.*, 2010). Aqui se tem o conceito de herança aplicado em modelos de componentes físicos.

Algumas porções de modelos de sistemas se repetem. Existem padrões para instanciar essas porções, ou seja, alguns parâmetros ou dados que devem ser alterados para que o modelo possa ser reaplicado. A necessidade de adaptação para diferentes padrões pode ser cumprida por transformações entre modelos (LUCREDIO *et al.*, 2010).

Modelos de transformação podem ser criados para capturar os padrões de reuso para facilitar a conversão entre modos de apresentação, mantendo ou não o nível de abstração. Um modelo analítico pode ser usado para refinar um modelo descritivo, como destaca (PAREDIS *et al.*, 2010). Portas estruturais de componentes descritivos são mapeadas para portas analíticas em modelos analíticos, de forma a interligar modelos abstratos a modelos de simulação, da mesma forma que propriedades descritivas são mapeadas para propriedades analíticas.

2.3 INTEGRAÇÃO ENTRE OBJETOS MULTIDISCIPLINARES

Dada a diversidade de disciplinas envolvidas no desenvolvimento de Sistemas de Automação e a complexidade que daí deriva, a criação de modelos abstratos visa a elaboração de objetos virtuais que se destinem a mapear comportamentos de objetos reais de naturezas diferentes.

Para tornar os modelos mais concisos, é necessário que as representações de objetos de distintas disciplinas possam trocar informações. Os dados podem transitar sob a forma de propriedades, parâmetros e descrições físicas entre representação de objetos abstratos e representação de objetos físicos.

Com o objetivo de melhorar o desempenho do sistema, a simulação virtual dos modelos pode ser usada de modo a prever comportamentos. De modo a facilitar a simulação, quando se trata de modelagem de sistemas industriais multidisciplinares, os objetos modelados devem ser

mapeados de forma a refletir a estrutura do sistema físico e o comportamento dos seus componentes.

A interligação entre objetos abstratos e físicos é um campo de estudo recente. Durante o trabalho, o conceito de Sistemas Físico-Cibernéticos, que combina sistemas cibernéticos e sistemas físicos, será detalhado.

2.4 ENGENHARIA DE REQUISITOS

Um requisito descreve algo que precisa ser realizado, transformado, produzido, provido ou limitado (HASKINS e FORSBERG, 2011). Requisitos especificam o que precisa ser obtido, sem, contudo, definir como. Serve como guia para que a equipe de engenharia saiba como o resultado deverá ser atingido.

A gestão de requisitos é realizada a partir da integração de necessidades provindas de múltiplas fontes. Os requisitos podem ter sua origem em procedimentos de trabalho, especificações, normas, decisões do cliente e diretivas de gestão.

A etapa de especificação dos requisitos objetiva minimizar os esforços tardios necessários para integrar os componentes multidisciplinares. Com ela, é possível diminuir ou administrar ambiguidades, conflitos e omissões para que o time de engenharia trabalhe num único e validado conjunto de requisitos. Proporciona a identificação de requisitos que requeiram análise posterior ou simulação de modo a estabelecer objetivos quantificáveis e mensuráveis.

A simulação virtual impacta a especificação dos limites do sistema de forma a proporcionar um mecanismo de validação dos requisitos inicialmente explicitados. Uma plataforma de engenharia de sistemas que proporcione mecanismos de simulação traz benefícios para a análise de requisitos.

A separação entre requisitos essenciais e requisitos técnicos corresponde à diferenciação entre requisitos oriundos do usuário ou do próprio sistema que deve ser realizado (WEILKIENS, 2011, l. 1143). Um requisito no contexto da modelagem é representado por uma

sentença textual e nenhuma suposição deve ser feita, quanto à realização, quando um requisito é introduzido no processo de análise.

Antes de se detalhar o método proposto, faz-se necessário destacar as teorias de engenharia de sistemas envolvidas neste trabalho.

2.5 ENGENHARIA CENTRADA EM DOCUMENTOS

Segundo WEILKIENS (2011), a Engenharia Baseada em Documento é o método tradicional de engenharia de sistemas que se caracteriza pela geração de especificações textuais e documentos de projeto que são trocados entre clientes, usuários e desenvolvedores. Os requisitos do sistema e as informações do projeto são expressos nesses documentos e a ênfase é controlar a documentação para se assegurar que as informações nela contidas sejam válidas, completas e consistentes.

As atividades de engenharia são planejadas através de estimativa de tempo e esforço necessários para gerar a documentação e o andamento do projeto é avaliado em relação à conclusão desses documentos (WEILKIENS, 2011). A análise e validação dos requisitos do sistema (desempenho, confiabilidade, segurança, propriedades dos materiais) são executadas independentemente pelas equipes de distintas disciplinas usando ferramentas específicas.

Apesar desse método criar uma documentação rica em detalhes, possui algumas limitações fundamentais. O fato da informação estar distribuída em diferentes documentos acarreta em morosidade quando se busca por uma informação específica. Pode causar armazenamento redundante de dados e gerar inconsistências. Além disso, o método falha na integração e coerência das informações, principalmente no que diz respeito à relação entre requisitos, dados de projeto, análises de engenharia e informações dos testes de validação. Isso faz com que a integração entre os requisitos e dados de projeto ao nível de sistema não sejam sincronizados com os dados contidos nos documentos – desenhos e especificações de *hardware* e *software*.

A descentralização da documentação leva a dificuldades no reaproveitamento dos esforços de engenharia em novos projetos, além de criar problemas de inconsistência em projetos de maior envergadura (RAUSCH *et al.*, 2005) inclusive nas fases de operação e manutenção.

Os projetos de engenharia são desenvolvidos de tal forma que cada disciplina utiliza ferramentas especializadas. A disciplina de engenharia elétrica utiliza-se de E-CADs, a disciplina de mecânica (incluindo pneumática e hidráulica) utiliza-se de CAD, CAE e CAM e a disciplina de *software* utiliza-se de ambientes de programação, compilação e depuração específicas para a plataforma de *hardware* empregada.

A integração das informações entre as disciplinas é uma característica desejável, já que alterações em parâmetros em componentes de uma disciplina podem ter impacto no resultado do projeto de outra disciplina. Nesse método de trabalho, a sincronização dessas interfaces é realizada manualmente através de referências textuais, chamadas de *Tags*. A incapacidade desse método de engenharia em lidar de forma dinâmica com as alterações em características interdisciplinares é um fator negativo que tem impulsionado novos estudos (SECCHI *et al.*, 2007; LEE, 2008; WEHRMEISTER;PEREIRA e RAMMIG, 2013).

Não se pode menosprezar, porém, o avanço realizado por essa abordagem no que diz respeito ao detalhamento dos processos de manufatura. A vasta simbologia gerada ao longo de décadas é utilizada em desenhos mecânicos e esquemas elétricos e eletrônicos para representar os componentes constituintes do sistema. Os símbolos fornecem uma visão detalhada e exata do funcionamento de um objeto e de como ele deve ser fabricado ou montado. Portanto, os procedimentos de interligação, montagem, lubrificação, acabamento e incluindo o funcionamento da máquina são detalhados com o uso dos símbolos desenvolvidos através da centralização em documentos, bastando que se conheçam os símbolos envolvidos.

2.6 ENGENHARIA DE SISTEMAS BASEADA EM MODELOS

Com o objetivo de resolver as dificuldades enfrentadas pela indústria com os métodos de engenharia tradicionais, oriundas principalmente do incremento da complexidade dos sistemas atuais e dos seus projetos, tem-se buscado alternativas de engenharia que suportem uma visão mais abrangente dos sistemas. Segundo FRITZSON (2004), que traz uma visão abstrata em relação ao tema, um sistema é um objeto ou conjunto de objetos cujas propriedades se quer estudar. Com o objetivo de simular o comportamento do sistema, pode-se exercitar as suas entradas realizando experimentos. Porém, experimentos podem ser dispendiosos e perigosos ou ainda o sistema pode não existir para viabilizá-los. Para resolver esse impasse, existem formas de se criar um modelo virtual do sistema. Pode-se estabelecer uma plataforma onde seja possível aplicar experimentos de modo a simular e responder perguntas sobre esse sistema antes mesmo que ele seja realizado fisicamente. A partir da simulação, um modelo virtual pode ser usado para aperfeiçoar o seu comportamento.

Com vistas a proporcionar uma plataforma de simulação e otimização de Sistemas de Automação além de capacitar as equipes de engenharia a desenvolver um projeto partindo de uma visão mais abstrata, tem-se buscado o desenvolvimento da teoria de *Model-Based Systems Engineering* (MBSE) ou Engenharia de Sistemas Baseada em Modelos (HYBERTSON, 2009).

Modelar sistemas aumenta o entendimento de projetos complexos e proporciona ferramentas para desenhá-los e aperfeiçoá-los, além de facilitar a verificação dos seus requisitos antes mesmo da sua construção física.

Os problemas enfrentados por engenheiros de sistemas usando MBSE estão relacionados à interação entre as diferentes disciplinas e/ou domínios. Também são enfrentados desafios na troca de informações entre múltiplas tecnologias e a interferência causada uma na outra. Deve-se lidar com a multiplicidade de componentes e com interfaces complexas frequentemente tendo que possuir uma visão multidisciplinar do contexto no qual o sistema sob

análise se insere. Problemas relacionados à ambiguidade e coerência das especificações, disponibilidade de informações, rastreabilidade, verificação e validação da proposta devem ser abordados. A MBSE vem ao encontro dessas necessidades trazendo símbolos genéricos e abstratos da Engenharia de *Software*. Portanto, os entregáveis de um projeto de sistemas usando modelos são especificações, desenho do sistema ou arquitetura, análise de possibilidades de solução, resultados de simulação e planos de teste.

Os modelos são usados para representar um sistema em uma camada abstrata e proporcionam melhor comunicação entre as equipes do projeto desde a fase de elaboração de requisitos. A utilização de símbolos específicos propicia rigor e precisão às especificações. O reconhecimento precoce de erros acaba preservando a competência da equipe multidisciplinar envolvida porque evita a incidência posterior de defeitos ou falhas.

O processo descrito por (EIGNER;GILZ e ZAFIROV, 2012) representado na Figura 1 é uma adaptação do método conhecido como Modelo V de engenharia de sistemas e inclui o aspecto da simulação virtual para validação do modelo. Parte-se da análise dos requisitos do sistema para criar modelos que possam ser simulados em ambientes virtuais. Na medida em que se avança no detalhamento e teste dos objetos, as propriedades são refinadas de forma a assegurar o funcionamento do sistema de acordo com o desejado. Após o desenvolvimento dos componentes em ferramentas específicas para cada disciplina, os objetos são detalhados e integrados, partindo-se para a etapa de testes físicos.

A diminuição de custos de manutenção é também um objetivo atribuído a projetos concebidos com o uso de modelos, uma vez que o sistema se torna consistente e modos de operação complexos são simulados e previstos. Em contrapartida, a inserção de MBSE pode significar a necessidade de aprendizados extras para a equipe envolvida, uma vez que novas ferramentas, procedimentos e símbolos são inseridos no processo.

Mesmo trazendo benefícios em relação ao entendimento e análise desde a fase inicial, os métodos baseados em MBSE não estão conectados com a camada de realização de projetos. Ou seja, partindo-se de modelos, não é possível extrair relatórios, desenhos mecânicos e esquemas elétricos que possibilitem a fabricação de componentes.

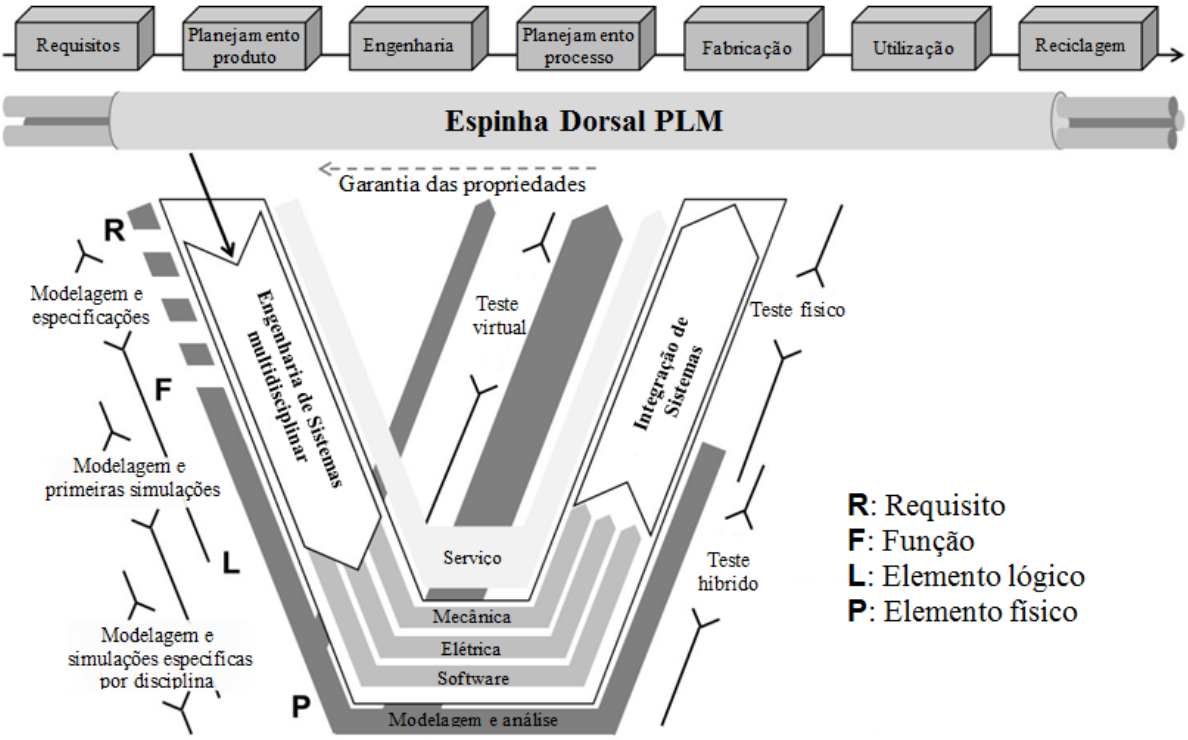


Figura 1 – Processo de desenvolvimento de sistemas baseado em MBSE (EIGNER;GILZ e ZAFIROV, 2012).

2.7 ENGENHARIA DE DOMÍNIO

A automação industrial possui uma ampla gama de aplicações e as soluções apresentadas têm formas variadas. Produtos de automação são caracterizados como sendo máquinas que executam sequências pré-definidas enquanto que plantas industriais são sistemas complexos que se destinam a executar um processo (GROOVER, 2007).

A Engenharia de Domínio objetiva diminuir os custos de projeto de equipamentos de automação a partir da reutilização de artefatos técnicos. Trata-se de um conceito importado da

Engenharia de *Software*, que agrupa uma família de problemas de acordo com determinadas características buscando soluções similares para projetos com requisitos distintos. O principal conceito da abordagem de Engenharia de Domínio é a reutilização (ALMEIDA, 2009).

A solução está baseada no desenvolvimento de 3 áreas: engenharia de domínio, engenharia de aplicação e repositório de domínio (MAGA e JAZDI, 2009). A Engenharia de Domínio vem sendo aplicada em Sistemas de Automação com bons resultados no que diz respeito à reutilização de componentes de *software* de PLC (*Programmable Logic Controller*) (JAZDI *et al.*, 2010).

Em relação à modelagem de objetos de *software*, (KIPPER, 2010) afirmou que os FBs (*function blocks*) de PLCs (*Programmable Logic Controllers*) caracterizam-se por ser a unidade comportamental reutilizável básica com relação à disciplina de Tecnologia da Informação, já que possuem códigos de aplicação e capacidade de se ligar a dados diferentes a cada reutilização, chamados DBs (*data blocks*).

Porém a Engenharia de Domínio aplicada a Sistemas de Automação ainda precisa resolver questões que vão além dos objetos de *software*, como a relação entre o processo técnico e o sistema técnico de automação, dependências físicas e documentação (MAGA e JAZDI, 2009). A diferença entre a Engenharia de Domínio empregada do âmbito da Engenharia de *Software* e da sua variação desenvolvida especificamente para a Engenharia de Sistemas está na presença de distintas disciplinas, como representa a Figura 2. A integração entre objetos de disciplinas distintas ainda é um assunto que carece de desenvolvimento (MAGA e JAZDI, 2010; JAZDI;MAGA e GOEHNER, 2011).

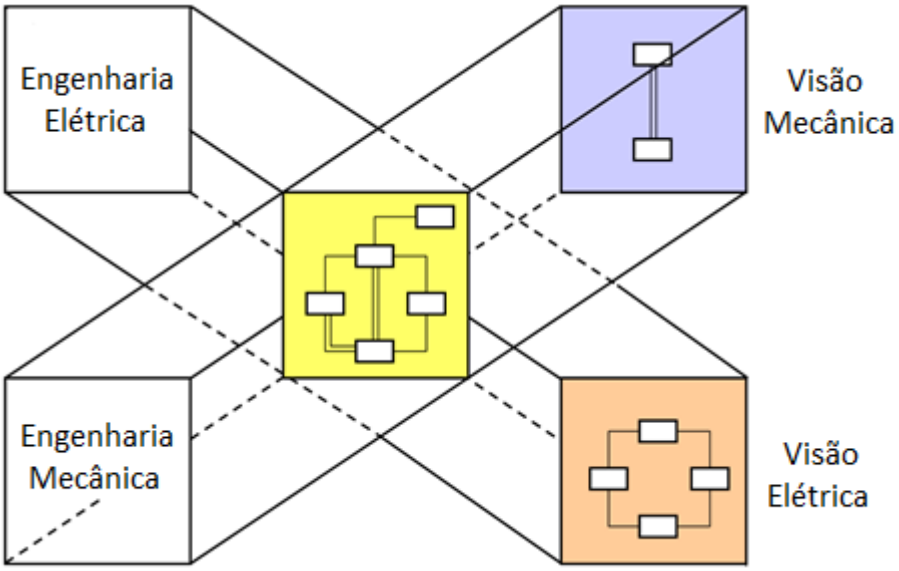


Figura 2 – Visões de um artefato multidisciplinar (MAGA e JAZDI, 2009).

O repositório do domínio caracteriza-se por ser um local onde informações previamente identificadas e organizadas são armazenadas para que sejam posteriormente reutilizadas em aplicações semelhantes (MAGA e JAZDI, 2009). As informações são adquiridas durante a construção do repositório, fase que se denomina engenharia de domínio, e durante o desenvolvimento do repositório, ou seja, durante a aplicação dos conhecimentos em novas soluções. As novas experiências são transformadas em conhecimento que atualizam os artefatos do repositório. A Figura 3 apresenta um esquema de como a Engenharia de Domínio realimenta o repositório a partir de novas experiências.

Um artefato técnico pode ser classificado quanto a sua capacidade de reutilização:

- Tipo A: artefatos estáticos, que podem ser reutilizados em diferentes aplicações sem sofrerem alterações;
- Tipo B: artefatos configuráveis e parametrizáveis, que podem ser adaptados aos requisitos dos clientes e/ou aplicações;
- Tipo C: artefatos genéricos, que contém apenas indicações quanto a funcionalidades e interfaces em uma abstração mais alta.

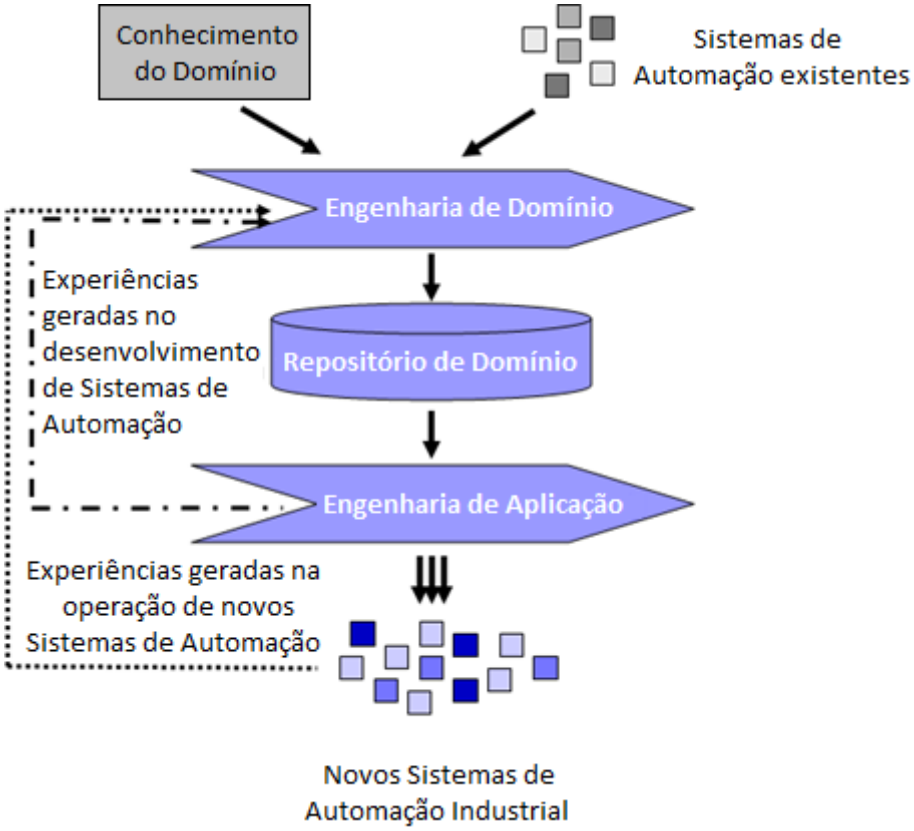


Figura 3 – Representação das etapas da Engenharia de Domínio (MAGA e JAZDI, 2009).

Projetos de sistemas de automação requerem que os artefatos guardados consigam armazenar informações relativas a *software* e *hardware* e que eles se relacionem entre si, ou seja, artefatos lógicos e físicos devem ser interdependentes.

Um artefato de *hardware* possui informações adicionais como dimensões físicas, limites de rotação, conexões, cabeamento, possibilidade de incompatibilidade eletromagnética, tensão de alimentação e local físico de instalação, ou seja, características físicas. Pode ser influenciado por radiação térmica ou eletromagnética e por isso informações a respeito de compatibilidades, recomendações ou possíveis incompatibilidades durante a operação devem ser armazenadas juntamente com o artefato no repositório de domínio.

Devido à complexidade que um artefato pode alcançar, (MAGA e JAZDI, 2009) sugere que a interação seja realizada através de visões. Isso significa que cada disciplina teria uma visão particular do artefato a ser criado/editado, de acordo com a Figura 2. Dessa forma,

características mecânicas de um artefato seriam omitidas ao eletricista, mas as regras e validações de disciplina de mecânica ainda estariam sendo observadas.

