



MINISTÉRIO DA CIÊNCIA E TECNOLOGIA

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m19@80/2010/06.15.00.51-MAN

MANUAL DO SOFTWARE SATBUDGETS

Bruno Bustamante Ferreira Leonor

Walter Abrahão dos Santos

Stephan Stephany

Protótipo de software baseado em conhecimento para auxílio ao projeto conceitual de satélites.

URL do documento original:

<<http://urlib.net/8JMKD3MGP7W/37M5APB>>

INPE

São José dos Campos

2010

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE

Gabinete do Diretor (GB)

Serviço de Informação e Documentação (SID)

Caixa Postal 515 - CEP 12.245-970

São José dos Campos - SP - Brasil

Tel.:(012) 3208-6923/6921

Fax: (012) 3208-6919

E-mail: pubtc@sid.inpe.br

CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO DA PRODUÇÃO INTELLECTUAL DO INPE (RE/DIR-204):**Presidente:**

Dr. Gerald Jean Francis Banon - Coordenação Observação da Terra (OBT)

Membros:

Dr^a Inez Staciarini Batista - Coordenação Ciências Espaciais e Atmosféricas (CEA)

Dr^a Maria do Carmo de Andrade Nono - Conselho de Pós-Graduação

Dr^a Regina Célia dos Santos Alvalá - Centro de Ciência do Sistema Terrestre (CST)

Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)

Dr. Ralf Gielow - Centro de Previsão de Tempo e Estudos Climáticos (CPT)

Dr. Wilson Yamaguti - Coordenação Engenharia e Tecnologia Espacial (ETE)

Dr. Horácio Hideki Yanasse - Centro de Tecnologias Especiais (CTE)

BIBLIOTECA DIGITAL:

Dr. Gerald Jean Francis Banon - Coordenação de Observação da Terra (OBT)

Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)

Deicy Farabello - Centro de Previsão de Tempo e Estudos Climáticos (CPT)

REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:

Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)

Yolanda Ribeiro da Silva Souza - Serviço de Informação e Documentação (SID)

EDITORAÇÃO ELETRÔNICA:

Vivéca Sant´Ana Lemos - Serviço de Informação e Documentação (SID)



MINISTÉRIO DA CIÊNCIA E TECNOLOGIA

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m19@80/2010/06.15.00.51-MAN

MANUAL DO SOFTWARE SATBUDGETS

Bruno Bustamante Ferreira Leonor

Walter Abrahão dos Santos

Stephan Stephany

Protótipo de software baseado em conhecimento para auxílio ao projeto conceitual de satélites.

URL do documento original:

<<http://urlib.net/8JMKD3MGP7W/37M5APB>>

INPE

São José dos Campos

2010

Resumo

Satélites estão tornando-se cada vez mais complexos, tornando questões técnicas de projeto significativamente relevantes. A crescente complexidade destes sistemas torna a atividade de engenharia de requisitos mais importante e mais difícil. Caso não se atinja um nível adequado de conformidade, vários problemas de projeto podem ocorrer, tais como falhas, custos excedidos e atraso. A solução proposta é apoiar a fase de projeto conceitual de satélite com base no reuso e na integração da informação, tratando melhor a natureza interdisciplinar do domínio através da adoção de uma engenharia dirigida por modelo (MDE - *Model Driven Engineering*). Foi escolhida para implantar este enfoque a linguagem SysML (*Systems Modeling Language*). Neste contexto foi desenvolvida uma ferramenta de software baseado em conhecimento, denominada SatBudgets, que pode ser aplicada a qualquer projeto de satélite, sendo realizado um estudo de caso para o projeto do satélite universitário ITASAT, atualmente em desenvolvimento pelo INPE, ITA e algumas universidades brasileiras.

LISTA DE FIGURAS

	<u>Pág.</u>
2.1 Diagrama de Taxonomia SysML	4
2.2 Diagrama de Blocos do ITASAT	5
2.3 Diagrama de Requisitos do Subsistema de Potência	6
2.4 Diagrama Simplificado de Casos de Uso do ITASAT	7
3.1 Tela Principal TOPCASED	9
3.2 IDE Eclipse	10
3.3 Estrutura de uma Regra DROOLS	11
3.4 Atributos de uma Regra DROOLS	12
3.5 Regra Convertida para DROOLS	13
3.6 Fluxo de funcionamento da API JasperReports	14
3.7 Ferramenta iReport	15
3.8 Exemplo de Relatório	16
4.1 Tela Principal da SatBudgets	17
4.2 Fluxo de funcionamento da SatBudgets	18
4.3 Seleção do modelo SysML	19
4.4 Selecionando os balanços	19
4.5 Modos de visualização	19
4.6 Barra de progresso	20
4.7 Relatório gerado	20

SUMÁRIO

	<u>Pág.</u>
1 Introdução	1
2 SysML	3
2.1 Diagramas SysML	3
2.2 Uso proposta da SysML	4
3 Ferramentas Utilizadas	8
3.1 Modelagem	8
3.2 Desenvolvimento	9
3.2.1 Ambiente de Desenvolvimento - Eclipse	9
3.2.2 Manipulação do Modelo SysML - JDOM	9
3.2.3 Regras de Negócio - DROOLS	10
3.2.3.1 Atributos da Regra	11
3.2.3.2 LHS (when) - Elementos Condicionais	11
3.2.3.3 RHS (then) - Ações	12
3.2.4 Geração do Relatório	13
3.2.4.1 JasperReports	13
3.2.4.2 iReport	14
3.2.4.3 JFreeChart	15
4 SatBudgets	17
4.1 Uso da ferramenta SatBudgets	18
5 Dicionário de Dados	21
6 Conclusão	22
REFERÊNCIAS BIBLIOGRÁFICAS	23

1 Introdução

Os sistemas modernos estão tornando-se cada vez mais complexos, por este motivo projetos conceituais estão ganhando cada vez mais espaço no desenvolvimento de um produto, podendo-se assim especificar, analisar, projetar e verificar sistemas sem a necessidade de produção de uma única peça. (LEONOR et al., 2009)

O Projeto Conceitual é a fase inicial do processo de projeto de um sistema e exige a aplicação de conhecimento no domínio do produto. Essa fase deve ser sistematizada, ou seja, o sistema deve ser decomposto em subsistemas e/ou componentes, para que se obtenha uma maior eficiência e velocidade no desenvolvimento. (ALMEIDA, 2000)

Devido à grande concorrência do mercado, a sistematização de sistemas é necessidade premente para os dias de hoje.