Comparado com componentes de *software*, que apenas interagem através de informações, os componentes de um sistema de automação além de trocar sinais, também interagem através de fluídos ou energia. Outras relações como incompatibilidades, recomendações ou alternativas entre as partes devem ser consideradas. Devido a essa complexidade, o artigo de (MAGA e JAZDI, 2009) propõe coordenar as interações entre os diferentes artefatos reutilizáveis através de uma camada separada, chamada de Camada de Conexão, que integra as informações com o ambiente de desenvolvimento DRE (*Domain Repository Engine*). A Figura 4 apresenta um esquema desse conceito.

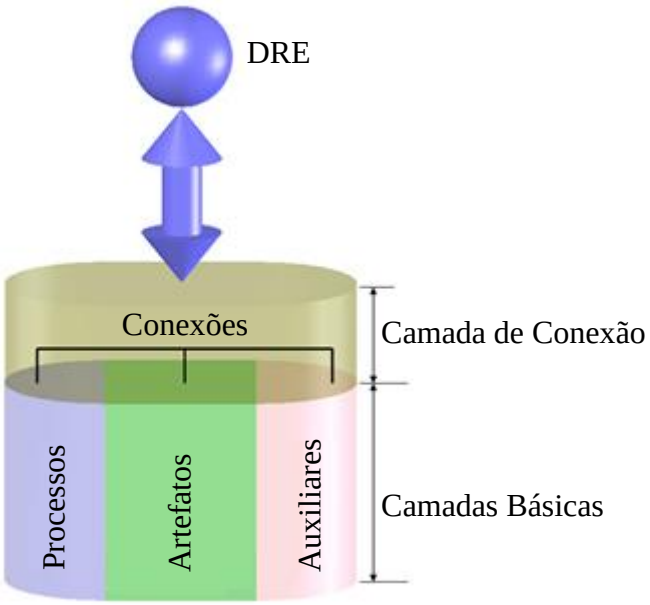


Figura 4 – Camadas de um Repositório de Domínio (MAGA e JAZDI, 2009).

Na partição Processos são armazenados os *workflows* necessários para gerenciar a criação do novo sistema de automação industrial. Na partição Artefatos, são guardados os elementos reutilizáveis como diagramas funcionais, modelos de *layout* de tubulações, desenhos de montagem e de instalação ou especificações de instrumentação e do sistema de controle. E

na partição Auxiliares estão materiais auxiliares que não necessariamente são vistos no produto final, como ferramentas de *software*, ferramentas ou máquinas especiais necessárias durante a montagem.

O trabalho desenvolvido por (MAGA e JAZDI, 2009) recomenda que sejam inseridos no repositório o próprio método de desenvolvimento da aplicação uma vez que os processos utilizados são quase sempre as mesmos (*workflow*, responsabilidades, custos, etc.).

De acordo com esse conceito, o desenvolvimento de sistemas de automação necessita de materiais auxiliares, como *softwares* CAx, bancadas de teste, documentos de padronização de processos, máquinas especiais ou ferramentas de montagem. Por isso, é um requisito adicionar esses materiais ao domínio de forma a melhorar o desenvolvimento da aplicação.

A granularidade com que os artefatos são desenvolvidos é um importante tema de estudo, uma vez que a incorreta definição do nível de detalhamento pode causar dificuldades na reutilização (JAZDI;MAGA e GOEHNER, 2011).

Uma plataforma única de desenvolvimento proporciona a integração e consistência de informações entre os artefatos de diferentes disciplinas, uma vez que a troca de informações entre objetos de disciplinas distintas é complexa (MAGA;JAZDI e GÖHNER, 2011). Apesar de existirem algumas ferramentas no mercado que são orientadas pelos conceitos acima, elas ainda pecam ao integrar artefatos abstratos (notadamente *software*) com artefatos físicos (de disciplinas mecânica e elétrica). As ferramentas disponíveis são proprietárias e incompletas, o que dificulta os estudos acadêmicos (MAGA e JAZDI, 2012).

Além do mais, a teoria de Engenharia de Domínio não engloba a modelagem de sistemas de forma abstrata e nem a simulação, como ocorre em MBSE. A proposta de utilização conjunta de Engenharia de Domínio com o MBSE procura complementar essa lacuna juntando as características positivas de cada teoria.

2.8 SISTEMAS FÍSICO-CIBERNÉTICOS

Segundo (LEE, 2008), um sistema físico-cibernético (*Cyber-Physical Systems* - CPS) é caracterizado pela integração entre computação e sistema físico e o processo físico, que enfatiza a interação e cooperação entre a informação e o meio físico.

As principais características de sistemas CPS são (BAHETI e GILL, 2011):

1. processamento computacional e processos físicos são tão integrados que não é possível identificar se atributos comportamentais são resultado de cálculos (programas de computador), leis físicas ou as duas trabalhando juntas;
2. funcionalidades e características proeminentes emergem da interação entre objetos físicos e computacionais;
3. computadores, redes, dispositivos e o ambiente no qual estão inseridos tem propriedades físicas que se inter-relacionam, consomem recursos e contribuem para o comportamento geral do sistema.

Sistemas Físico-Cibernéticos são constituídos por sistemas físicos, como componentes mecânicos, hidráulicos, elétricos, e componentes eletrônicos e de *software*. A integração entre componentes de características distintas é um dos desafios.

O estudo de CPS pretende alcançar o desenvolvimento de sistemas estáveis com propriedades e comportamentos previsíveis para automação segura mesmo em ambiente com condições instáveis. O meio para efetivar tais objetivos é a integração entre os modelos oriundos da ciência da computação, engenharia elétrica e engenharia mecânica. A modelagem do sistema é concebida de forma interdisciplinar, incorporando *software*, eletrônica e sistemas físicos.

Segundo (ACATECH, 2011), para alcançar os objetivos propostos há a necessidade de fundir os modelos criados pela ciência da computação, que lidam com informação e conhecimento, com os modelos físicos, que descrevem tempo e espaço. A modelagem e simulação de sistemas CPS ainda estão em estágio embrionário, faltando tecnologias que

suportem o desenvolvimento de modelos complexos e análise de sistemas heterogêneos. O desenvolvimento de uma linguagem que suporte a descrição de um sistema híbrido ainda é um desafio.

3 ESTADO DA ARTE

A busca por trabalhos realizados nos últimos anos teve como premissa a identificação de esforços realizados no sentido de resolver os problemas enfrentados pela engenharia de sistemas com simulação e uso de modelos. A seguir são apresentadas as iniciativas que utilizaram linguagens Modelica ou SysML (ou alguma de suas variantes) e ferramentas baseadas nos conceitos de MBSE, Engenharia de Domínio ou CPS.

3.1 MBSE COM UML/SYSML

Iniciativas recentes utilizam SysML para modelar Sistemas de Automação (FRIEDENTHAL;MOORE e STEINER, 2011; WEILKIENS, 2011). A linguagem abarca diagramas que possibilitam a especificação e documentação de requisitos, elementos estruturantes do sistema, funções, comportamentos e propriedades e é fundamental para diferenciar na prática a Engenharia Baseada em Modelos da Engenharia Centrada em Documentos. SysML também é bem conceituada como uma linguagem para expressar processos, informação e conhecimentos gerados durante o emprego de um método de desenvolvimento de sistemas (PAREDIS *et al.*, 2010).

Partindo-se do princípio de que requisitos oriundos do cliente devam ser tranformados num sistema que os atenda, a linguagem SysML auxilia a MBSE fornecendo uma gama de diagramas que permitem representar graficamente os processos, métodos e artefatos. A linguagem SysML pode ser utilizada desde a análise dos requisitos, desenho da solução, avaliação das alternativas até a verificação e validação. Portanto, a linguagem SysML adere à filosofia de MBSE fornecendo uma base sólida na qual a engenharia de sistemas se suporta, mas carece de refinamento quando o objetivo é conectar modelos abstratos a modelos físicos. Faltam métodos específicos para transformar os modelos em artefatos de *software* e *hardware*.

A SysML foi desenvolvida para proporcionar estruturas simples e funcionais para modelar ampla gama de problemas de engenharia de sistemas. É efetiva ao especificar requisitos, estruturas, comportamento, alocações e restrições em propriedades de sistemas para suportar a análise de engenharia. Apoia a especificação, análise, design, verificação e validação de sistemas que incluem *hardware*, *software*, estrutura de dados, pessoas, processos e instalações, mas não fornece documentos suficientes para produzir o sistema sob desenvolvimento.

No contexto de engenharia de *software*, diagramas UML podem ser suficientemente detalhados para criar produtos fora do modelo, enquanto que a linguagem SysML trabalha apenas num nível mais alto de abstração. No caso de sistemas de automação, os desenhos ou diagramas para a produção ficam em outras ferramentas ou ambientes de desenvolvimento como o AutoCAD®, Invertor®, E-Plan®, COMOS® (de forma geral, *softwares* CAD), que detalham a maneira pela qual o produto será realizado.

Em SysML, é possível definir as condições de verificação e o comportamento de objetos, o que se encaixa perfeitamente para componentes de sistemas do tipo *software*. Porém esse procedimento não é eficaz ao retratar um componente físico, uma vez que pode especificar o seu comportamento, mas não verificá-lo. Em SysML o comportamento de um sistema pode ser precisamente descrito, mas não é possível simulá-lo.

No caso de MBSE suportada por SysML, a interligação entre o modelo abstrato e os ambientes de desenvolvimento é usualmente feita de forma manual pelo projetista ou engenheiro, podendo ocorrer erros de interpretação ou digitação na transcrição do que foi modelado.

A linguagem SysML não define uma metodologia de modelagem, restringindo-se a uma ferramenta para analisar requisitos, estruturar e arquitetar o projeto. A especificação completa

do sistema estará distribuída em vários diagramas, dentre os quais vale ressaltar os diagramas de requisitos, diagramas de comportamento, diagramas de estrutura e diagramas paramétricos.

O diagrama de estrutura é responsável por definir a arquitetura do sistema, seus componentes e suas ligações de forma genérica. O diagrama de comportamento define como o sistema funciona, indica quais os passos para se alcançar o objetivo, e pode ser alocado a determinado componente. O diagrama de requisitos detalha as especificações que um componente ou sistema precisa atender e o diagrama paramétrico define as leis que regem o sistema. O diagrama paramétrico é interligado ao diagrama de estrutura de forma a obter informações que possibilitem a verificação do sistema em termos do atendimento dos requisitos. A relação entre os diagramas SysML está representada na Figura 5.

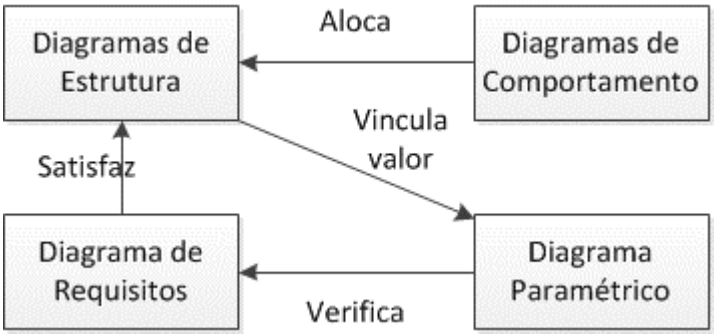


Figura 5 – relação semântica entre diagramas SysML.

Os blocos SysML não necessariamente representam a visão de um componente ou a estrutura do produto, mas sim uma abstração do sistema especificado. O controle da configuração de estruturas hierárquicas normalmente não é suportado porque SysML foca na definição do comportamento do sistema e não da arquitetura que será empregada nem da estruturação hierárquica entre componentes.

Segundo (FRITZSON, 2004), nem toda a informação relevante para a construção de um sistema é passível de ser modelada. Elementos que detalham o processo de fabricação de um objeto físico, não são, via de regra, componentes que figuram em modelos. Isso significa

que os documentos tradicionais, que detalham elementos para a fabricação do sistema, devem continuar existindo. A Figura 6 apresenta parte do esquema de ligação de um motor onde podem ser vistos detalhes quanto a caracteríticas e ligações físicas dos condutores elétricos que não são usualmente modelados.

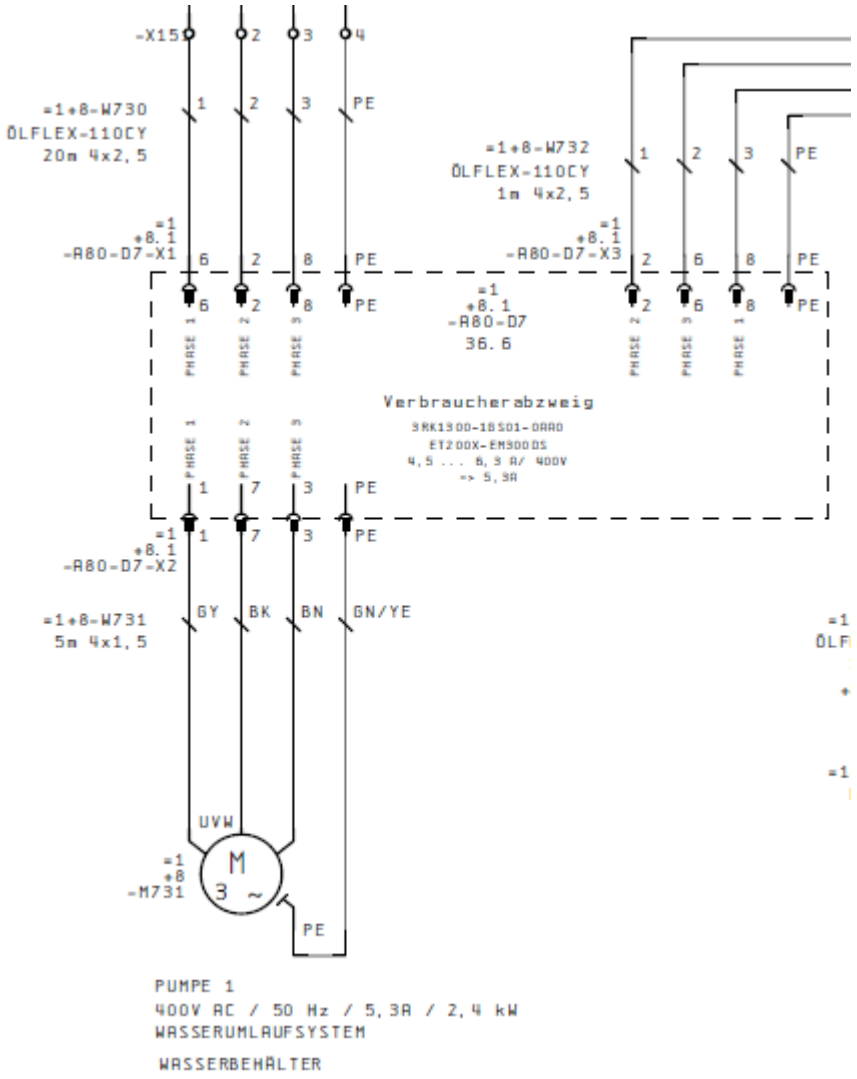


Figura 6 – Exemplo de esquema elétrico usado para fabricação de Sistemas de Automação.

O nível de utilização de SysML no processo de engenharia de sistemas é caracterizado por (FRIEDENTHAL;MOORE e STEINER, 2011) da seguinte forma:

- Custo mínimo: uso de notações SysML em apresentações e estruturação do projeto em seu início, apenas para análise e discussão da arquitetura;

- MBSE híbrido: uso de SysML para modelar elementos-chave de um sistema mas predomina o uso de ferramentas centradas em documentação;
- MBSE completo: adoção completa de SysML em todos os passos do desenvolvimento do sistema.
- Alternativos: usam ferramentas de engenharia de sistemas com notação proprietária.

3.2 ENGENHARIA DE DOMÍNIO COM COMOS®

Dentro do portfólio de soluções para *Product Lifecycle Management* (PLM) da Siemens®, a empresa conta com um pacote de gestão de ativos chamado COMOS®. Essa ferramenta destina-se a acompanhar as fases de projeto, montagem, testes, operação e manutenção de plantas industriais. COMOS® suporta a interligação entre as disciplinas de engenharia de processos, desenho conceitual, projeto de tubulações, desenho funcional, planejamento de automação e controle, engenharia elétrica, instrumentação e controle. Também são suportados o gerenciamento de ativos, gestão de manutenção e de documentos (SIEMENS, 2012b).

Um projeto do COMOS® é caracterizado por coincidir com o projeto de engenharia sendo desenvolvido em termos de componentes. A esse projeto podem ser atribuídas *working layers*, que proporcionam ao projetista um ambiente de trabalho baseado numa cópia física do projeto sem alterar a base de dados original. Uma vez que o usuário tem confiança de que suas alterações estão corretas, pode-se aplicá-las à base de dados principal do projeto.

Um objeto da plataforma COMOS® é a unidade básica para encapsulamento de informações (SIEMENS, 2012a), que pode ter propriedades e funções. Os objetos-base são

elementos estruturados que representam um componente de *software* ou *hardware* em termos de suas propriedades, conexões, funções e comportamento.

Objetos de engenharia são os objetos-base modificados pelo engenheiro de sistemas. Os objetos-base são instanciados e parametrizados de acordo com a função que devem realizar. Nesse sentido, o COMOS® funciona com o conceito de herança, ou seja, se o objeto base for modificado de alguma forma, todos os objetos-filho serão atualizados com as novas mudanças.

A Figura 7 apresenta algumas propriedades de um objeto “bomba centrífuga” chamado no projeto de mainPump. Quando o objeto mainPump é inserido no projeto, as propriedades do objeto-base “bomba centrífuga” são herdadas (como por exemplo diâmetro nominal, pressão nominal, tipo de flange). Além disso, o COMOS® fornece a possibilidade de estender características dos objetos-base através da inserção de objetos, como conexões ou propriedades (caso fosse necessário inserir a cor da bomba, no exemplo da Figura 7). Objetos-base personalizados para um determinado domínio podem ser reutilizados.

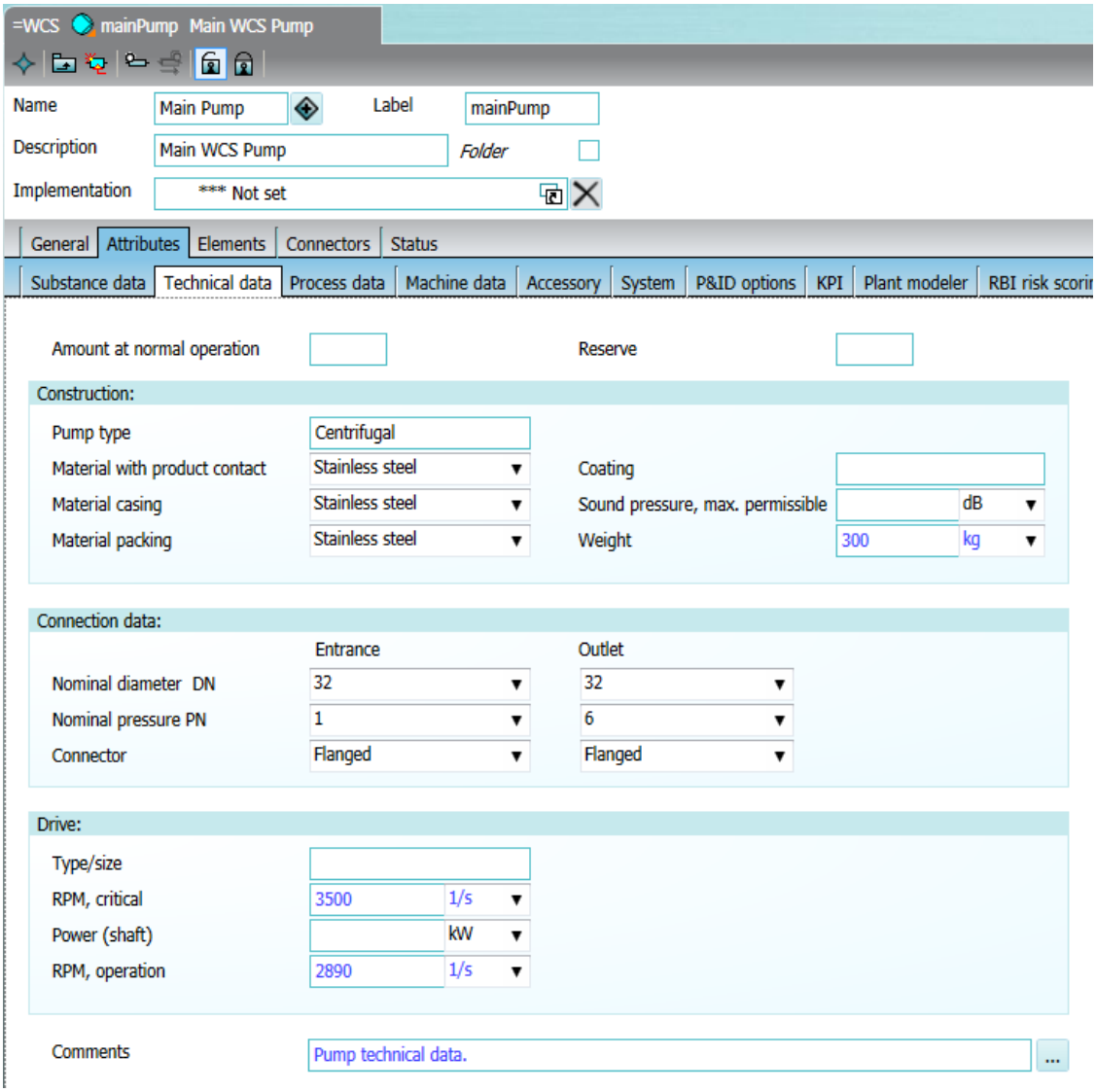


Figura 7 – Propriedades de um objeto COMOS®.

A Figura 8 demonstra o uso de conectores para interligar objetos. No caso apresentado, o conector de saída de uma bomba centrífuga (returnPump:O1) está interligado com o conector de entrada de uma tubulação (pipeMain:I0). Se o diâmetro da saída da bomba *returnPump* for alterado, o *software* alterará o diâmetro do tubo *pipeMain* e tratará eventuais conflitos com outros objetos automaticamente.

No contexto da plataforma COMOS®, símbolos pretendem representar um objeto qualquer num relatório. Um objeto pode ter tantos símbolos quantos forem necessários. Isso proporciona uma flexibilidade interessante quando se trabalha com múltiplas disciplinas. Dessa

forma, um disjuntor eletromagnético pode ser representado pelo símbolo elétrico bem como por seu desenho mecânico, dependendo em qual relatório ele deverá ser apresentado. Mudanças em suas propriedades são automaticamente atualizadas nos relatórios que o contém. Esses símbolos são estabelecidos no objeto-base, ou seja, na classe do objeto dentro do COMOS® e as instâncias herdam os modos de representação. O que fará um objeto ser representado de determinada forma será o contexto no qual ele é mostrado.

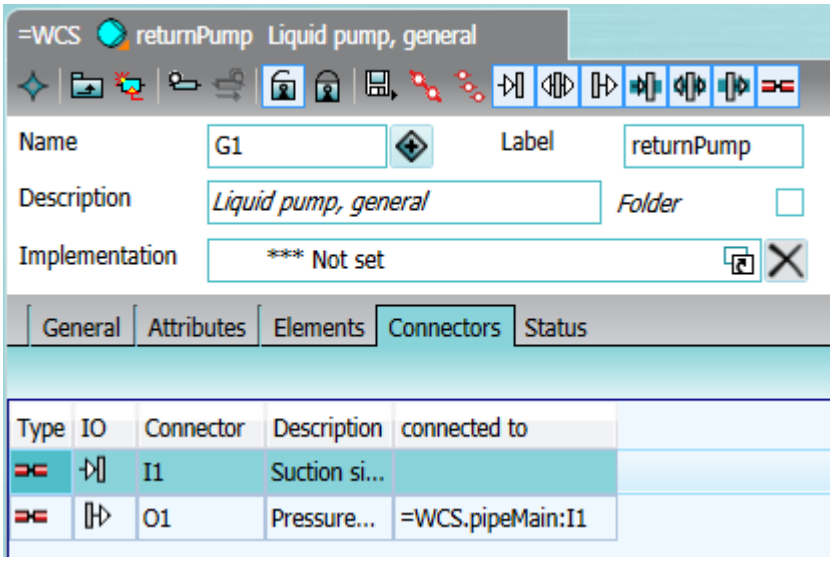


Figura 8 – Conectores em COMOS®.

Atributos de um objeto podem ser ligados a atributos de outro (SIEMENS, 2012c). Essa característica do COMOS® é vantajosa quando o projeto contém tubulações ou sistemas elétricos, já que seus objetos apresentam a característica natural de serem fisicamente conectados. A disciplina na qual os objetos interligados pertencem, contudo, deve ser a mesma. Isso significa que um objeto-base de instrumentação não pode ser ligado a um objeto-base de tubulação, por exemplo. A disciplina de mecânica não é suportada por essa ferramenta. Apenas é possível armazenar desenhos mecânicos estáticos, sendo essa uma importante limitação do *software*.

Um objeto possui nome, etiqueta e texto, que o identifica dentro do contexto do projeto. Os objetos podem ser classificados e estruturados por unidade do projeto ou por local. COMOS® fornece suporte à gestão de documentos. Cada documento armazenado dentro da estrutura de um projeto é tratado como um objeto. Essa característica pode ser usada por métodos baseados em MBSE para centralizar o armazenamento de modelos gerados durante o processo de engenharia, sob forma de documentos.

COMOS® possui um módulo que trata dos relatórios do projeto, que são divididos da seguinte forma:

- Relatórios de avaliação: a ferramenta é capaz de gerar relatórios de avaliação do projeto que servirão para realizar gestão sobre a aquisição de materiais ou fabricação, como por exemplo, lista de peças e diagrama de terminais;
- Relatórios interativos: são relatórios que contém informações semelhantes às contidas em relatórios de avaliação, porém nessas páginas é possível realizar modificações dinâmicas e inclusive arrastar e soltar novos objetos;
- Relatórios tipo “desenho”: esses relatórios são semelhantes aos desenhos extraídos a partir de ferramentas CAD tradicionais, como diagramas de circuitos elétricos, esquemas de montagem e diagramas de tubulação e instrumentação (SIEMENS, 2012c).

O COMOS® está alinhado com os conceitos de repositório de domínio (MAGA;JAZDI e GÖHNER, 2011), já que a ferramenta é baseada em orientação a objetos, é possível controlar o acesso ao conteúdo do repositório e possibilita meios de realizar gestão de mudanças dentro do banco de dados.

3.3 SISTEMAS FÍSICO-CIBERNÉTICOS COM MODELICA

Modelica é uma linguagem para modelagem de sistemas físicos multidisciplinares, representando sistemas dinâmicos através de uma abordagem orientada a objetos. Comportamentos estáticos também são passíveis de representação, porém as ferramentas disponíveis possuem restrições no que diz respeito à sua simulação, uma vez que podem levar o compilador a erros (HÖGER, 2012). Esses erros são comumente gerados por divisões por zero em tempo de simulação que não são tratados pelo compilador. Modelica.org é uma associação aberta, sem fins lucrativos, que foi estabelecida em 2000 e desenvolve a biblioteca Modelica tanto para uso comercial como não-comercial sob as condições do *The Modelica License*.

Plataformas baseadas nessa linguagem fornecem suporte à modelagem de sistemas híbridos (envolvendo distintas disciplinas) discretos ou contínuos através de mecanismos de sincronização de dados. Fornecem interface bidirecional (para fluxo de energia) e interface direcional (para fluxo de sinais) para conectar diferentes objetos. Modelica fornece a linguagem básica para especificação e interligação de modelos de sistemas multidisciplinares e de múltiplos domínios, vide Figura 9. Além de programação textual, algumas ferramentas fornecem o suporte visual para interligação de classes de distintas disciplinas, sendo esse um de seus principais pontos fortes (FRITZSON, 2004; 2011).

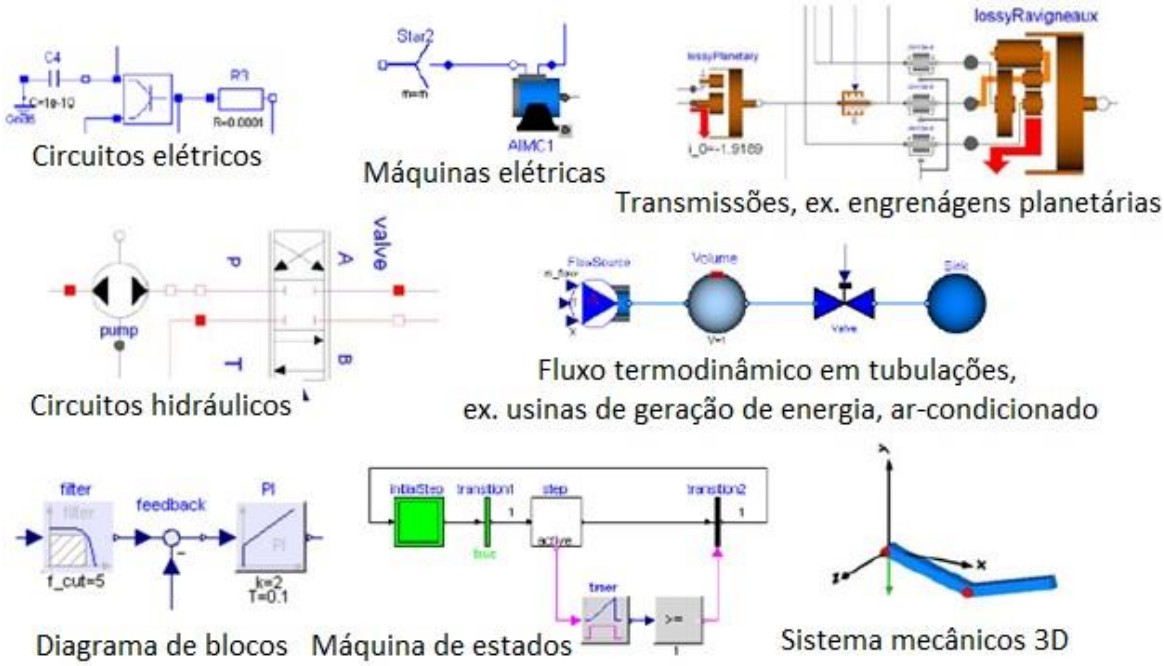
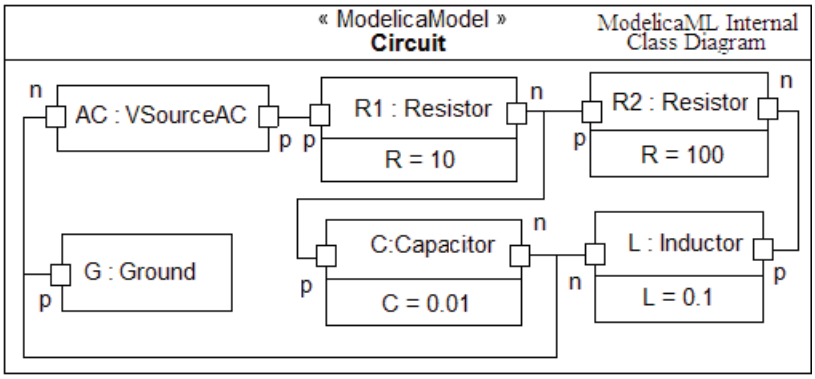


Figura 9 – Componentes multidisciplinares de Modelica (MODELICA ASSOCIATION, 2012).

Modelica possibilita modelar o comportamento de componentes de sistemas físicos de forma não-causal. Linguagens de programação convencionais não admitem o uso de equações mas sim um mecanismo de atribuição, que não é bidirecional. No caso de Modelica, a solução do modelo não é limitada a uma única direção e é baseada na atribuição de variáveis a partir de um contexto ao invés de serem estabelecidas pelo usuário. Essa característica aumenta a capacidade de objetos Modelica serem reutilizados em diferentes projetos, além de proporcionar uma ferramenta de especificação de alto nível.

Algumas ferramentas baseadas em Modelica oferecem interface gráfica que facilita a modelagem. Uma comparação entre os principais compiladores disponíveis no mercado, Dymola, OpenModelica e JModelica, pode ser localizada em (FRENKEL *et al.*, 2011). Há também uma plataforma sendo desenvolvida para integrar Modelica com ferramentas de modelagem de alto nível em ambiente Eclipse, que se chama Modelica IDE (SAMLAUS *et al.*, 2011).

Os objetos ou componentes do sistema são identificados e posicionados de forma que suas ligações representem a relação física entre eles. Essas ligações são chamadas de conectores ou portas e descrevem as possibilidades de interação entre os componentes. As portas são instâncias da classe *connector* e podem representar variáveis de fluxo (corrente elétrica, por exemplo) ou de potencial/nível de energia (tensão elétrica, por exemplo). A linguagem descritiva do Modelica é mapeada para equações diferenciais, algébricas e discretas que descrevem matematicamente o sistema modelado. A Figura 10 apresenta um exemplo de circuito no qual os componentes conectados visualmente foram mapeados para código Modelica.




```
model Circuit
  Resistor  R1(R=10);
  Capacitor C(C=0.01);
  Resistor  R2(R=100);
  Inductor  L(L=0.1);
  VSourceAC AC;
  Ground    G;
equation
  connect (AC.p, R1.p);
  connect (R1.n, C.p);
  connect (C.n, AC.n);
  connect (R1.n, R2.p);
  connect (R2.n, L.p);
  connect (L.n,  C.n);
  connect (AC.n, G.p);
end Circuit;
```

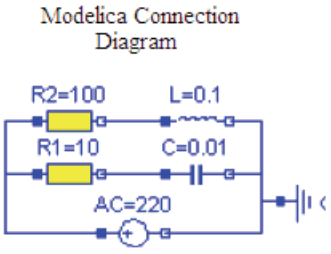


Figura 10 – Comparação entre ModelicaML *Internal Class Definition* e Modelica *Connection Diagram* (POP;BĂLUȚĂ e FRITZSON, 2007).

Além de classes (que são os modelos de simulação e representam componentes físicos), a especificação do Modelica permite a definição de funções (para criar algoritmos), pacotes (usados para estruturar o modelo) e modelos parciais (usados como modelos-base ou *templates*). Na modelagem visual, as linhas que interligam os componentes representam suas conexões físicas, por exemplo um cabo elétrico, uma conexão mecânica ou fluxo de calor. As variáveis na interface descrevem a interação com outros componentes e o comportamento físico dos objetos é descrito por equações. Portanto, existe uma decomposição hierárquica dos componentes que constituem o sistema, os quais são interligados de acordo com sua interação física. Nesse sentido, a biblioteca padrão do Modelica fornece objetos multidisciplinares que podem ser integrados num mesmo modelo. O modelo apresentado na Figura 11 representa virtualmente um motor DC com rampa de aceleração que aciona uma carga. Os parâmetros desse sistema podem ser alterados até que o comportamento simulado esteja de acordo com o desejado.

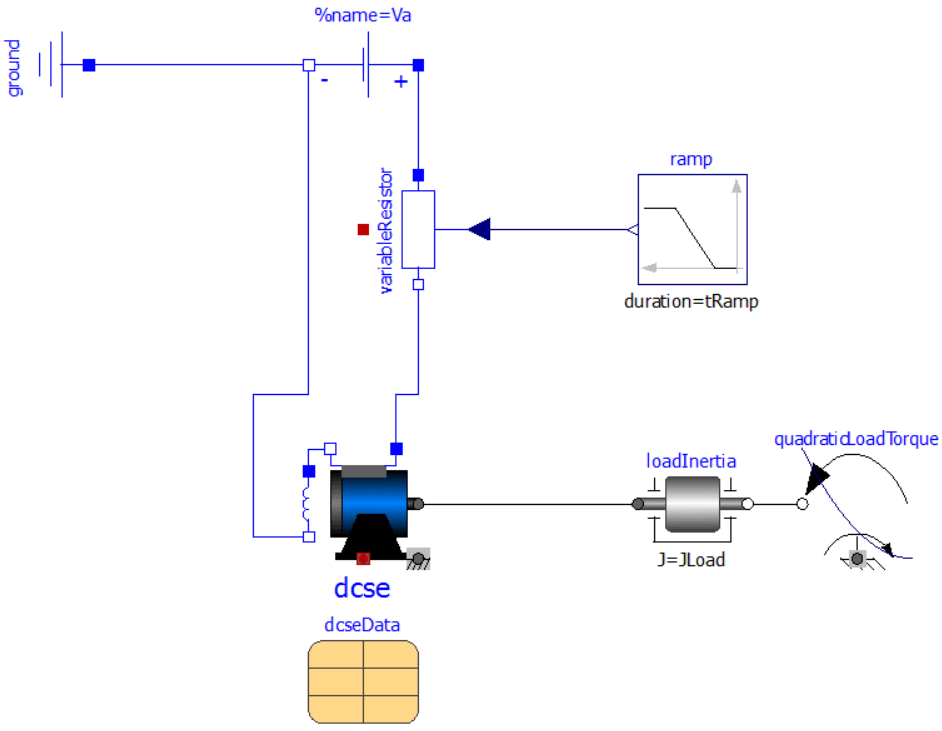


Figura 11 – Sistemas com múltiplas disciplinas modelados com Modelica (MODELICA ASSOCIATION, 2013).

A possibilidade de simulação do modelo numa única plataforma é uma vantagem da linguagem. A simulação é importante porque possibilita um melhor entendimento de sistemas complexos, otimização e restrição das propriedades dos objetos (JARDIN *et al.*, 2011). Proporciona assim uma plataforma de prototipagem virtual e de verificação dos requisitos do sistema, complementando a linguagem SysML nesse aspecto.

A linguagem Modelica se caracteriza pela orientação a objetos no que diz respeito à estruturação da linguagem. É possível decompor o sistema incialmente em suas partes principais e refinar o modelo à medida que essa necessidade se impõe. Essa flexibilidade confere à linguagem a capacidade de decompor recursivamente os modelos de alta complexidade em componentes simples. Durante o processo de modelagem, a relação topológica de partes físicas e a relação lógica de partes computacionais (de componentes de *software*) precisa ser mantida, já que é uma linguagem estruturada.

Por outro lado, em linguagens de modelagem orientadas à diagrama de blocos, como exemplo pode-se citar o Simulink/Matlab, a decomposição da topologia do sistema não corresponde à estrutura natural do sistema físico e sim à sua representação através de sinais. É necessário um trabalho intenso de conversão de equações que representam o sistema físico em representações correspondentes em fluxo de sinal. Isso se torna uma desvantagem quando se trabalha naturalmente com objetos, por exemplo, em sistemas mecânicos ou elétricos como da Figura 11. Modelos físicos são difíceis de interpretar em representações que usam modelos de sinais. Pequenas mudanças no modelo físico repercutem em completo redesenho do diagrama de blocos, dificultando o reuso. A vantagem dos diagramas de bloco fica por conta da representação de sistema de controle, que é mais intuitiva sob a forma de sinais.

A herança e polimorfismo são suportados pela especificação Modelica, mesmo que essa última ainda seja parcialmente desenvolvida (BROMAN;FRITZSON e FURIC, 2006).

Para tornar eficientes os modelos desenvolvidos com o uso de Modelica, os modelos físicos devem ser simplificados quando as partes se tornam muito complexas, quando o tempo de processamento durante a simulação se torna extenso, quando há instabilidades numéricas ou dificuldades em interpretar resultados devido à existência de muitos detalhes de baixo nível (FRITZSON, 2004). O autor ainda sugere negligenciar efeitos que não são importantes para o fenômeno sendo modelado e simplificar subsistemas com comportamento dinâmico próximo do estável. Modelos de conexões hidráulicas cuja influência no comportamento do sistema é desprezível são exemplos de componentes que podem ser omitidos durante a simulação.

Em seus trabalhos, (HENRIKSSON e ELMQVIST, 2011) e (TANG *et al.*, 2012) demonstram a utilização de Modelica para auxiliar na modelagem de sistemas CPS. Portanto, apesar de CPS ser um conceito relativamente recente (WOLF, 2009) percebe-se que suas características são desejadas em sistemas embarcados e de automação e são condizentes com as necessidades do presente trabalho.

Modelica é usado em diversos domínios com o intuito de modelar e simular sistemas físicos. O trabalho realizado por (STROBEL *et al.*, 2011) analisa o uso da linguagem para a modelagem e simulação de turbinas eólicas, apresentando os primeiros resultados de uma biblioteca específica criada pelo grupo para tal. Já o trabalho de (HOU *et al.*, 2011) fornece um exemplo de modelagem multidisciplinar que investiga um sistema de absorção de impactos. Outro domínio cuja aplicação do Modelica como plataforma de modelagem e simulação foi testada diz respeito ao setor de Petróleo e Gás (GUO;QIN e GERHARD, 2011), onde o comportamento de um motor de combustão interna é simulado a partir do uso de diferentes misturas de gás natural, GLP e biogás para alimentá-lo. A aplicação da linguagem em engenharia civil para simulação de transferência de calor em quartos foi investigada por (WETTER;ZUO e NOUIDUI, 2013) trazendo resultados do consumo de energia para diferentes

tipos de construções durante fases distintas do ano. Um modelo e simulação de sistemas biológicos baseados em redes de Petri também foi testado por (PROß *et al.*, 2012)

O trabalho apresentado por (BOUSKELA *et al.*, 2011) investiga a aplicação da linguagem Modelica no mapeamento de incertezas usando modelos físicos. Valendo-se da natural aptidão da linguagem para a matemática, os autores desenvolveram uma proposta de blocos que calculam a incerteza a partir de medições efetuadas no ambiente no qual o sistema está inserido.

Os esforços de (MARTIN-VILLALBA;URQUIA e DORMIDO, 2013) procuram provar a aptidão da Modelica para a prática do ensino a partir de um modelo que simula uma caldeira industrial. Essa solução foi utilizada pelo grupo como laboratório virtual para ensinar a teoria de controle. Proposta similar foi realizada por (ZHENGYIN;SHENGLIN e SHAN-AN, 2011) ao desenvolver um laboratório virtual baseado em *internet* para alunos de engenharia elétrica exercitarem a teoria de circuitos e eletrônica digital e analógica.

A capacidade de rápida prototipagem da linguagem Modelica é estendida pela iniciativa de (ELSHEIKH *et al.*, 2013) no sentido de integrar esta plataforma a outras plataformas de simulação através de FMI (*Funtional Mockup Interface*), uma funcionalidade nativa da Modelica. Essa ferramenta exporta descrições das entradas, saídas e unidades físicas para unidades chamadas FMU (*Functional Mockup Units*) que podem ser importadas por outras ferramentas de simulação. FMI também foi testado por (BENJELLOUN-TOUIMI *et al.*, 2011) no contexto de desenvolvimento de controles para motores de combustão de automóveis. O trabalho objetivou integrar o ambiente de simulação Modelica com outras plataformas de simulação já desenvolvidas para o ramo automobilístico. O mesmo foi realizado por (SUN;VOGEL e STEUER, 2011) que constatou que a integração entre diferentes ferramentas de simulação pode ser prejudicada pela falta de clareza em erros eventualmente gerados pela transcrição do modelo.

A iniciativa de (SONNTAG e HÜFNER, 2011) propõe um algoritmo para transformar modelos de sistemas híbridos descritos em CIF (*Compositional Interchange Format*) para a linguagem Modelica. Durante o desenvolvimento do projeto, o autor utilizou a ferramenta gPROM para modelar o comportamento de um PLC (*Program Logic Controller*). Os modelos gerados foram transformados para CIF e depois transformados para um conjunto de equações booleanas na linguagem Modelica. O objetivo do trabalho foi integrar Modelica com outras ferramentas baseadas em modelo.

A partir da utilização de Modelica em conjunto com MBSE numa plataforma baseada em Windows® na empresa SAAB Aeronautics, (LIND e ANDERSSON, 2011) concluíram que Modelica carece de simplificação quando lida com simulação de sistemas complexos, uma vez que há um esforço computacional considerável para resolver os modelos. O trabalho conclui que há a necessidade de suportar computação distribuída. Também a falta ou carência de documentação foi apontada como um entrave para a utilização da linguagem em áreas cuja exigência relacionada à rastreabilidade é alta.

3.3.1 ModelicaML

A linguagem ModelicaML é uma especialização da UML e traz grande parte dos avanços realizados em SysML. Essa linguagem oferece componentes visuais para modelar os construtores utilizados na linguagem Modelica. Nasceu com o objetivo de unificar os pontos fortes das linguagens Modelica e SysML para criar uma linguagem mais expressiva e formal para MBSE. Se, por um lado, objetiva fornecer à linguagem SysML meios de detalhar o comportamento de objetos, por outro se propõe a disponibilizar ao ambiente Modelica ferramentas de projeto, como modelagem de requisitos e diagrama de herança oriundos de SysML (SCHAMAI *et al.*, 2009).

A estrutura da ModelicaML pode ser verificada na Figura 12. Os seguintes diagramas foram criados ou modificados a partir da especificação SysML:

- Diagrama de Equações: esse diagrama foi inserido para modelar o comportamento de um objeto usando equações matemáticas, fundamento da linguagem Modelica;
- Diagrama de Classes: Foi modificado a partir do *Block Definition Diagram* definido por SysML para modelar apenas as classes suportadas por Modelica, que são ModelicaClass, ModelicaModel, ModelicaBlock, ModelicaConnector, ModelicaFuntion e ModelicaRecord;
- Diagrama Interno de Classes: modela a estrutura interna da classe em termos de partes e de conectores;
- Diagrama de Pacotes: foi modificado para possibilitar o agrupamento lógico e estruturado de elementos definidos pelo usuário;
- Diagrama de Simulação: usado para modelar, configurar e documentar os parâmetros e resultados das simulações necessárias para validar os requisitos.

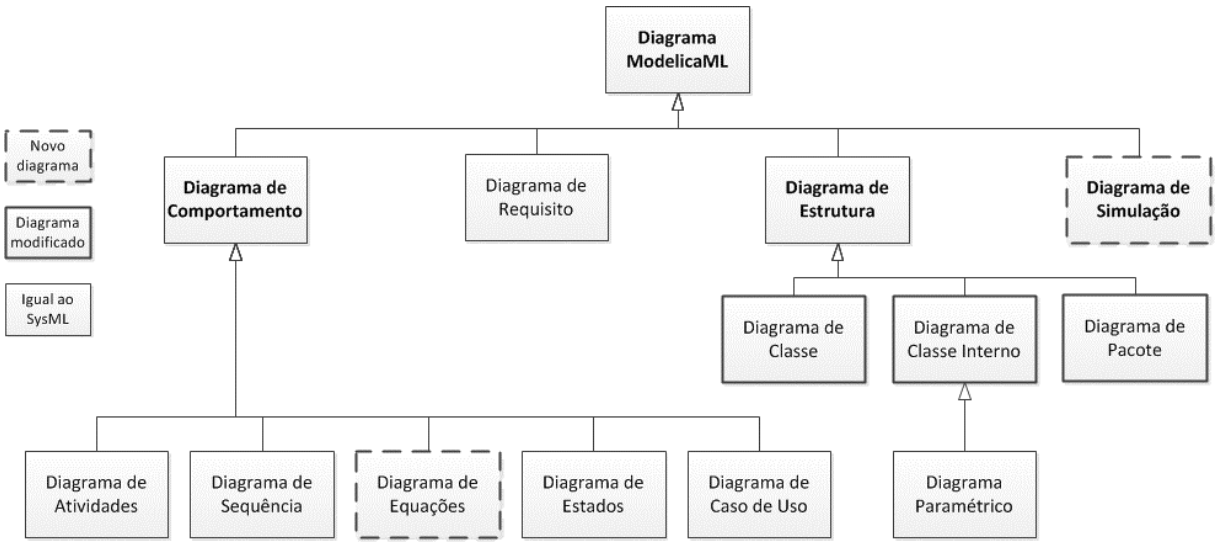


Figura 12 – Estrutura de diagramas ModelicaML (POP;BĂLUŢĂ e FRITZSON, 2007).

O diagrama de simulação é um importante avanço dessa linguagem, pois proporciona consistência dos dados simulados, rastreabilidade dos testes executados e armazenamento de dados para posterior validação ou conferência.

ModelicaML fornece suporte à tradução entre modelos Modelica e SysML através de XMI (*XML Metadata Interface*). A implementação de ModelicaML sobre Eclipse desenvolvida a partir do Plug-in Papyrus possibilita desenvolver diagramas UML e SysML (HAKAM, 2011) e transformar os modelos em código Modelica para simulação.

O uso de ModelicaML possibilita modelar diversas fases do processo de desenvolvimento de um sistema, como a análise de requisitos, a concepção da solução, a implementação, verificação, validação e integração. O comportamento do sistema é modelado através de equações ou algoritmos. ModelicaML serve como interligação entre SysML e Modelica uma vez que proporciona aos usuários de UML/SysML um melhor entendimento da linguagem Modelica através de construtores (classes, equações e variáveis) bem como uma ferramenta que traz benefícios (principalmente através da simulação) para uma melhor engenharia de sistemas. Também estende a capacidade de documentação das abordagens citadas uma vez que estabelece meios de especificar de forma mais precisa o comportamento de objetos e de estruturar e guardar resultados de simulações.

3.4 QUADRO COMPARATIVO

Observa-se que o Projeto Centrado em Documento não fornece as vantagens da Orientação a Objetos e nem suporte à Engenharia de Requisitos, sendo que o reuso de componentes é realizado manualmente. MBSE fornece ferramentas em nível abstrato de modelagem trazendo as vantagens da Orientação a Objetos, porém a modelagem de sistemas físicos e a integração entre objetos multidisciplinares é limitada. A DE é um conceito baseado em Orientação a Objetos e ligado à realização dos componentes, não fornecendo as ferramentas de modelagem abstrata que o MBSE fornece. Já os recentes avanços realizados por CPS e

Modelica trazem benefícios no que diz respeito à interligação entre objetos multidisciplinares, reuso de componentes e a oferta de uma plataforma de simulação. A Tabela 1 se propõe a resumir as principais características no contexto de Sistemas de Automação das abordagens analisadas.

Tabela 1. Comparação entre diferentes perspectivas de desenvolvimento de Sistemas.

	Centrada em documento	MBSE	Engenharia de Domínio	CPS
Paradigma de Orientação a Objetos	Não	Sim	Sim	Sim
Reuso de componentes	Realizada manualmente	Sim	Sim	Traz benefícios
Integração multidisciplinar	Realizada através de *TAG*s	Parcial (nível abstrato)	Parcial	Sim
Suporte à engenharia de requisitos	Não	Sim	De forma superficial	Contribui
Suporte à modelagem de sistemas físicos	Sim	Em partes	Sim	Sim

As características das ferramentas analisadas podem ser resumidas de acordo com a Tabela 2. As ferramentas CAD, CAE e CAM ou E-CAD não fornecem suporte à orientação a objetos. O SysML não possui mecanismos para conectar componentes distintos e trocar informações entre eles. Percebe-se que os métodos ou ferramentas estudadas não cobrem por completo as necessidades de projetos de Sistemas de Automação e acredita-se que a sua utilização integrada necessita ser arquitetada e testada.

Tabela 2. Comparação entre diferentes ferramentas que suportam o desenvolvimento de Sistemas.

	CAx	SysML	COMOS®	Modelica
Modelagem Orientada a Objetos	Não	Sim	Sim	Sim
Orientação a conexão	Não	Não	Sim	Sim
Mecanismo de herança	Não	Sim	Sim	Sim
Mecanismo de polimorfismo	Não	Sim	Em partes	Sim
Suporte à modelagem de sistemas físicos	Sim	Em partes	Sim	Sim

4 PROPOSTA

A partir da discussão estabelecida nos capítulos anteriores, conclui-se que a utilização isolada de ferramenta oriunda apenas de uma das abordagens de projeto apresentadas acarreta em lacunas no desenvolvimento de sistemas de automação.

Antes de avançar no detalhamento da proposta do trabalho, é necessário especificar o funcionamento da engenharia centrada em documentos da GEIT.

4.1 PROJETO CENTRADO EM DOCUMENTOS DA GEIT

Os projetos desenvolvidos pela área de engenharia da GEIT passam por uma análise de requisitos que se baseia em experiências adquiridas em desenvolvimentos passados. A área de produção tem influência importante nos requisitos de projeto porque carrega a experiência da realização de projetos anteriores. Participam também as áreas de *marketing*, vendas e planejamento de produção. Os dados colhidos pelo gerente do projeto são armazenados em documentos Word® e Excel® e arquivados para servir de referência.

A área de engenharia é constituída de 3 equipes distintas: projeto mecânico, projeto elétrico e *software* de PLC. Cada uma trabalha de forma independente e com ferramentas específicas, da família de *softwares* CAx (*Computer Aided*). A equipe de projeto mecânico utiliza a solução da Autodesk® chamada Inventor®. Os desenvolvedores da disciplina de engenharia elétrica utilizam a ferramenta chamada DDS-C® fornecida pela empresa CoCreate® e os desenvolvedores de *software* PLC utilizam-se do pacote Step 7® da Siemens®. O produto final da área de engenharia é um pacote de documentos que são entregues à área de produção para que o sistema seja fabricado. Para efeitos de simplificação, o código de PLC apto a ser transferido para o controlador também foi considerado uma saída desse processo, ou seja, documentação. Um desenho explicativo do que foi exposto pode ser visto na Figura 13.

Os novos documentos gerados são armazenados na mesma estrutura de arquivamento de projetos anteriores. A localização de dados nos projetos antigos é limitada à busca textual, realizada principalmente através do número do projeto. O reuso é efetivado através da cópia simples de documentos de projetos anteriores, que são editados manualmente para se adequar aos novos desafios do projeto sendo desenvolvido.

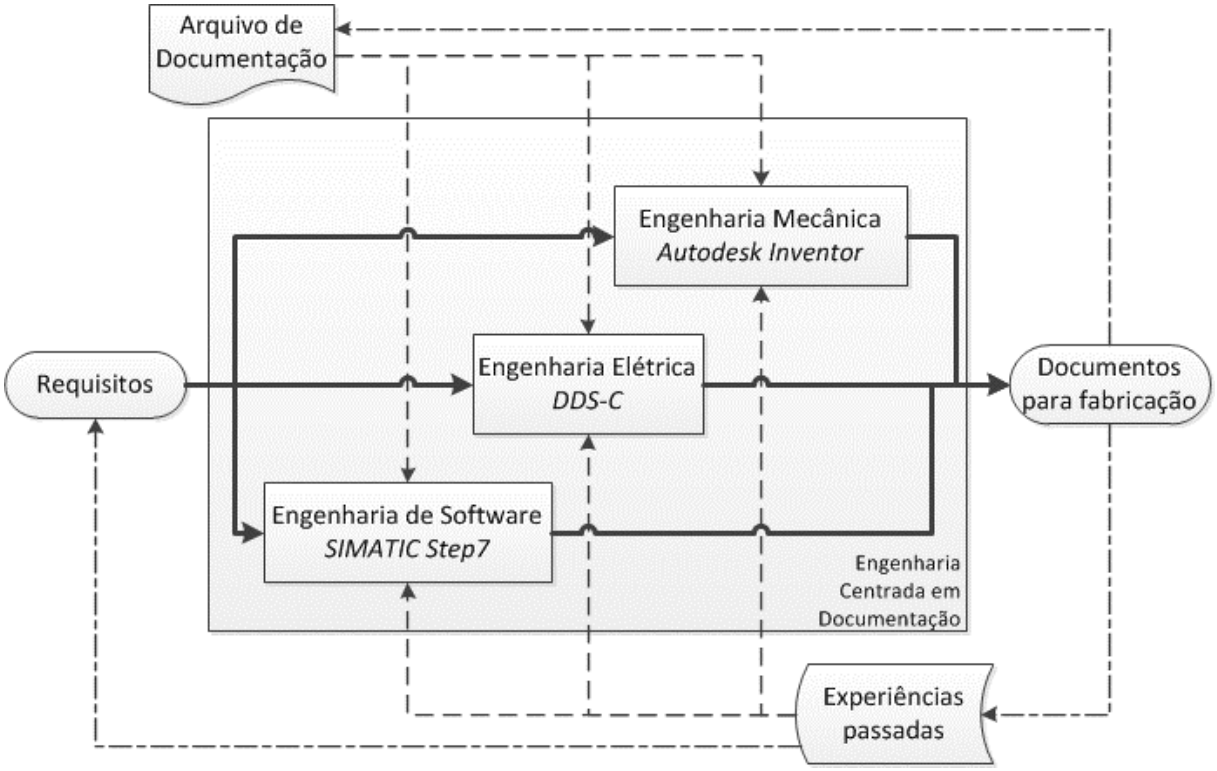


Figura 13 – Processo de engenharia centrado em documento.

Para realizar a comparação entre o método utilizado pela empresa e o método proposto, partiu-se do pressuposto que o novo método deva suportar a geração de saída semelhante ao que o método atual fornece, partindo dos mesmos requisitos. Para criar a plataforma MBSE, foi necessário localizar métodos e ferramentas que suportassem a geração de relatórios para fabricação do sistema.

Ao longo da pesquisa não foi localizado um método de engenharia baseado em modelos único capaz de fornecer uma saída em nível semelhante ao que fornece a engenharia centrada

em documentos, ou seja, documentos e relatórios capazes de proporcionar ao fabricante informações suficientes para construir o equipamento de automação. Por isso, propõe-se o desenvolvimento de um método de engenharia baseada em modelos que parta da definição dos requisitos, seja capaz de simular partes importantes do projeto e esteja conectada com um ambiente de engenharia de domínio de onde se possa extrair informações para a produção do sistema sob a forma de relatórios.

4.2 PROPOSTA DE MÉTODO DE ENGENHARIA DE SISTEMAS

O método de engenharia de Sistemas de Automação desenvolvido se denomina MoDElS.iDEAS (*Model-Driven Engineering and Simulation integrated with Domain Engineering for Automation Systems*). Caracteriza-se por apresentar uma visão multidisciplinar de componentes cujo sistema ao qual pertencem é desenvolvido a partir da modelagem abstrata. O detalhamento dos componentes se dá através da simulação de comportamentos e os seus parâmetros são integrados na ferramenta de Engenharia de Domínio, onde os artefatos são especificados e os dados para a produção são extraídos.

Partindo-se das premissas levantadas por (KOSSIAKOFF *et al.*, 2011; WEILKIENS, 2011) sobre engenharia de sistemas e se baseando no processo descrito na Figura 1, procurou-se reunir métodos e ferramentas que, em conjunto, atingissem de forma eficiente o objetivo proposto.

Com a integração da plataforma MBSE centralizada em Eclipse, que utiliza a linguagem ModelicaML, com a linguagem de simulação matemática OpenModelica pretende-se reconhecer erros de concepção na fase de projeto (HASKINS e FORSBERG, 2011). Para que seja viável a geração de documentos e relatórios que tornem a fabricação do sistema viável a partir de uma plataforma MBSE, utiliza-se a ferramenta de Engenharia de Domínio chamada COMOS®.

No que diz respeito à Engenharia de Domínio, os esforços foram concentrados em testar o método proposto por (KIPPER, 2010) e para tal o *software* COMOS®, desenvolvido e comercializado pela Siemens®, foi mantido como ferramenta. Avaliou-se a sua capacidade de trabalhar em conjunto com ferramentas MBSE.

Um aplicativo escrito em Java foi necessário para conectar os modelos abstratos (MBSE) às representações abstratas de objetos físicos e lógicos – chamados de artefatos dentro da proposta de Engenharia de Domínio. A conexão das ferramentas baseadas em modelos à camada de realização do projeto foi efetuada a partir do mapeamento de propriedades e funções dos objetos e artefatos. Propriedades de objetos abstratos da ferramenta MBSE foram mapeados para propriedades dos artefatos da ferramenta de Engenharia de Domínio. A Figura 14 mostra essa troca de informações. A representação abstrata de um objeto qualquer descrita nas linguagens SysML/ModelicaML tem suas propriedades mapeadas às representações dos artefatos físicos e lógicos na ferramenta de Engenharia de Domínio. O comportamento dos artefatos no COMOS® é restrito em função de parâmetros, descrições físicas e funções. Em contrapartida, o COMOS® fornece à plataforma MBSE as limitações do artefato escolhido para desempenhar determinada função. Essa troca de dados é importante uma vez que proporciona ao ambiente de simulação OpenModelica informações mais robustas e acaba contribuindo para que as análises de requisitos e simulações ocorram de forma mais condizente com o funcionamento real do artefato.

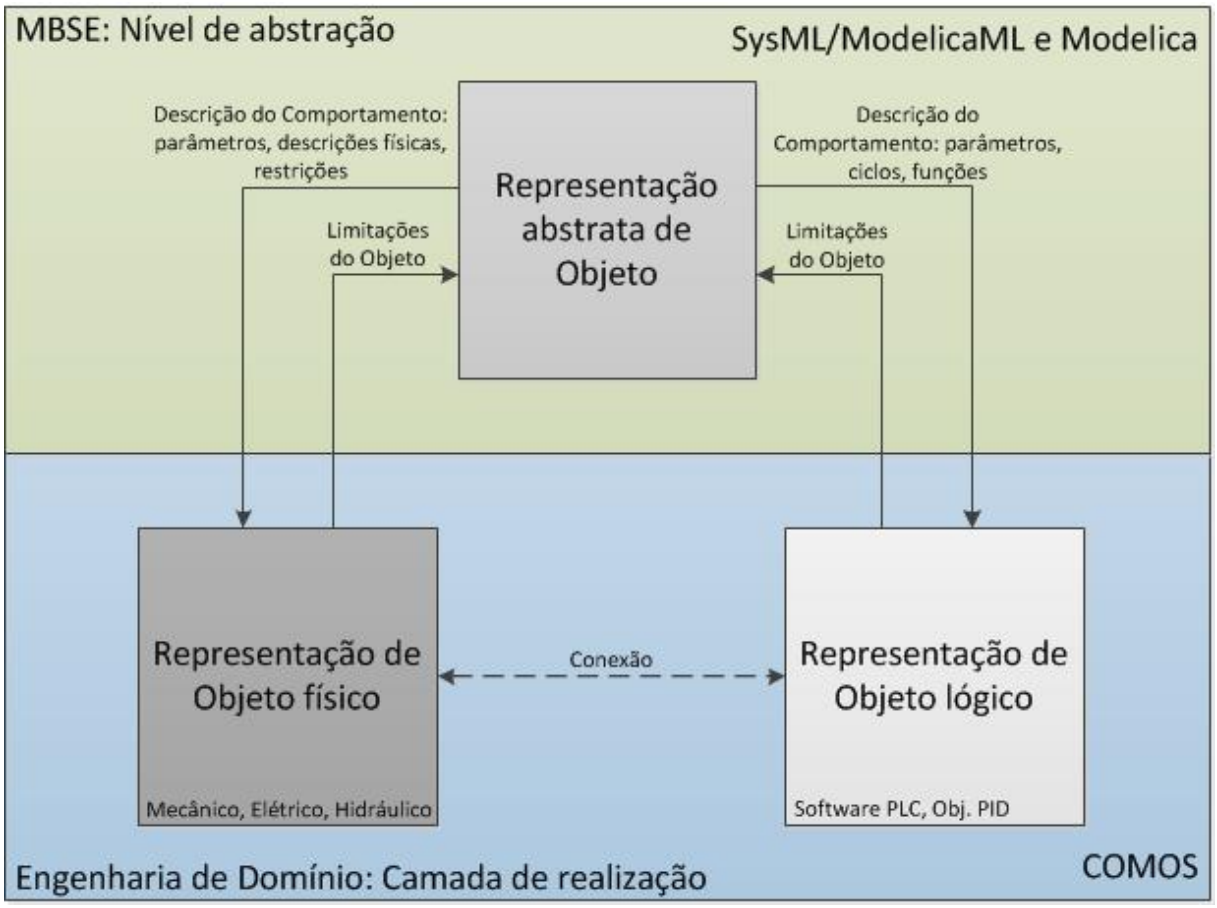


Figura 14 – Troca de informações entre distintas representações de objetos nos modelos.

A plataforma desenvolvida foi utilizada para estudar a interface dos objetos abstratos com os artefatos físicos e lógicos, uma vez que os benefícios levantados no domínio analisado, em termos de tempo de desenvolvimento, estão na reutilização de objetos multidisciplinares e na integração das várias visões envolvidas no projeto (Engenharia Mecânica, Engenharia Elétrica e Engenharia de *Software*).

Portanto, a partir do procedimento proposto por (FRITZSON, 2004) para modelagem e simulação de sistemas físicos e (KIPPER, 2010) para a definição do Repositório de Domínio, propõe-se o método de engenharia de sistemas de automação detalhado na Figura 15.

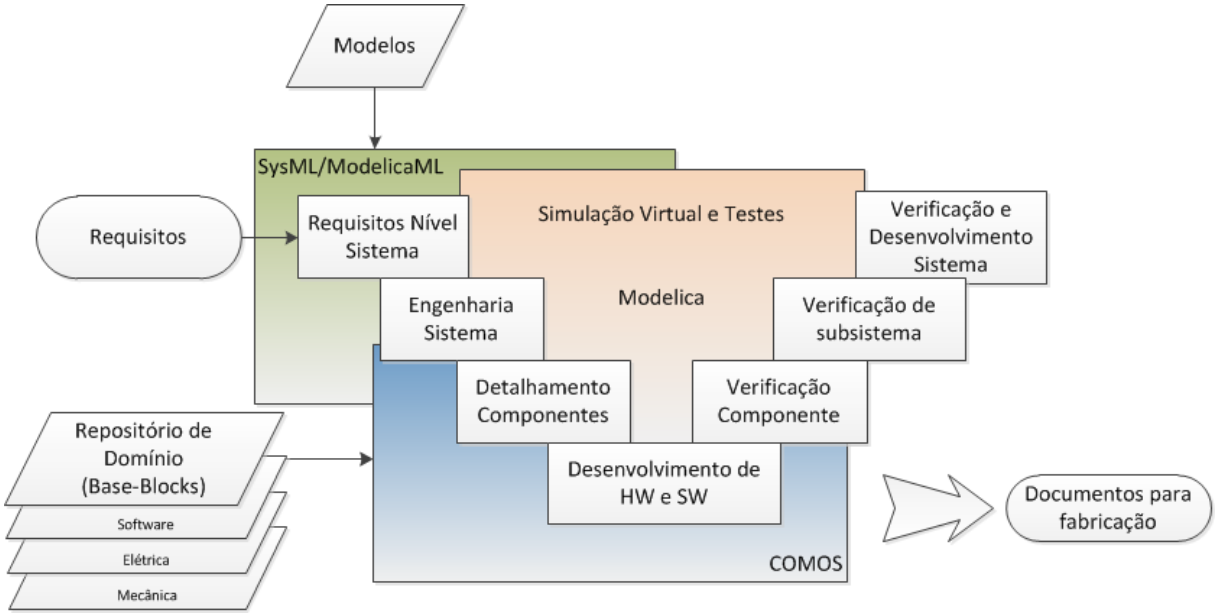


Figura 15 – Método de projeto proposto – MoDElS.iDEAS.

A partir desse ponto, as fases de projeto são detalhadas.

4.2.1 Fase 1: Análise de Requisitos

A análise de requisitos é realizada, a partir da descrição do problema, de forma a conter as necessidades das partes envolvidas. Essa fase deve ser discutida preferencialmente por uma equipe multidisciplinar para que eventuais conflitos entre visões distintas sejam trabalhados imediatamente. Modelos de comportamento e de estrutura utilizados no passado podem ser consultados para reaproveitar experiências bem-sucedidas. Como resultado dessa etapa, deverá ser criado um Diagrama de Requisitos em linguagem ModelicaML que indique os requisitos e suas dependências. Nessa fase é importante criar o Diagrama de Simulação, que conterá informações relativas aos aspectos do projeto que serão posteriormente testados e validados.

4.2.2 Fase 2: Arquitetura da solução

As possíveis soluções para o sistema são descritas usando os Diagramas de Estrutura e Diagrama de Comportamento da ModelicaML com foco na proposição da arquitetura física que solucionará o problema descrito na Fase 1. Nessa etapa deve-se estabelecer a estrutura básica

do sistema, como os comportamentos desejados, as unidades de processamento de informações, a estrutura mecânica e elétrica que suportará a solução. Para aproveitar o ambiente de alto nível de especificação, é prudente utilizar equipes multidisciplinares nessa fase do projeto.

4.2.3 Fase 3: Detalhamento de componentes

Os componentes são detalhados de forma a assumirem características provenientes da Orientação a Objetos, como herança, polimorfismo e encapsulamento. As propriedades e função são atribuídas aos objetos para que o comportamento necessário seja atingido. Os componentes não fundamentais ao funcionamento do sistema podem ser omitidos para que a simulação transcorra de forma a poupar recursos.

Os objetos criados a partir da ModelicaML e passíveis de simulação são transformados para a linguagem Modelica. Para tanto, é necessário que o comportamento dos componentes tenha sido especificado sob forma de equações. Principalmente o Diagrama de Classes e o Diagrama de Equações devem ser um produto dessa fase de projeto.

4.2.4 Fase 4: Atualização do Repositório de Domínio

Os artefatos do Repositório de Domínio são criados a partir da lista de objetos extraída do ModelicaML. Deve-se observar as propriedades importantes para a especificação e fabricação dos componentes de modo que a posterior sincronização entre a camada de abstração e a camada de realização seja possível.

Os componentes passíveis de interligação entre o ModelicaML e o COMOS® são conectados através de programa específico escrito em Java de forma a estabelecer a rede de interação descrita na Figura 14.

Caso haja a necessidade de adicionar uma propriedade nova, proveniente da simulação, o objeto-base do COMOS® deve ser atualizado de forma que todos os demais artefatos instanciados contenham essa funcionalidade.

4.2.5 Fase 5: Sincronização dos objetos ModelicaML e artefatos COMOS®

Os parâmetros dos objetos modelados com ModelicaML são transmitidos para a camada de realização, no COMOS®. Da mesma forma, as propriedades dos artefatos COMOS® são transferidas para os objetos dos modelos em ModelicaML. Essa sincronização é realizada pelo *software* escrito em Java.

4.2.6 Fase 6: Simulação e validação

A simulação virtual do comportamento dos componentes e do sistema ocorre no ambiente OpenModelica a partir do que foi determinado no Diagrama de Simulação. Os parâmetros de simulação como tempo de operação, temperatura ambiente, desgaste de componentes podem ser alterados para verificar o comportamento dinâmico do sistema. A plataforma constituída a partir do Eclipse possibilita a vinculação dos requisitos com parâmetros do sistema e a validação é realizada de forma automática através da ligação dos requisitos com as propriedades das classes, como apresenta a Figura 16. Nesse exemplo, o requisito de nível do tanque principal, chamado de levelInTank (*Client*), é conectado à propriedade tankLevel (*Provider*) do objeto mainTank. Ao realizar a simulação do sistema, o ambiente ModelicaML verifica automaticamente se os valores retornados pelo OpenModelica atendem ao requisito e informa caso alguma condição seja violada. Essa funcionalidade do ModelicaML se chama *Value Binding* (SCHAMAI *et al.*, 2012).

As variáveis de interesse são utilizadas para gerar gráficos do comportamento do sistema. Caso alguma variável não esteja dentro dos limites estabelecidos pelos requisitos, os parâmetros do objeto são alterados e o sistema é simulado novamente. Sempre que há alteração em alguma propriedade, a Fase 5 é disparada para verificar incompatibilidades entre o que as propriedades dos objetos estão necessitando e o que as propriedades dos artefatos podem

realizar. Esse ciclo se repete até que o comportamento do sistema esteja condizente com os requisitos inicialmente estabelecidos.

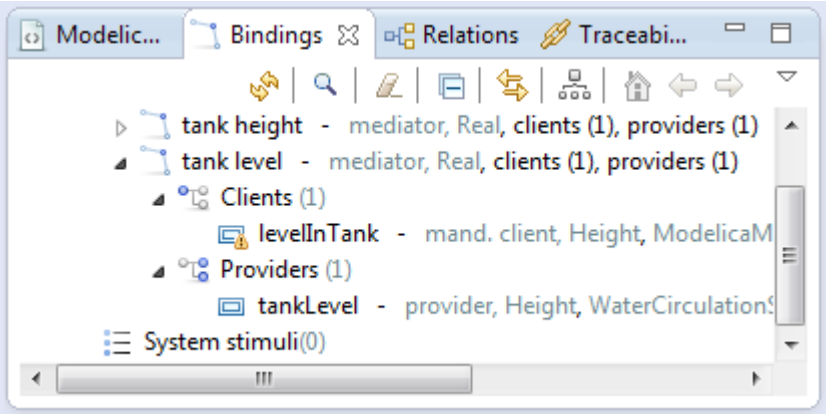


Figura 16 – Validação de requisitos de entrada.

4.2.7 Fase 7: Relatórios de produção

Os relatórios para produção, como desenhos mecânicos e esquemas elétricos, são gerados e completam o ciclo de engenharia.

A proposta do método de engenharia de sistemas é exemplificada na Figura 17. Modelos abstratos são criados para especificar e simular o comportamento do sistema na camada de abstração usando técnicas da MBSE. No detalhe, o funcionamento de uma bomba centrífuga é especificado.

A especificação e análise dos requisitos (Fase 1) é realizada usando a abordagem MBSE e os diagramas específicos para essa etapa. Para o problema de fornecimento de água, são determinadas a vazão, o tipo de líquido a ser bombeado e a temperatura de operação. A arquitetura da solução é discutida (Fase 2). Nesse exemplo, optou-se por uma bomba centrífuga. A bomba centrífuga é detalhada (Fase 3) usando o Diagrama de Classes e Diagrama de Equações. O artefato bomba centrífuga é criado no COMOS® contendo as propriedades e funções especificadas no ModelicaML (Fase 4). Os valores das propriedades são sincronizados (Fase 5) entre as duas plataformas (Eclipse/ModelicaML e COMOS). Os parâmetros da bomba

centrífuga são transferidos para o OpenModelica e a simulação é realizada (Fase 6). Os resultados da simulação são analisados. Caso haja a necessidade de alterar um parâmetro na bomba centrífuga para que os requisitos sejam atingidos, a Fase 5 é executada novamente. Esse passo é importante para que haja consistência no projeto. Se há a necessidade de incrementar a rotação da bomba para atingir o requisito de vazão, primeiramente é necessário verificar na plataforma COMOS® se o artefato escolhido comporta a nova rotação de acordo com os dados do fabricante. A última etapa (Fase 7) compreende a geração de relatórios, esquemas e desenhos que permitam a fabricação ou aquisição do componente de prateleira.

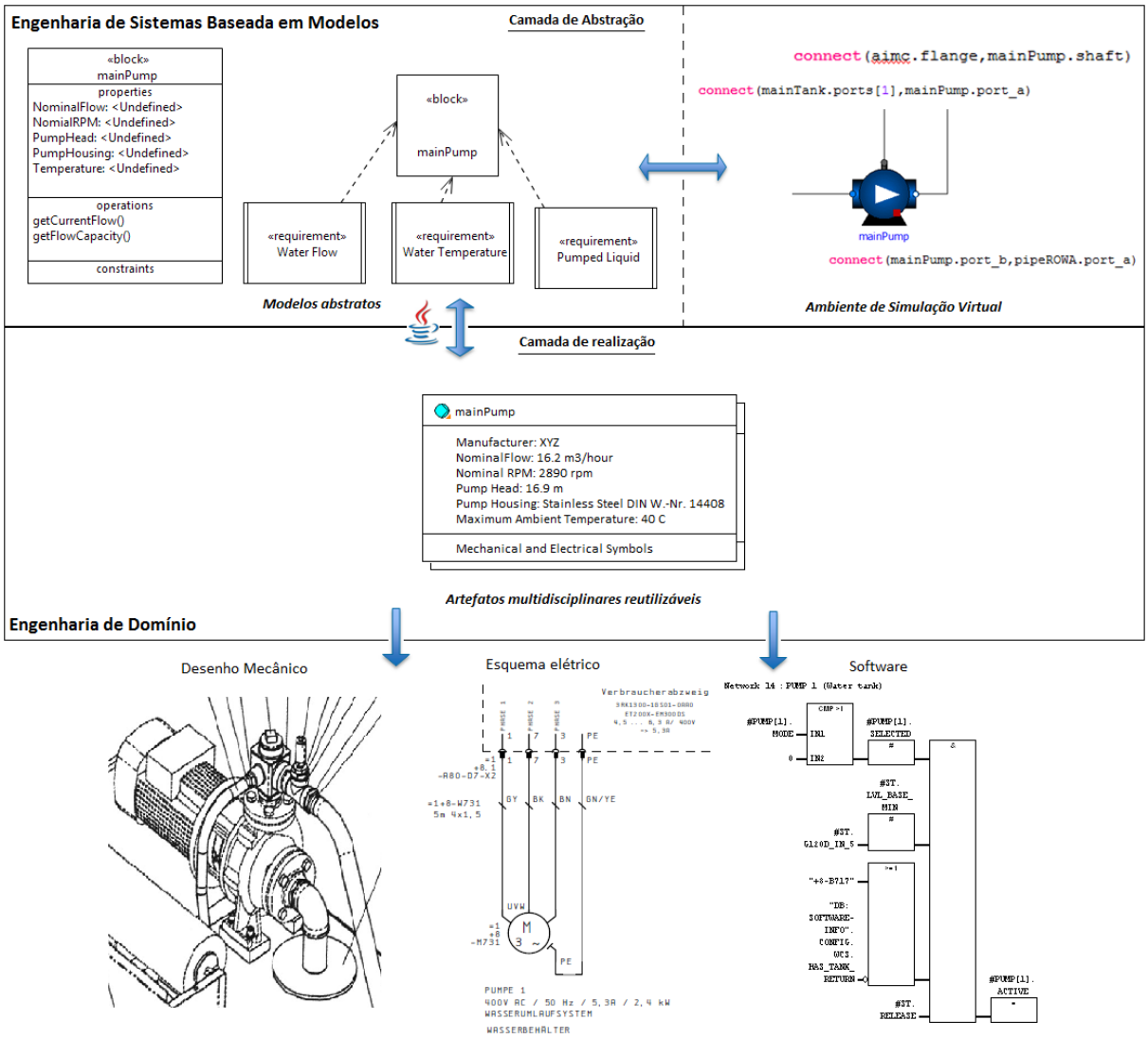


Figura 17 – Interligação entre diferentes camadas de abstração.

4.3 COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTAÇÃO E MÉTODO BASEADO EM MODELOS

Após o estabelecimento do método MoDEls.iDEAS, realizou-se a sua comparação com o método de engenharia estabelecido na GE. Uma vez que o processo de engenharia utilizado pela empresa parte da definição de requisitos do sistema até a geração de documentos, da mesma forma que o método MoDEls.iDEAS, foi possível comparar as duas abordagens. Essa comparação foi realizada com o intuito de validar a proposta como um novo processo de engenharia que pudesse substituir o processo existente, de forma que, a partir das mesmas entradas (Análise de Requisitos) fossem geradas saídas semelhantes (Documentação). Com a disponibilidade das informações dos tempos de desenvolvimento de sistemas existente e com o levantamento dos tempos gastos com o novo método, objetivou-se comparar os dois processos em termos de custos. O resumo da proposta de comparação pode ser verificado na Figura 18.

Um estudo de caso foi utilizado para validação do método proposto e permitiu a comparação com a abordagem centrada em documentos.

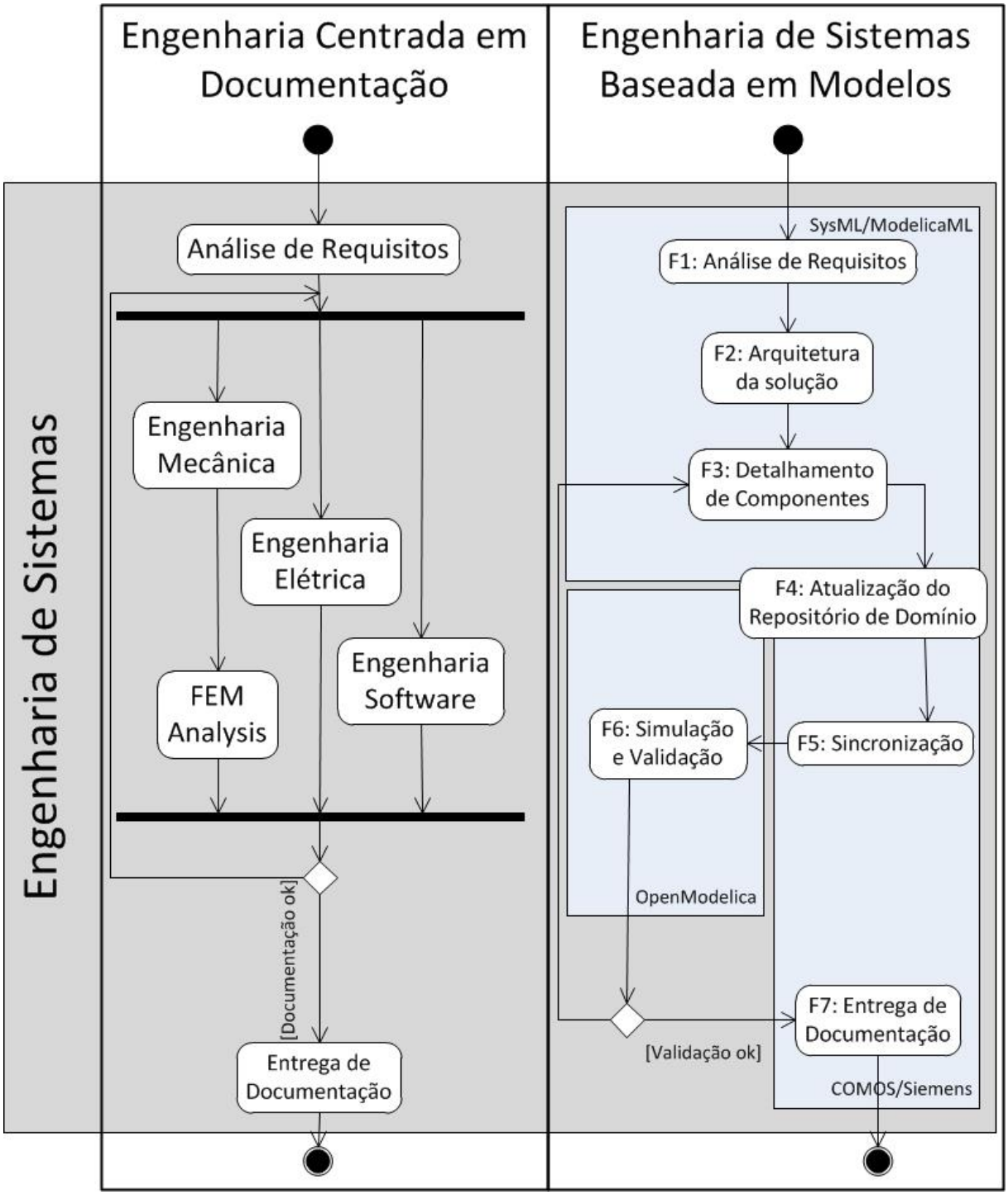


Figura 18 – Proposta de comparação entre métodos.

5 VALIDAÇÃO

A validação foi desenvolvida sob dois aspectos. A execução do projeto do estudo de caso com o novo método foi realizada com vistas a validar o seu funcionamento. Os resultados de tempos de engenharia foram comparados com o método de projeto usado pela GE. Foram executadas pequenas mudanças no projeto para avaliar as vantagens e desvantagens da abordagem em relação à reutilização de esforços de engenharia.

5.1 DESCRIÇÃO DO OBJETO DE ESTUDO DE CASO

Para atingir os objetivos propostos, procurou-se um projeto de sistema de automação que possuísse aderência às seguintes premissas:

a) Ser composto de artefatos de distintas disciplinas (Engenharia de *Software*, Engenharia Elétrica e Engenharia Mecânica e suas derivações);
b) Apresentar variabilidade dos componentes entre os projetos para possibilitar a análise do reuso, que é um importante benefício das abordagens baseadas em modelos;
c) Possuir informações a respeito do esforço de engenharia necessário para o projeto do sistema usando métodos centrados em documentação.

Os equipamentos de ensaio não destrutivo por ultrassom fabricados pela GEIT, também chamados de *Testing Machines*, são compostos por 4 unidades funcionais representadas na Figura 19. A porção 1 (Movimentação Mecânica) do equipamento é responsável por movimentar a peça a ser inspecionada em relação aos sensores ultrassônicos. Os sistemas de movimentação fabricados pela empresa possuem todas as disciplinas necessárias (premissa a), mas a variabilidade de requisitos entre os projetos é tamanha (premissa b) que inviabiliza as análises propostas.

A segunda parte da máquina é responsável pelo ensaio de Ultrassom (Inspeção por Ultrassom). O ensaio é realizado por cristais piezoelétricos especificados e posicionados para captar os defeitos no material ensaiado. Placas eletrônicas que utilizam DSP (*Digital Signal Processing*) analisam os dados obtidos. Essa porção do equipamento foi descartada da análise, pois a disciplina mecânica não apresenta variabilidade (premissa b), ou seja, os componentes mecânicos que perfazem essa porção do equipamento são sempre iguais.

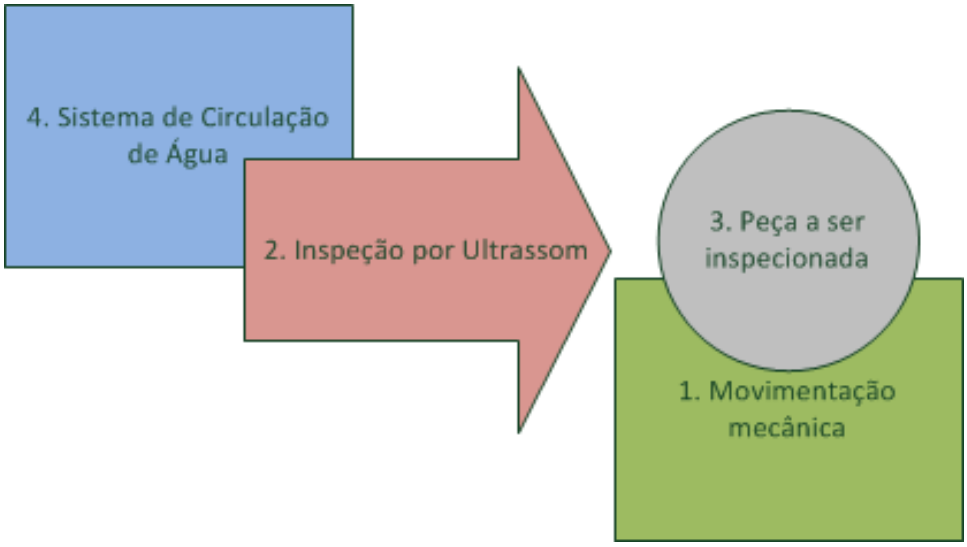


Figura 19 – Estrutura do sistema de Ensaio Não Destrutivo por ultrassom.

O terceiro componente do sistema trata-se da peça a ser inspecionada. Varia de acordo com o produto do cliente e pode ser desde fuselagens de fibra de carbono para aviões até tubos de aço sem costura. São produtos mecânicos que não apresentam componentes da disciplina elétrica (premissa a) e por isso foram descartados da análise.

A quarta parte do equipamento se chama Sistema de Circulação de Água ou WCS (*Water Circulation System*). É responsável por fornecer a água necessária para realizar o acoplamento sônico entre o cristal piezoelétrico e o material a ser inspecionado. Esse sistema possui características que são condizentes com as premissas acima elencadas e foi selecionado como estudo de caso.

O WCS possui duas unidades básicas. A primeira, chamada de Tanque Principal, é responsável por fornecer água no volume adequado ao equipamento de ensaio. A segunda unidade é chamada de Tanque de Retorno e funciona captando a água do processo e devolvendo-a ao Tanque Principal para recirculação. A Figura 20 apresenta uma fotografia dos componentes principais do WCS. Há equipamentos auxiliares para filtragem de partículas e controle de temperatura.

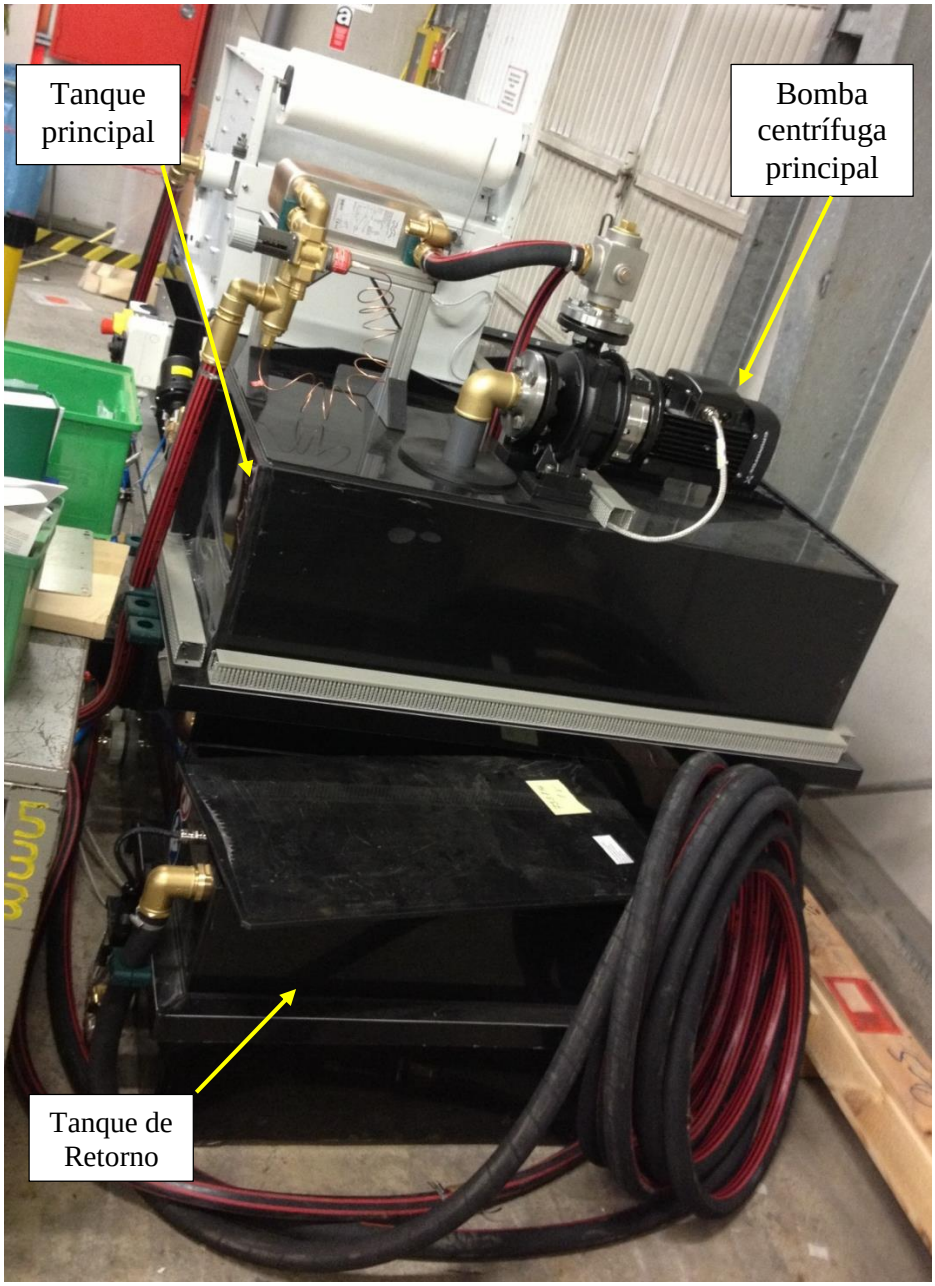


Figura 20 – Componentes constituintes do WCS.

A entrada de água no tanque principal serve para repor as perdas ocorridas por evaporação e é controlada por uma válvula pneumática. Sensores de nível instalados no tanque principal fornecem as informações necessárias para que o controlador decida quando a entrada de água deve ser acionada. A bomba principal fornece o volume de água necessário para que a máquina funcione adequadamente. A tubulação que liga a bomba principal até a *Testing Machine* varia em termos de diâmetro e altura dependendo dos requisitos impostos pelo projeto. Após a água passar pela *Testing Machine*, escoa por gravidade até o tanque de retorno. A bomba de retorno é responsável por manter o nível do tanque de retorno dentro dos patamares informados pelos sensores de nível. A água de retorno é enviada através de tubulação até o tanque principal, completando o ciclo. Um resumo do que foi descrito pode ser visto na Figura 21, que apresenta os objetos de OpenModelica conectados.

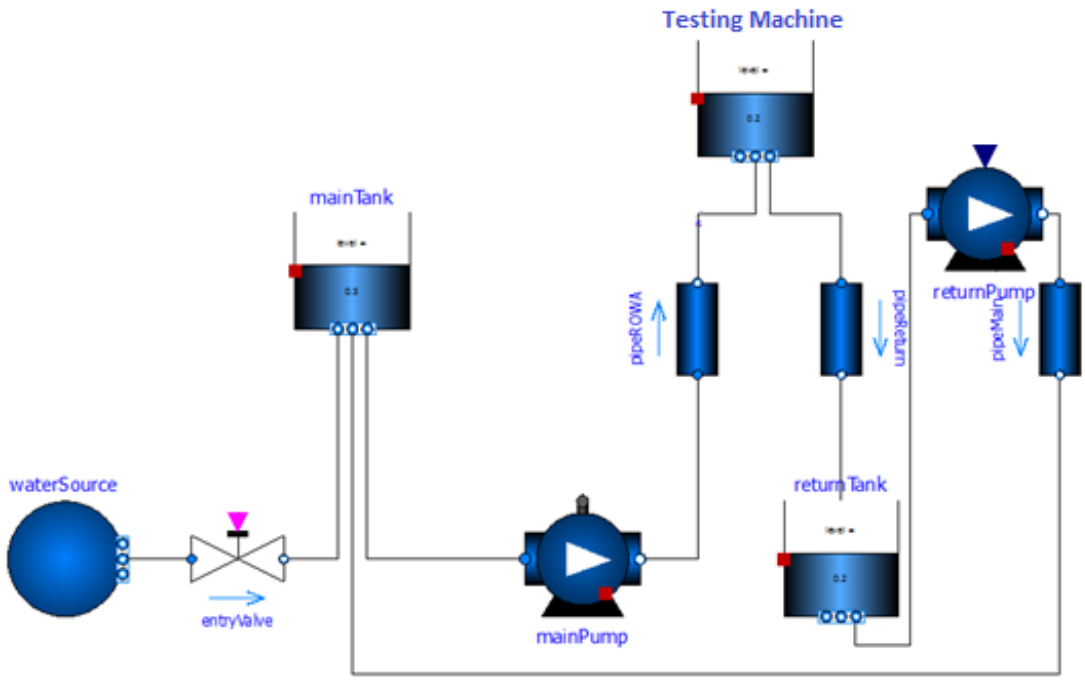


Figura 21 – Representação do WCS em objetos do OpenModelica.

5.2 VALIDAÇÃO DO MÉTODO PROPOSTO

Para a realização do que foi proposto, foi utilizada a plataforma composta de Eclipse, OpenModelica e COMOS®. O detalhamento da interligação dessas ferramentas será apresentado a seguir.

Uma vez que a linguagem ModelicaML é uma extensão que usa características de UML e de SysML, a plataforma em Eclipse fornece a possibilidade de construir diagramas em quaisquer dessas linguagens, o que torna o ambiente flexível (POP;AKHVLEDIANI e FRITZSON, 2007; SCHAMAI *et al.*, 2009). Com o uso de diagramas SysML e ModelicaML realizou-se a análise dos requisitos do sistema estudado.

5.2.1 Fase 1: Análise de Requisitos

A fase de Análise de Requisitos inicia com a elaboração de documento constando os requisitos do sistema em estudo. Esse documento é elaborado por engenheiros qualificados ou por grupo de trabalho que carrega experiência no contexto. A partir dessas informações é gerado o Diagrama de Requisitos no ModelicaML, vide exemplo da Figura 22, que será utilizado para repassar os parâmetros de aceite ao modelo.

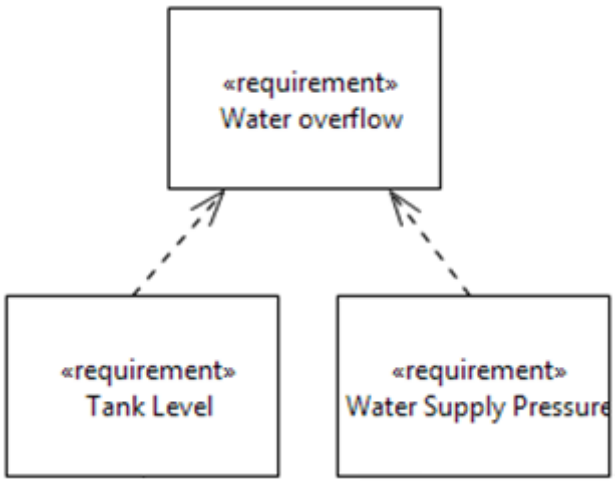


Figura 22 – Exemplo de Diagrama de Requisitos do ModelicaML.

5.2.2 Fase 2: Arquitetura da solução

Em função da análise realizada na Fase 1, deve-se optar por uma arquitetura que solucione o problema.

O sistema utilizado atualmente pela GEIT foi modelado e simulado, buscando avaliar se o seu comportamento era condizente com os requisitos. Como o sistema atendeu aos requisitos especificados, optou-se por manter a arquitetura existente, que é apresentada na Figura 23.

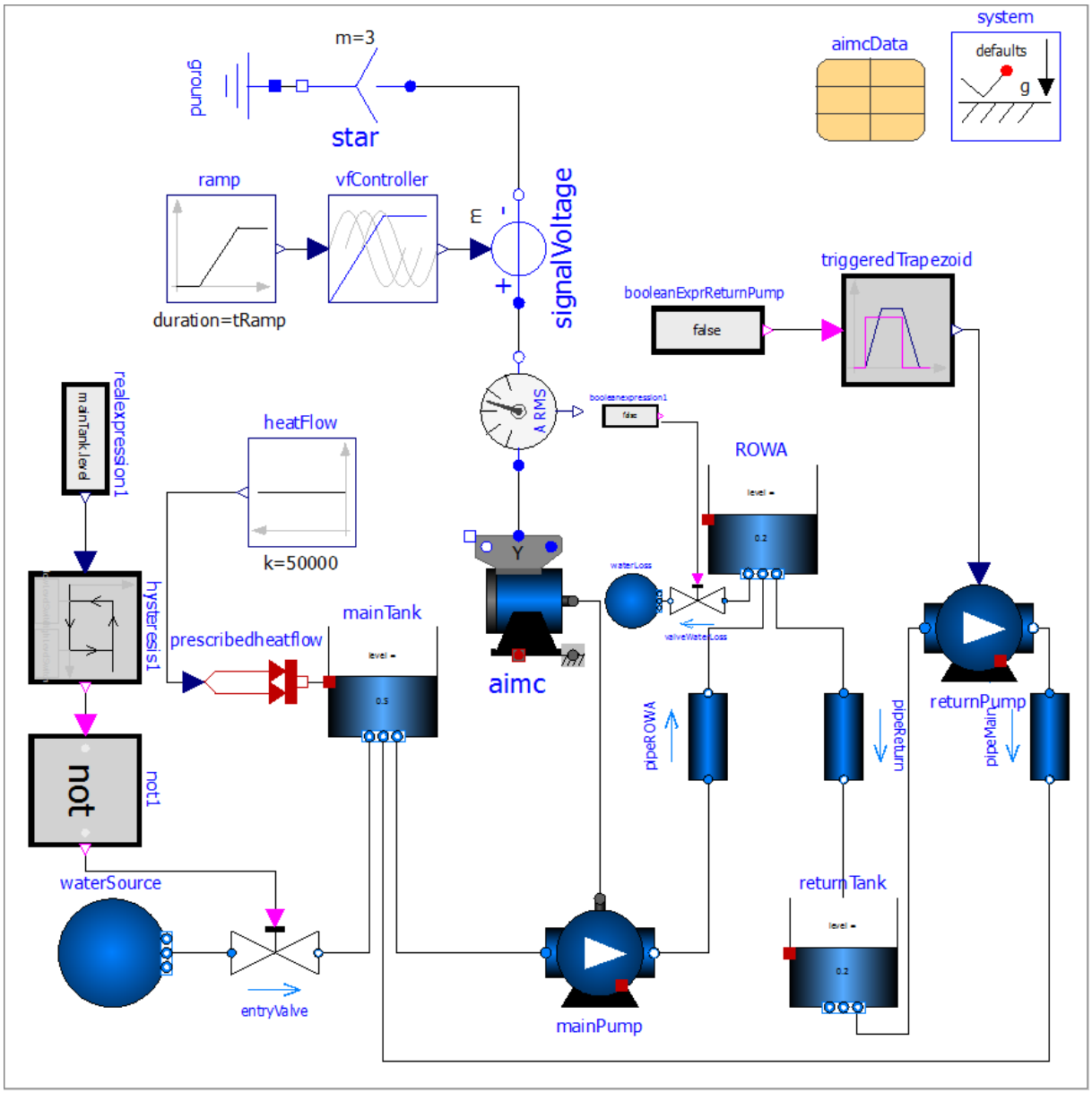


Figura 23 – Arquitetura física do WCS representada no OpenModelica.

5.2.3 Fase 3: Detalhamento de componentes

Os componentes constituintes do sistema foram detalhados com as propriedades e funções necessárias para a especificação e simulação. A Figura 24 apresenta a estrutura em árvore gerada e mostra o detalhe das propriedades do componente *waterSource*.

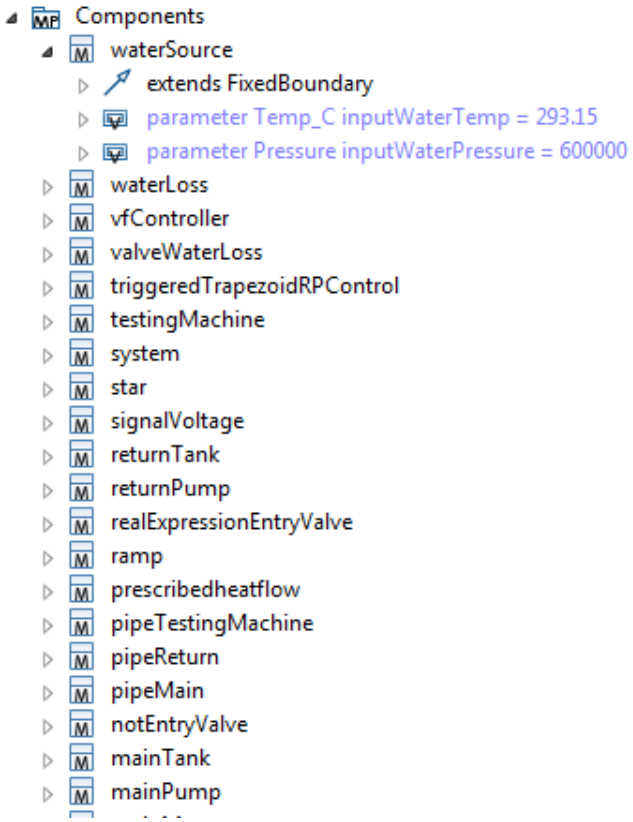


Figura 24 – Estrutura de componentes do sistema WCS.

A descrição do comportamento dos objetos foi realizada diretamente nas propriedades, sob a aba *Equations*. É possível detalhar o comportamento dos componentes em termos do seu funcionamento físico ou lógico bem como realizar a interligação entre os componentes do sistema, como mostra a Figura 25.

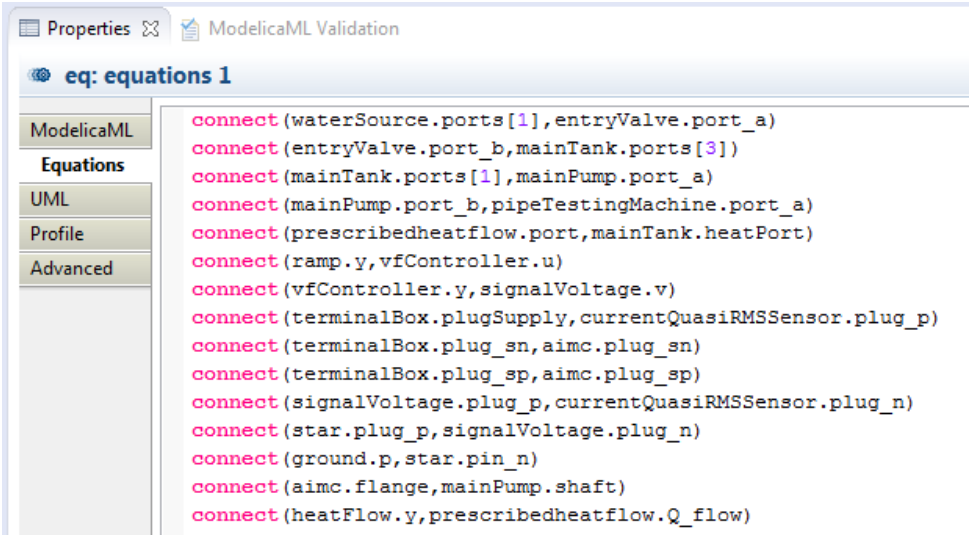


Figura 25 – Descrição do comportamento de objetos.

5.2.4 Fase 4: Atualização do Repositório de Domínio

A Análise do Domínio, parte inicial da proposta de Engenharia de Domínio para Sistemas de Automação de (KIPPER, 2010), foi realizada usando SysML devido às vantagens que a linguagem fornece, como flexibilidade de representações e interdisciplinaridade.

Inicialmente tentou-se realizar a transformação dos objetos ModelicaML para artefatos COMOS® usando o padrão XMI. Porém, o COMOS® somente possui função de importar componentes XMI do tipo *software* oriundos especificamente de controladores programados com a ferramenta PCS7® da Siemens®.

De modo a atingir maior flexibilidade e abrangência, a geração e atualização dos artefatos COMOS® a partir dos objetos ModelicaML foi realizada por programa escrito em Java, na plataforma Eclipse, que realiza as modificações diretamente no banco de dados Access® do COMOS®.

O Repositório do Domínio foi desenvolvido de modo a classificar os componentes do sistema quanto à sua repetição e foram organizados em formato de árvore. A Figura 26 representa a estrutura em árvore do projeto do WCS no COMOS®.

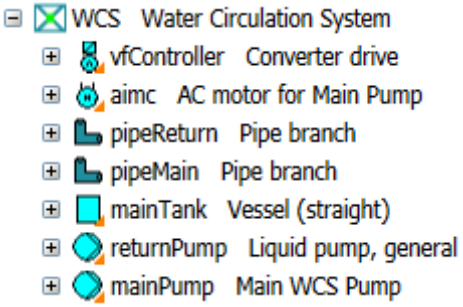


Figura 26 – Estrutura de componentes do Domínio.

Os artefatos do Repositório do Domínio foram detalhados em termos de interligação, como conexões hidráulicas ou elétricas.

5.2.5 Fase 5: Sincronização dos objetos

Além da criação e atualização dos artefatos no COMOS® a partir dos diagramas do ModelicaML, o programa escrito em Java realiza a atualização dos valores das propriedades em ambas as plataformas e verifica se há alguma incompatibilidade. Significa que, se alguma propriedade de objeto do ModelicaML possuir valor não compatível com o seu artefato equivalente no COMOS®, o programa gera uma mensagem de aviso.

A Figura 27 mostra o tubo pipeMain recebendo os parâmetros do aplicativo em Java. Nesse exemplo, se o tubo pipeMain do COMOS® estivesse recebendo um valor de temperatura acima de 40 graus, o programa Java informaria que, para o sistema projetado, o tubo escolhido na plataforma COMOS® não é adequado. Deve-se então procurar um tubo adequado à nova condição de funcionamento ou alterar as variáveis do sistema para que a temperatura não atinja esse valor.

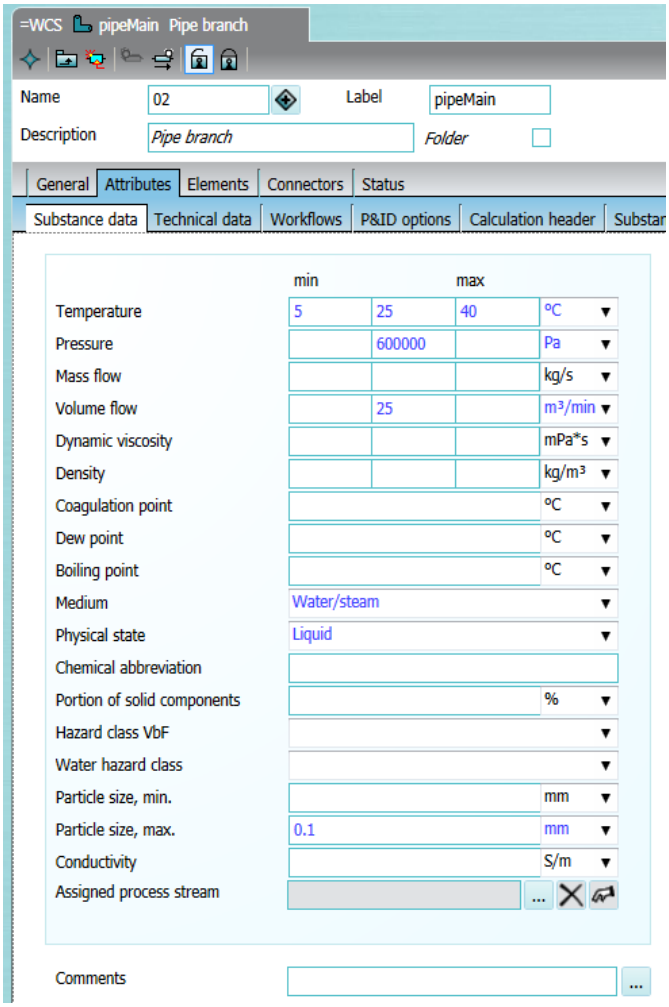


Figura 27 – Integração dos parâmetros simulados na ferramenta COMOS®.

5.2.6 Fase 6: Simulação e validação

Os objetos passíveis de simulação, elencados na Fase 1, são automaticamente convertidas para linguagem Modelica com o auxílio da transformação SysML/Modelica realizada pela plataforma Eclipse (PAREDIS *et al.*, 2010). Componentes como bombas centrífugas, tanques, válvulas, motores e inversores de frequência estão disponíveis na biblioteca padrão do OpenModelica e foram reutilizados para compor algumas partes do sistema.

A Figura 28 mostra como os requisitos contidos nos Diagramas de Requisitos são conectados aos objetos ModelicaML e aos valores simulados no OpenModelica. Ao requisito

Max_level_of_liquid_in_tank são atribuídas as entradas levelInTank, oriunda da simulação realizada no OpenModelica, e tankHeight, proveniente do projeto do Sistema de Automação. A simulação do comportamento dinâmico do tanque transfere os valores das propriedades para o ModelicaML que realiza a análise e apresenta no *Status* do Requisito, se ele foi atendido ou não. Há a possibilidade de descrever o requisito de forma textual, como forma de comentário, da mesma forma que em SysML.

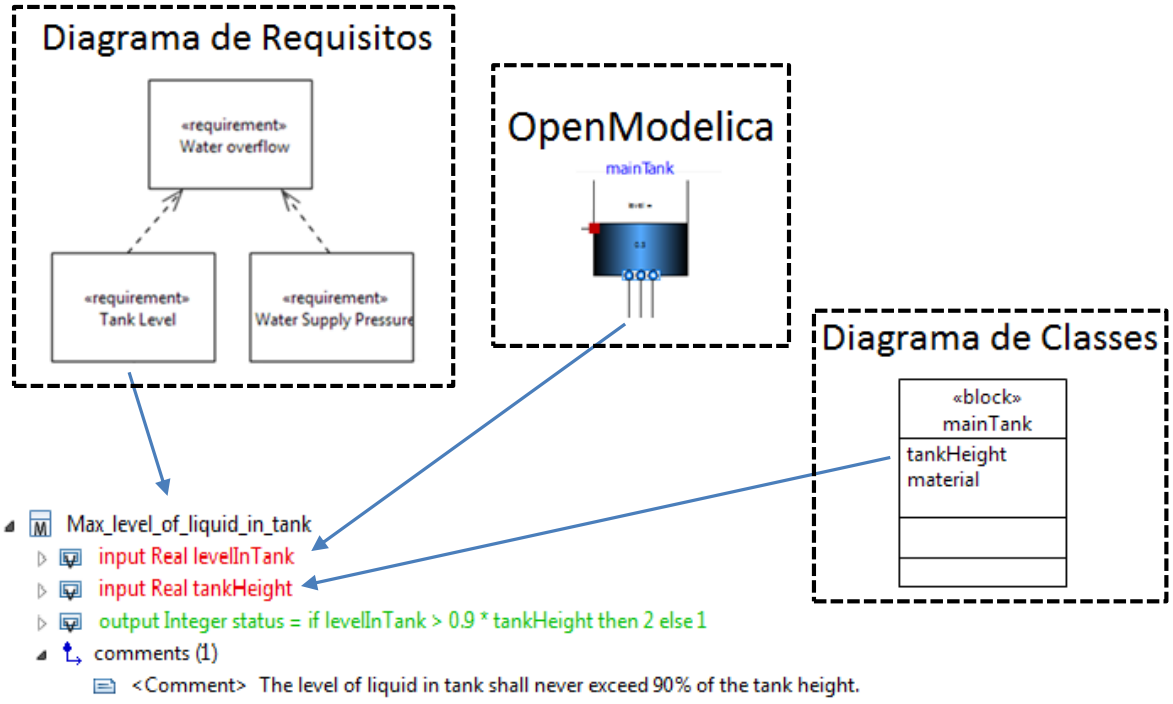


Figura 28 – Exemplo de avaliação de requisitos do domínio WCS.

O processo de alteração de parâmetros dos objetos, simulação e validação foi repetido até que os requisitos tenham sido atingidos pelo sistema. Alternativas de solução podem ser investigadas ao longo da iteração de modo que uma resposta adequada, de acordo com os requisitos, seja escolhida. Os novos parâmetros simulados no OpenModelica são importados para os artefatos reutilizáveis do Repositório de Domínio usando o aplicativo Java (retornando à Fase 5 – Sincronização dos objetos).

5.2.7 Fase 7: Relatórios de produção

Os documentos necessários à fabricação foram gerados pelo COMOS® com as modificações realizadas pela importação de parâmetros. A partir desse ponto se pode adquirir componentes COTS que atendam às especificações oriundas do processo de engenharia ou fabricá-los.

5.3 COMPARAÇÃO ENTRE MÉTODO CENTRADO EM DOCUMENTAÇÃO E MÉTODO BASEADO EM MODELOS

O WCS é uma versão simplificada de uma planta industrial e pode ser categorizada como um sistema automatizado de fornecimento de água para ensaios não destrutivos via ultrassom. Um desenho tridimensional parcial do sistema pode ser visto na Figura 29. Como tal, sofre modificações para se adaptar às necessidades de diferentes projetos. Para testar a proposta, foram executadas alterações no projeto utilizando o método MoDElS.iDEAS.

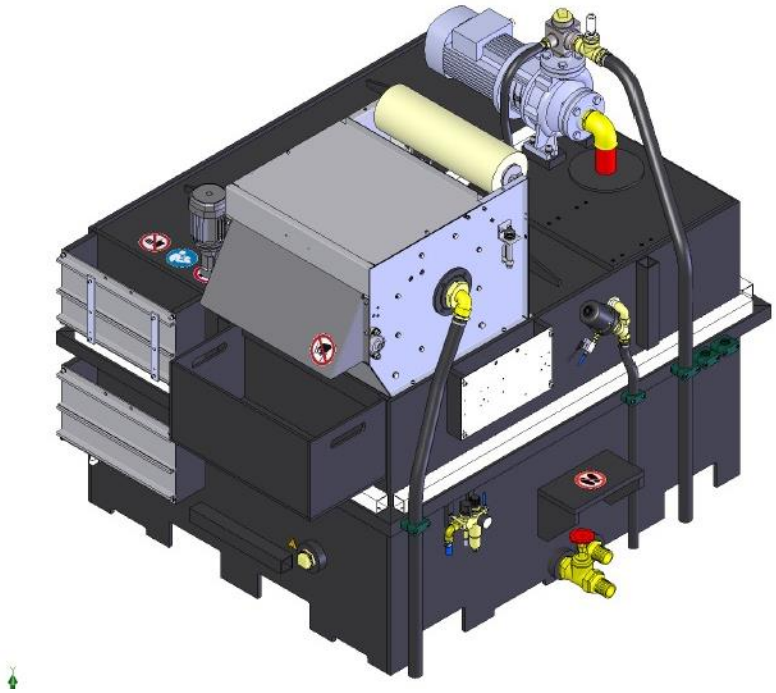


Figura 29 – Desenho tridimensional do WCS.

5.3.1 Mudança de tamanho da bomba principal

De acordo com os dados fornecidos pela GE, a alteração de tamanho da bomba principal ocupa 8 horas de um engenheiro mecânico até que o processo completo de engenharia seja cumprido.

Para utilizar o método proposta neste trabalho, primeiro analisou-se os requisitos do sistema. No que diz respeito à bomba principal, as principais características desejadas são o atendimento da vazão adequada e a manutenção da temperatura, de acordo com a Figura 30.

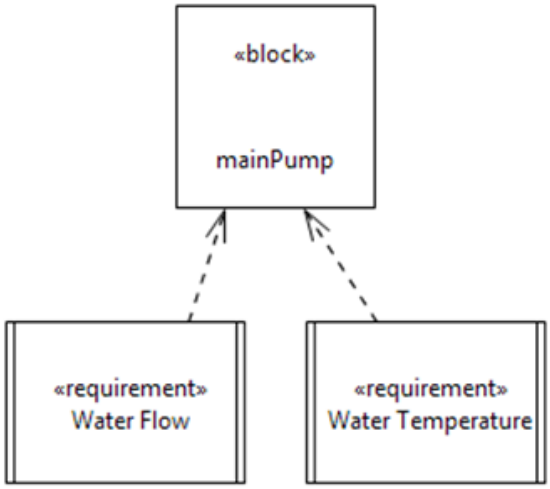


Figura 30 – Requisitos do sistema relacionados à bomba principal.

Após estabelecer os limites de vazão e coluna d'água, foi possível determinar o comportamento da bomba centrífuga do modelo Modelica no que diz respeito à curva quadrática de volume em função da pressão. Para a bomba centrífuga modelo NB32-160.1/139 50Hz de fabricação Grundfos, a referida curva é apresentada na Figura 31.

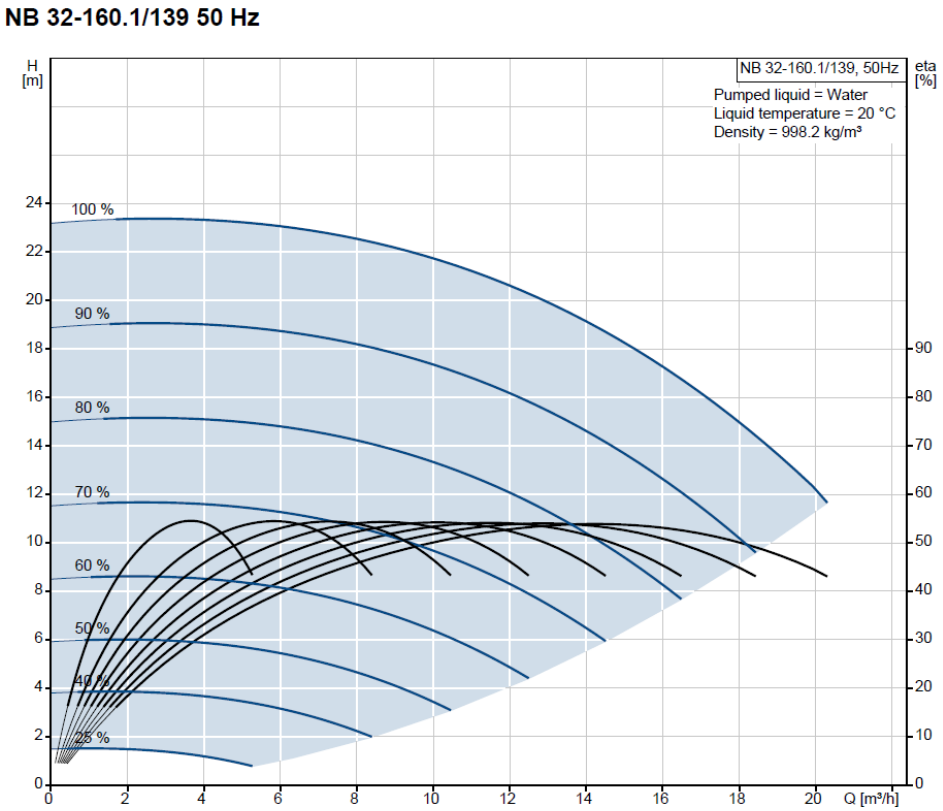


Figura 31 - Curva característica da bomba Grundfos NB32-160.1/139 (GRUNDFOS, 2013).

Ao especificar 3 pontos de operação, é possível informar ao Modelica a curva característica da bomba, que pode ser vista na Figura 32.

```
function flowCharacteristic =
Modelica.Fluid.Machines.BaseClasses.PumpCharacteristics.quadraticFlow
                                    (
                                    V_flow_nominal = {0,3,6},
                                    head_nominal = {4,3.91,3.32}
                                    ),
```

Figura 32 – Curva característica da bomba principal representada em código Modelica.

Ao verificar o nível dos tanques através da simulação do modelo no OpenModelica, a Figura 33 demonstra que o nível de água na *Testing Machine* (linha vermelha do gráfico) diminui ao longo do tempo. Isso significa que o volume de água disponibilzado não é adequado para efetuar o ensaio. É necessário, portanto, alterar parâmetros da bomba.

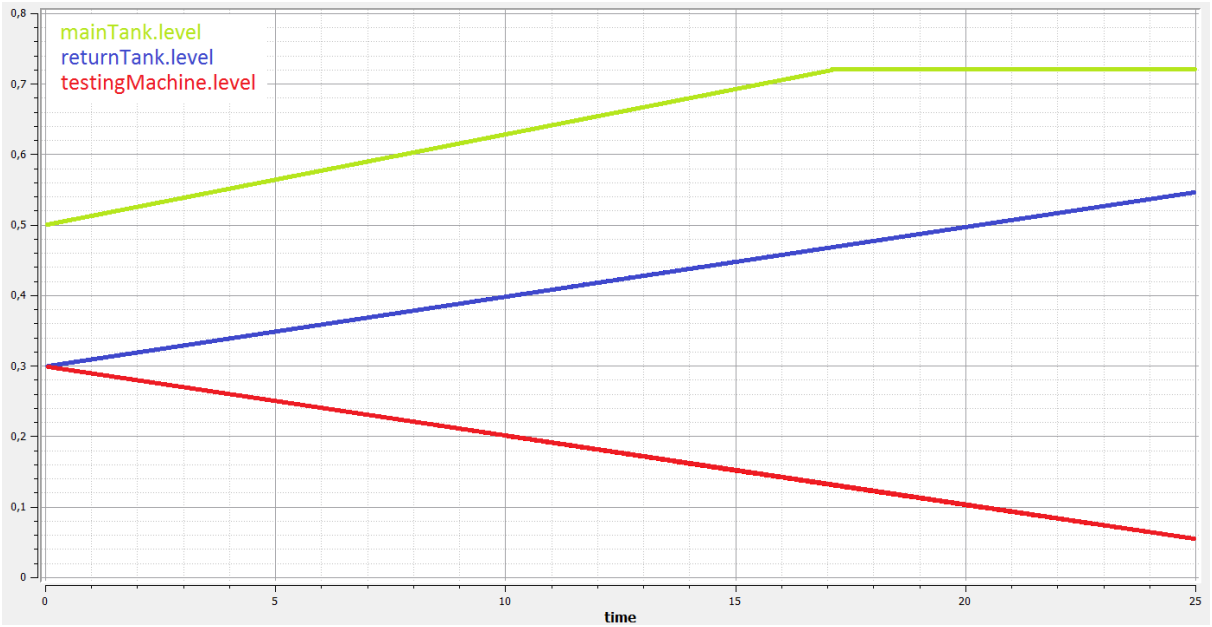


Figura 33 – Simulação dos níveis dos tanques do sistema antes da modificação da bomba.

Ao trocar os parâmetros de fluxo, o novo modelo altera seu comportamento e a vazão de água na *Testing Machine* é estabilizada, de acordo com o que está representado na Figura 34.

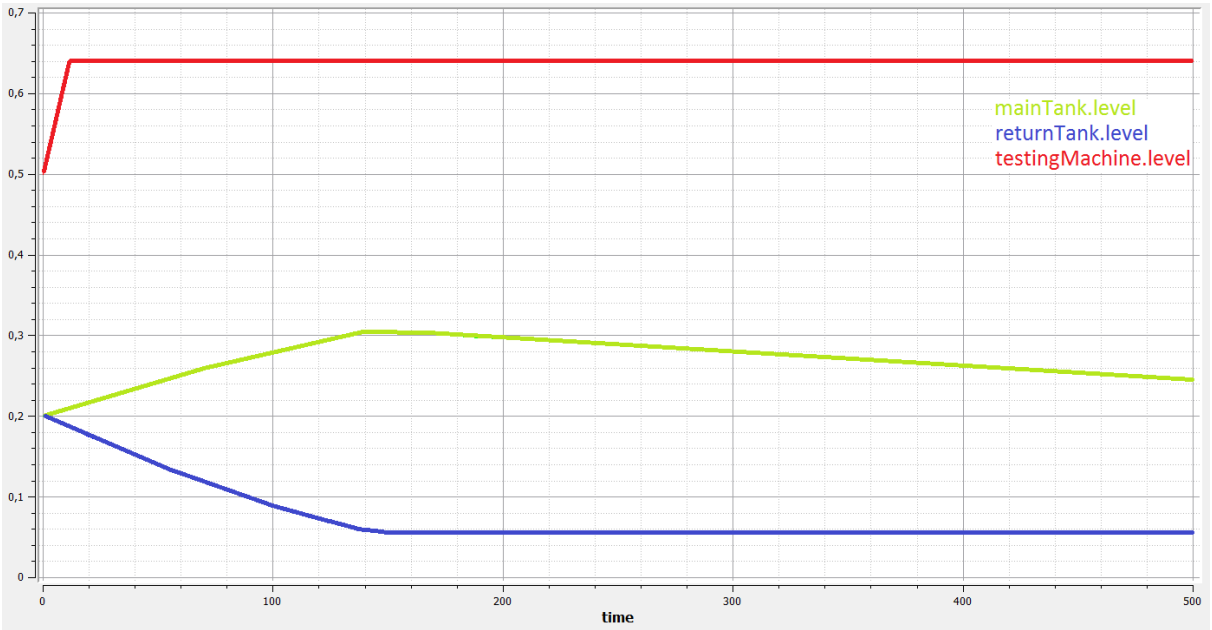


Figura 34 – Simulação dos níveis dos tanques do sistema após a modificação da bomba.

De posse dos novos parâmetros da bomba centrífuga principal, o aplicativo Java transfere os valores para o artefato técnico chamado mainPump no COMOS®, realizando a integração e a validação entre o modelo MBSE e o modelo em COMOS®, como pode ser visto na Figura 35.

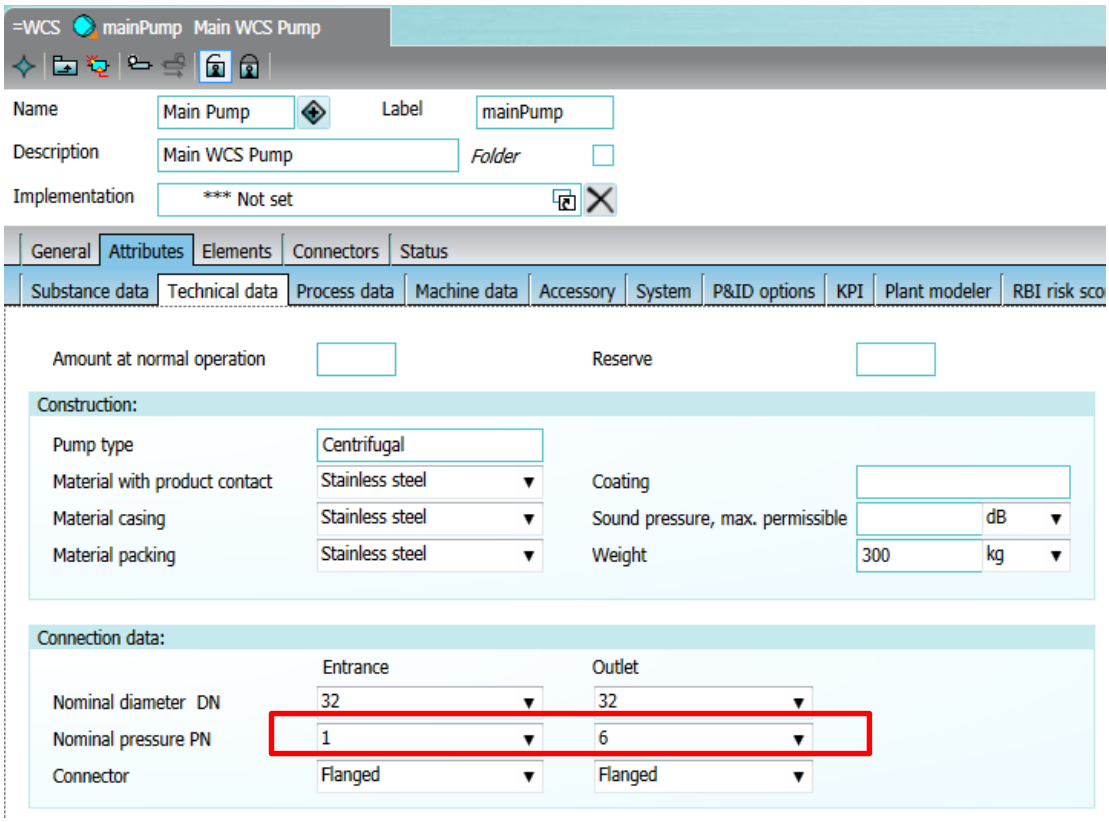


Figura 35 – Detalhe da integração de parâmetros do modelo abstrato no COMOS®.

Caso a bomba centrífuga utilizada no COMOS® não atenda às novas especificações, o engenheiro pode escolher um novo produto com a ajuda dos artefatos armazenados no repositório do domínio. A partir dos relatórios gerados pelo COMOS® é possível iniciar o processo de aquisição das peças e a produção do sistema.

A troca de parâmetros da bomba principal usando os modelos apresentados foi realizada 10 vezes e em média utilizou 3 horas de um engenheiro treinado na plataforma, representando considerável economia de tempo em relação ao método anterior. A maior parte do tempo foi empregada na análise dos requisitos e simulação do modelo físico com os novos parâmetros

propostos, ou seja, em atividades de engenharia e não com a manipulação de arquivos, como ocorre na abordagem centrada em documentação. A integração dos parâmetros no COMOS® e a geração de relatórios são realizadas de forma automatizada e não ocuparam mais do que 30 min.

Portanto, em resumo, pretendeu-se:

1. Utilizar métodos dirigidos a modelos para especificar um sistema de automação existente em um estudo de caso real;
    a. Refinar os objetos do modelo através da integração com ferramenta de simulação matemática chamada OpenModelica e avaliar as vantagens e desvantagens da sua utilização;
2. Criar um repositório de objetos de um domínio existente com o uso do *software* COMOS® da Siemens®;
    a. Realizar análise da ferramenta COMOS® em relação à sua capacidade de integração com ferramentas baseadas em MBSE.
3. Comparar os métodos Centrado em Documentação e Baseado em Modelos de engenharia de Sistemas de Automação;
    a. Quanto ao tempo de desenvolvimento da plataforma e curva de aprendizado;
    b. Quanto à possibilidade de integração entre diferentes disciplinas;
    c. Quanto ao tempo necessário para criar novas variantes de componentes;
4. Realizar estudo sobre a integração de ferramentas MBSE e Engenharia de Domínio a fim de minimizar erros de projetos.

6 CONCLUSÃO

Um documento do tipo CAx funciona como um relatório seja para conferência futura ou como uma base sólida para que a fábrica realize o componente especificado. Na engenharia centrada em documento, os engenheiros trabalham em um nível próximo ao relatório, quando não trabalham diretamente nele, como é o caso de desenhos mecânicos 2D ou esquemas elétricos. A abordagem proposta, que inicia na concepção de sistemas em nível abstrato, procura fornecer uma plataforma que proporcione a geração de projetos de forma ampla e superficial, num primeiro momento. Esse aspecto possibilita engenheiros a projetarem um sistema mais consistente e menos propenso a erros, uma vez que fornece condições de simulação virtual do seu comportamento desde as fases iniciais. O ambiente de simulação proposto nesse trabalho traz benefícios no que diz respeito ao prognóstico de comportamento do sistema em função da interligação de objetos de distintas disciplinas.

Mesmo utilizando métodos baseados em modelos, a engenharia de sistemas necessita de uma etapa de geração de documentos para aquisição ou produção dos componentes. O presente trabalho procurou demonstrar os benefícios que a utilização integrada de métodos baseados em modelos com ferramentas de engenharia de domínio podem trazer na geração integrada de documentação.

A vasta gama de símbolos criada para especificar procedimentos de montagem, acabamento, formatos e conexões físicas proporciona uma base sólida para as especificações de procedimentos de fabricação e aquisição de materiais. Nesse sentido, a ferramenta COMOS® fornece suporte à geração de documentos detalhados e se mostrou um produto eficaz no armazenamento centralizado das informações do Domínio.

A interligação de plataformas MBSE com ferramentas de Engenharia de Domínio foi testada e as vantagens comprovadas, uma vez que não seria possível extrair informações que pudessem conduzir a construção do sistema somente a partir dos modelos abstratos.

Com SysML foi possível modelar vários aspectos do sistema, partindo-se dos requisitos necessários. Contudo, deve-se analisar com cuidado o que é necessário modelar e simular, uma vez que o processo demanda esforço. A curva de aprendizado necessária para trabalhar com as novas ferramentas e métodos exige tempo da equipe envolvida. O tempo necessário para desenvolver modelos físicos condizentes com a realidade torna a modelagem dispendiosa e exige constante aperfeiçoamento.

A construção da plataforma proposta demandou cerca de 300 horas enquanto que os modelos do estudo de caso proposto foram criados em 160 horas. Isso demonstra o esforço necessário para atingir os objetivos de uma plataforma MBSE, atestando que os modelos devem ser aplicados apenas em partes específicas do projeto, onde podem trazer benefícios para o correto funcionamento e redução de custo do projeto (WEILKIENS, 2011).

Objetos simples, como conexões hidráulicas e dispositivos pneumáticos, foram omitidos da simulação física por entender-se que não influenciariam de forma significativa no comportamento do sistema. Essa simplificação trouxe benefícios em termos do esforço computacional necessário para simular o WCS.

SysML é apontada como sendo uma ferramenta apropriada para modelagem de sistemas que usam *software* ou *firmware* de forma intensiva devido à sua ligação com UML. Mas a linguagem ainda é limitada no que se refere à conexão com outras disciplinas que não *software*. Nesse aspecto, ModelicaML e Modelica se mostraram linguagens promissoras no sentido de fornecer suporte à modelagem e simulação físicas.

O uso da linguagem Modelica trouxe o benefício da simulação para dentro do ambiente de MBSE, mas, em contrapartida, restringiu a linguagem SysML uma vez que limitou os componentes do Diagrama de Blocos a se comportarem exclusivamente de acordo com a teoria de objetos, sob pena de não ser possível representá-los na linguagem Modelica.

Devido à complexidade dos sistemas atuais, a integração entre partes ou entre sistemas que trabalham em regime de cooperação ocorre de forma menos direta e intuitiva, o que (CALVANO e JOHN, 2004) chamou de *Weakly Integrated Systems*. No caso do sistema analisado, existem relações de causa e efeito entre o comportamento do sistema de ultrassom e o sistema de circulação de água que não são naturalmente intuitivas. Caso haja algum defeito na bomba centrífuga que proporcione uma diminuição no fluxo de água ou mesmo um fluxo turbulento, podem ocorrer problemas no sinal do ultrassom cuja causa seria dificilmente detectada. Com a abordagem proposta, o fluxo turbulento pode ser simulado e parâmetros de velocidade de rotação de bomba passados para o COMOS® de forma a evitar distúrbios ou mesmo informar ao operador quando venham a acontecer.

Mudar a forma como se analisam os requisitos inicias de "baseada na experiência" para "baseada no comportamento do sistema" traz maior flexibilidade para a área de engenharia. Caso seja necessário alterar com profundidade os produtos em função de necessidade do mercado, o ambiente de simulação pode trazer benefícios. Se for necessário mudar o sistema de circulação de água para sistema de circulação de óleo, por exemplo, a vantagem da orientação a objetos da linguagem Modelica pode ser utilizada para se atingir a estabilidade da engenharia do sistema de forma mais rápida. Uma vez que o método baseado em modelos apresenta maior integração entre as disciplinas e componentes, a simulação do comportamento do sistema a partir de modificações em objetos pode resultar em aprendizados mais rápidos.

No caso dos modelos criados, o objeto Modelica chamado *Medium* atribuído ao comportamento da água líquida poderia ser alterado de forma a herdar o comportamento de um objeto que contenha as características do óleo. O ciclo de engenharia seria encurtado, uma vez que a curva de aprendizado (baseada em iterações) poderia ser realizada no próprio modelo virtual.

Durante a elaboração da plataforma de testes, muitos erros foram causados pela falta de maturidade das ferramentas empregadas. Problemas foram enfrentados na compatibilidade de *Plug-Ins* do Eclipse, uma vez que alguns pacotes instalados ocasionavam erros em outros.

O OpenModelica apresentou erros ao integrar objetos de disciplina elétrica com mecânica. A interação entre o motor elétrico e a bomba centrífuga não pode ser simulada em baixas rotações porque causava falha no OpenModelica. O suporte técnico da ferramenta informou que o erro foi causado por divisões por zero e que esse aspecto seria corrigido em versões futuras. Isso demostra a falta de maturidade das ferramentas utilizadas, corroborando com a ideia de que a Engenharia de Sistemas usando modelos encontra-se em estágio de pesquisa.

A integração entre os objetos abstratos e os artefatos do domínio representou um avanço no sentido de estabelecer consistência entre o ambiente de modelagem e simulação com o ambiente de realização do projeto. Mas dificuldades foram detectadas devido à falta de padronização na transferência de parâmetros entre as ferramentas utilizadas. A incapacidade do COMOS® de sincronizar componentes via XMI foi um importante limitador do projeto, uma vez que os artefatos tiveram de ser sincronizados via programa externo escrito em Java.

Nos testes realizados com o método MoDElS.iDEAS, percebeu-se maior emprego de tempo na análise dos requisitos e simulação do comportamento do sistema do que na geração de documentos para a fabricação. Esse aspecto traz benefícios, pois possibilita aos engenheiros focarem esforços na simulação e detecção de erros ou falhas no sistema antes de serem produzidos.

Entende-se que o método proposto pode ser usado com benefícios em partes do projeto críticas para o sucesso no atendimento dos requisitos. O tempo necessário para modelar componentes simples e sem impacto no funcionamento final do sistema torna o emprego do método oneroso frente aos benefícios verificados.

O repositório de domínio trouxe benefícios em termos de consistência de dados uma vez que possibilitou o armazenamento das informações de projeto de forma centralizada.

6.1 TRABALHOS FUTUROS

Frente às dificuldades enfrentadas durante o projeto, entende-se que há a necessidade de concentrar esforços na geração de uma ferramenta aberta de Engenharia de Domínio ou que as ferramentas existentes suportem padrões abertos de trocas de informações, como o XMI. Traria o benefício de facilitar a integração de objetos abstratos com os artefatos técnicos do domínio.

Além disso, entende-se que o estabelecimento de formatos abertos para a especificação de componentes traria benefícios para a Engenharia de Domínio. Padrões como CAEx e ISO1303/AP233 podem ser usados para integrar o portfólio de produtos de fornecedores à plataforma de Engenharia de Domínio. A partir dessa integração, componentes de prateleira poderiam ser adicionados ao projeto automaticamente partindo-se das especificações geradas pela simulação virtual.

Um método de MBSE utilizado de forma objetiva para projetos de sistemas de automação pode trazer benefícios durante a fase de operação e manutenção. Uma vez que o modelo físico do sistema pode ser incorporado ao sistema, modificações tardias visando tanto a melhoria operacional do sistema como a eliminação de falhas de operação podem ser simuladas com antecedência antes que a alteração efetivamente seja executada. Esse aspecto é interessante quando se trabalha com plantas industriais e processo contínuos com alto custo de interrupção.

Outro aspecto que carece de pesquisa futura diz respeito à descrição automática de objetos mecânicos do Tipo C, ou seja, genéricos, a partir de propriedades extraídas da simulação.

Por fim, entende-se que integrar o modelo físico escrito em Modelica ao sistema realizado, de forma a possibilitar a simulação do comportamento frente a mudanças pretendidas durante a fase de operação, é um interessante objeto de estudo.

REFERÊNCIAS

ACATECH. **Cyber-physical systems**: driving forces for innovations in mobility, health, energy and production. Heidelberg: Springer Verlag, 2011.

ALMEIDA, E. **RiDE - The RiSE process for domain engineering**. Saarbrücken: VDM Verlag, 2009.

BAHETI, R.; GILL, H. **Cyber-physical systems**. [S. l.]: IEEE Control Systems Society, 2011, 161-166 f. Disponivel em: <http://ieeecss.org/sites/ieeecss.org/files/documents/IoCT-FullReport.pdf>. Acesso em: 21 fev. 2012.

BENJELLOUN-TOUIMI, Z., *et al.* From physical modeling to real-time simulation: feedback on the use of modelica in the engine control development toolchain. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 763-772.

BLANCHARD, B. S.; FABRYCKY, W. J. **Systems engineering and analysis**. 5 ed. New Jersey: Prentice Hall, 2010.

BOUSKELA, D., *et al.* Modelling of uncertainties with modelica. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 673-685.

BROMAN, D.; FRITZSON, P.; FURIC, S. Types in the modelica language. In: INTERNATIONAL MODELICA CONFERENCE, 5., 2006, Vienna. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2006. p. 303-315.

BROWN, A. W.; WALLNAN, K. C. Engineering of component-based systems. In: IEEE INTERNATIONAL CONFERENCE ON ENGINEERING OF COMPLEX COMPUTER SYSTEMS, 2., 1996, Montreal. **Proceedings...** Los Alamitos: IEEE, 1996. p. 414-422.

CALVANO, C. N.; JOHN, P. Systems engineering in an age of complexity. **Systems Engineering**, New York, v. 7, n. 1, p. 25-34, Dec. 2003.

EIGNER, M.; GILZ, T.; ZAFIROV, R. **Interdisciplinary Product Development**. 2012. Disponivel em: <http://www.plmportal.org/research-in-detail/items/interdisciplinary-product-development.html>. Acesso em: 01 mai 2013.

ELSHEIKH, A., *et al.* Modelica-enabled rapid prototyping of cyber-physical energy systems via the functional mockup interface. In: IEEE WORKSHOP ON MODELING AND SIMULATION OF CYBER-PHYSICAL ENERGY SYSTEMS (MSCPES), 1., 2013, Berkeley. **Proceedings...** Berkeley: IEEE, 2013. p. 1-6.

FRENKEL, J., *et al.* Towards a benchmark suite for Modelica compilers: large models. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 143-152.

FRIEDENTHAL, S.; MOORE, A.; STEINER, R. **A practical guide to SysML**: the systems modeling language. 2 ed. Heidelberg: Morgan Kaufmann, 2011.

FRITZSON, P. **Principles of object-oriented modeling and simulation with Modelica 2.1**. New Jersey: Wiley-IEEE Press, 2004.

_____. **Introduction to modeling and simulation of technical and physical systems with Modelica**. New Jersey: Wiley-IEEE Press, 2011.

GOROD, A.; SAUSER, B.; BOARDMAN, J. System-of-Systems engineering management: a review of modern history and a path forward. **IEEE Systems Journal**, [S. l], v. 2, n. 4, p. 484-499, Dec. 2008.

GROOVER, M. P. **Automation, production systems, and computer-integrated manufacturing**. 3 ed. New Jersey: Prentice Hall Press, 2007.

GRUNDFOS. **NB 32-160.1/139 50 Hz**. 2013. Disponivel em: <http://net.grundfos.com/Appl/WebCAPS/ProductDetailCtrl?cmd=com.grundfos.webcaps.productdetail.commands.ProductDetailCommand&productnumber=98069824&freq=50&page=0&selectedRow=0&detailid=1403455905219>. Acesso em: 22 mai. 2013.

GUO, J.-S.; QIN, C.-K.; GERHARD, S. Simulation of internal combustion engine fueled by natural gas/liquefied petroleum gas-biogas blends with Modelica. **Gongcheng sheji xuebao**, v. 18, n. 1, p. 28-33.

HAKAM, I. **ModelicaML graphical modeling environment based on Eclipse MDT Papyrus**. 2011. 60 f. Tese (Mestrado em Informática e Ciência da Computação) - Department of Computer and Information Science, Linköping University, Linköping, 2011.

HASKINS, C.; FORSBERG, K. **Systems engineering handbook**: a guide for system life-cycle processes and activities. 3 ed. Seattle: INCOSE, 2011.

HENRIKSSON, D.; ELMQVIST, H. Cyber-physical systems modeling and simulation with modelica. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 502-509.

HÖGER, C. ModIM - A modelica frontend with static analysis. **Mathematical modelling**, Vienna, v. 7, n. 1, p. 1075-1080, Feb. 2012.

HOU, Y., *et al.* Shock absorber modeling and simulation based on Modelica. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 843-846.

HYBERTSON, D. W. **Model-oriented systems engineering science**: a unifying framework for traditional and complex systems. Florida: Auerbach Publications, 2009.

INCOSE. **Systems engineering primer**. 1997. Disponivel em: <http://www.incose.org/ProductsPubs/pdf/SEPrimerAIAA-INCOSE_1997-08.pdf>. Acesso em: 19 set. 2013.

JARDIN, A., *et al.* Modelling of system properties in a modelica framework. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 579-592.

JAZDI, N.; MAGA, C.; GOEHNER, P. Reusable models in industrial automation: experiences in defining appropriate levels of granularity. In: IFAC WORLD CONGRESS, 18., 2011, Milano. **Proceedings...** Milano: IFAC, 2011. p. 9145-9150.

JAZDI, N., *et al.* Mehr Systematik für den Anlagenbau und das industrielle Lösungsgeschäft-Gesteigerte Effizienz durch Domain Engineering. **at-Automatisierungstechnik**, v. 58, n. 9, p. 524-532, 2010.

KERNSCHMIDT, K., *et al.* **Possibilities and challenges of an integrated development using a combined SysML-model and corresponding domain specific models**. 2013. Disponivel em: <http://www.arscontrol.org/publications/2013-KerBarFanVog-MIM.pdf>. Acesso em: 21 set. 2013.

KERNSTINE, K. H. Inadequacies of traditional exploration methods in Systems-of-Systems simulations. **IEEE Systems Journal**, [S. l.], v. 7, n. 4, p. 528-536, Dec. 2013.

KIPPER, M. M. **Proposta de metodologia de engenharia de domínio para o desenvolvimento de sistemas de automação industrial**. 2010. 106 f. Dissertação (Mestrado em Engenharia Elétrica) - Departamento de Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2010.

KOSSIAKOFF, A., *et al.* **Systems engineering principles and practice**. 2 ed. New York: Wiley-Interscience, 2011.

LEE, E. A. Cyber-Physical Systems: design challenges. In: IEEE INTERNATIONAL SYMPOSIUM ON OBJECT ORIENTED REAL-TIME DISTRIBUTED COMPUTING (ISORC), 11., 2008, Orlando. **Proceedings...** Los Alamitos: IEEE Computer Society, 2008. p. 363-369.

LIND, I.; ANDERSSON, H. Model-based Systems Engineering for aircraft systems: how does Modelica-based tools fit? In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 856-864.

LONG, D.; SCOTT, Z. **A Primer For Model-Based Systems Engineering**. 2 ed. Raleigh: lulu.com, 2012.

LUCREDIO, D., *et al.* Designing domain architectures for Model-Driven Engineering. In: BRAZILIAN SYMPOSIUM ON SOFTWARE COMPONENTS, ARCHITECTURES AND REUSE (SBCARS), 4., 2010, Bahia. **Proceedings...** Bahia: IEEE, 2010. p. 100-109.

MAGA, C.; JAZDI, N. Interdisciplinary modularization in product line engineering: A case study. In: IEEE INTERNATIONAL CONFERENCE ON AUTOMATION QUALITY AND TESTING ROBOTICS (AQTR), 18., 2012, Cluj-Napoca. **Proceedings...** Cluj-Napoca: IEEE Computer Society, 2012. p. 179-184.

MAGA, C. R.; JAZDI, N. Concept of a domain repository for industrial automation. In: INTERNATIONAL WORKSHOP ON DOMAIN ENGINEERING, 1., 2009, Amsterdam. **Proceedings...** Stuttgart: Institute of Industrial Automation and Software Engineering (IAS), Universität Stuttgart, 2009. Disponível em: <http://www.ias.uni-stuttgart.de/forschung/veroeffentlichungen/pdf/DE-v24.pdf>.

_____. An approach for modeling variants of industrial automation systems. In: IEEE INTERNATIONAL CONFERENCE ON AUTOMATION QUALITY AND TESTING ROBOTICS (AQTR), 16., 2010, Cluj-Napoca. **Proceedings...** Cluj-Napoca: IEEE Computer Society, 2010. p. 1-6.

MAGA, C. R.; JAZDI, N.; GÖHNER, P. Requirements on engineering tools for increasing reuse in industrial automation. In: IEEE CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION (ETFA), 16., 2011, Toulouse. **Proceedings...** Toulouse: IEEE, 2011. p. 1-7. Disponível em: <http://www.scopus.com/inward/record.url?eid=2-s2.0-80655128530&partnerID=40&md5=976a99f6835c222961c304665de133c3>. Acesso em: 5 dez. 2012.

MARTIN-VILLALBA, C.; URQUIA, A.; DORMIDO, S. Development of an industrial boiler virtual-lab for control education using Modelica. **Computer applications in engineering education**, [S. l.], v. 21, n. 1, p. 36-45, mar. 2013.

MODELICA ASSOCIATION. **The Modelica language specification**. Modelica Association, 2012. Disponivel em: <https://www.modelica.org/documents/ModelicaSpec33.pdf>. Acesso em: 03 maio 2013.

_____. **The Modelica standard library**. 2013. Disponivel em: <http://www.modelica.org/libraries/Modelica>. Acesso em: 23 abr. 2013.

OBJECT MANAGEMENT GROUP. **UML infrastructure specification**. Needham: OMG, 2011a. Disponivel em: <http://www.omg.org/spec/UML/2.4.1/>. Acesso em: 25 jan. 2013.

_____. **UML superstructure specification**. Needham: OMG, 2011b. Disponivel em: <http://www.omg.org/spec/UML/2.4.1/>. Acesso em: 12 jan. 2013.

_____. **Systems modeling language**. Needham: OMG, 2012. Disponivel em: <http://www.omg.org/spec/SysML/1.3/>. Acesso em: 12 fev. 2013.

PAREDIS, C., *et al.* An overview of the SysML-Modelica transformation specification. In: INCOSE INTERNATIONAL SYMPOSIUM, 20., 2010, Chicago. **Proceedings...** Chicago: INCOSE, 2010.

POP, A.; AKHVLEDIANI, D.; FRITZSON, P. Integrated UML and modelica system modeling with ModelicaML in Eclipse. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND APPLICATIONS, 11., 2007, Cambridge. **Proceedings...** Cambridge: IASTED, 2007.

POP, A.; BĂLUȚĂ, V.; FRITZSON, P. Eclipse support for design and requirements engineering based on ModelicaML. In: SCANDINAVIAN CONFERENCE ON SIMULATION AND MODELING, 48., 2007, Göteborg. **Proceedings...** Göteborg: Linköping: Linköping University Electronic Press, 2007. p. 93-100. Disponível em: <http://www.ep.liu.se/ecp/027/ecp07027.pdf>. Acesso em: 13 fev. 2013.

PROJECT MANAGEMENT INSTITUTE. **A guide to the project management body of knowledge (PMBOK guide)**. 5 ed. Newtown: Project Management Institute, 2013.

PROß, S., *et al.* PNlib - A Modelica library for simulation of biological systems based on extended hybrid Petri nets. In: INTERNATIONAL WORKSHOP ON BIOLOGICAL PROCESSES AND PETRI NETS, 3., 2012, Hamburg. **Proceedings...** Hamburg: CEUR, 2012. p. 47-61.

RAUSCH, A., *et al.* The V-Modell XT applied–model-driven and document-centric development. In: WORLD CONGRESS FOR SOFTWARE QUALITY, 3., 2005, Munich. **Proceedings...** Munich: Technical University Munich, 2005. p. 131-138.

SAMLAUS, R., *et al.* Towards a model driven Modelica IDE. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 528-536.

SCHAMAI, W., *et al.* Towards unified system modeling and simulation with ModelicaML: modeling of executable behavior using graphical notations. In: INTERNATIONAL MODELICA CONFERENCE, 7., 2009, Como. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2009. p. 612-621.

SCHAMAI, W., *et al.* ModelicaML value bindings for automated model composition. In: SYMPOSUIM ON THEORY OF MODELING AND SIMULATION-DEVS 2012. **Proceedings...** International Society for Computer Simulation, 2012. p. 31.

SECCHI, C., *et al.* Object-oriented modeling of complex mechatronic components for the manufacturing industry. **IEEE Transactions on Mechatronics**, [S. l.], v. 12, n. 6, p. 696-702, 2007.

SIEMENS. **COMOS class documentation COMOS_dll**. Nürnberg: Siemens AG, 2012a. Disponivel em: <http://www.automation.siemens.com/mcms/industrial-automation-systems-simatic/en/manual-overview/comos-manuals/pages/comos-10-0--sp1.aspx>. Acesso em: 20 abr. 2013.

_____. **COMOS platform getting started**. Nürnberg: Siemens AG, 2012b. Disponivel em: <http://www.automation.siemens.com/mcms/industrial-automation-systems-simatic/en/manual-overview/comos-manuals/pages/comos-10-0--sp1.aspx>. Acesso em: 20 abr. 2013.

_____. **COMOS process P&ID**. Nürnberg: Siemens AG, 2012c. Disponivel em: <http://www.automation.siemens.com/mcms/industrial-automation-systems-

simatic/en/manual-overview/comos-manuals/pages/comos-10-0--sp1.aspx>. Acesso em: 20 abr. 2013.

SONNTAG, C.; HÜFNER, M. On the connection of equation-and automata-based languages: transforming the compositional interchange format to Modelica. In: IFAC WORLD CONGRESS, 18., 2011, Milano. **Proceedings...** Milano: IFAC, 2011. p. 12515-12520.

STARK, J. **Product life cycle management**. 2 ed. New York: Springer, 2011.

STROBEL, M., *et al.* The OnWind Modelica Library for offshore wind turbines: implementation and first results. In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 603-609.

SUN, Y.; VOGEL, S.; STEUER, H. Combining advantages of specialized simulation tools and Modelica models using Functional Mock-up Interface (FMI). In: INTERNATIONAL MODELICA CONFERENCE, 8., 2011, Dresden. **Proceedings...** Linköping: Linköping University Electronic Press, Linköpings Universitet, 2011. p. 491-494.

TANG, J., *et al.* Cyber-Physical Systems modeling method based on Modelica. In: IEEE INTERNATIONAL CONFERENCE ON SOFTWARE SECURITY AND RELIABILITY COMPANION (SERE-C), 2012, Gaithersburg. **Proceedings...** Gaithersburg: Conference Publishing Services, 2012. p. 188-191.

WEHRMEISTER, M.; PEREIRA, C.; RAMMIG, F. Aspect-oriented Model-driven engineering for embedded systems applied to automation systems. **IEEE Transactions on industrial informatics**, [S. l.], v. 9, n. 4, p. 2373-2386, Nov. 2013.

WEILKIENS, T. **Systems engineering with SysML/UML**: modeling, analysis, design. Burlington: Morgan Kaufmann, 2011.

WETTER, M.; ZUO, W.; NOUIDUI, T. S. Modeling of heat transfer in rooms in the Modelica Buildings Library. In: CONFERENCE OF INTERNATIONAL BUILDING PERFORMANCE SIMULATION ASSOCIATION, 12., 2013, Sydney. **Proceedings...** Sydney: IBPSA, 2012. p. 1096-1103. Disponível em: <http://www.escholarship.org/uc/item/95p4c6c5>.

WOLF, W. **Cyber-Physical Systems**. 2009. Disponivel em: <http://www.jiafuwan.net/download/cyber_physical_systems.pdf>. Acesso em: 10. abr. 2013.

ZHENGYIN, S.; SHENGLIN, Z.; SHAN-AN, Z. An internet-based electrical engineering virtual lab: using Modelica for unified modeling. In: IEEE INTERNATIONAL

CONFERENCE ON COMMUNICATION SOFTWARE AND NETWORKS (ICCSN), 3., 2011, Xian. **Proceedings...** Xian: IEEE, 2011. p. 555-559.