Para projeto de satélites isso não é diferente, levando em consideração a grande quantidade de recursos, tanto humanos quanto financeiros, envolvidos em sua produção e também os benefícios trazidos pelos mesmos à população. Um bom planejamento é essencial, pois evita erros que podem comprometer todo o empreendimento. (SANTOS et al., 2009)

A falta de ferramentas adequadas para auxiliar na fase conceitual de projetos foi a principal motivação para o desenvolvimento desta ferramenta, que tem como objetivo mostrar a viabilidade do desenvolvimento deste tipo de ferramental.

Uma nova abordagem chamada de MDE (*Model-Driven Engineering*) (SCHMIDT, 2006) pode auxiliar na fase de projeto conceitual de sistemas permitindo o reuso e integração da informação. Esta abordagem foca a criação de modelos ou abstrações maximizando a compatibilidade entre sistemas e simplificando o processo de projeto. A utilização de uma metodologia MDE eficaz de trabalho envolve especialistas no domínio, plataformas, teste, verificação, validação, etc. Para isto, MDE requer o uso de uma linguagem de descrição para a arquitetura do sistema.

Para este trabalho em particular, foi adotada a linguagem SysML (*Systems Modeling Language*) (OMG, 2009a), uma linguagem proposta pela OMG (*Object Ma-*

nagement Group) para se tornar uma linguagem padrão na modelagem de sistemas que surgiu como complemento a UML (*Unified Modeling Language*) (OMG, 2009b). Assim como a MDE, a SysML também não é empregada para descrição de arquiteturas de satélites.

É apresentado neste trabalho um protótipo de software baseado em conhecimento, denominado SatBudgets, criado para auxiliar na fase conceitual de projeto de satélite fazendo uso de diagramas SysML. O software será aplicado na fase conceitual do projeto de um sistema de satélite universitário, o ITASAT, iniciado no ano de 2005, atualmente sendo desenvolvido pelo INPE (Instituto Nacional de Pesquisas Espaciais), ITA (Instituto Tecnológico de Aeronáutica) e algumas universidades públicas brasileiras. Além de funcionalidades experimentais o ITASAT contará com um transmissor para transmissão de dados coletados das PCDs (Plataformas de Coleta de Dados) distribuídas por todo o país, representando um importante reforço para o Sistema Nacional de Coleta de Dados.

SatBudgets será aplicado para auxiliar na geração de balanços (mecânico, elétrico, entre outros) de um sistema de satélite. O conhecimento inserido na ferramenta para a geração do balanço foi adquirido minimamente através de captura de regras de decisão de projeto. Esta atividade faz parte do escopo de Engenharia do Conhecimento, que integra o conhecimento do especialista a um ambiente computacional. (LEONOR, 2010)

2 SysML

Muitos processos de engenharia de sistemas tendem a ser documentados e empregam uma combinação de várias técnicas de diagramas que são muitas vezes imprecisas e incoerentes. Com objetivo de atingir esta meta, o OMG e o INCOSE ([INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING, 2009](#)) sugeriram a criação de uma ADL (*Architecture Description Language*) padrão e de uso geral dentro da Engenharia de Sistemas, chamada SysML, como extensão da UML, que possibilita a modelagem de qualquer tipo de domínio, padroniza a comunicação entre as pessoas e grupos envolvidos no projeto, melhora a gestão do ciclo de vida do produto e captura artefatos da engenharia de sistemas.

SysML é uma linguagem gráfica de modelagem que apoia a análise, especificação, projeto, verificação e validação de sistemas complexos ([FRIEDENTHAL et al., 2008](#)).

Como é derivada da UML, SysML possibilita a criação de modelos de sistemas orientado a objetos, permitindo expressar tanto a estrutura como o funcionamento do sistema.

2.1 Diagramas SysML

A SysML reutiliza um subconjunto da UML e acrescenta alguns novos diagramas especificamente concebidos para apoiar a engenharia de sistemas.

São nove os diagramas SysML, a saber: (1) Diagrama de Requisitos; (2) Diagrama de Atividades; (3) Diagrama de Sequência; (4) Diagrama de Máquina de Estados; (5) Diagrama de Casos de Uso; (6) Diagrama de Definição de Blocos; (7) Diagrama de Bloco Interno; (8) Diagrama Paramétrico e (9) Diagrama de Pacotes.

Os diagramas SysML são portanto classificados em:

- **Diagramas Estruturais:** Diagrama de Blocos, Diagrama de Blocos Internos, Diagrama de Pacotes e Diagrama de Parâmetros;
- **Diagramas Comportamentais:** Diagrama de Atividades, Diagrama de Sequência, Diagrama de Máquina de Estados e Diagrama de Casos de Uso, e

- **Diagramas Transversais (*Cross-cutting Constructs*):** Diagrama de Requisitos.

Uma organização básica dos diagramas SysML e sua relação com a UML são apresentadas na Figura 2.1.

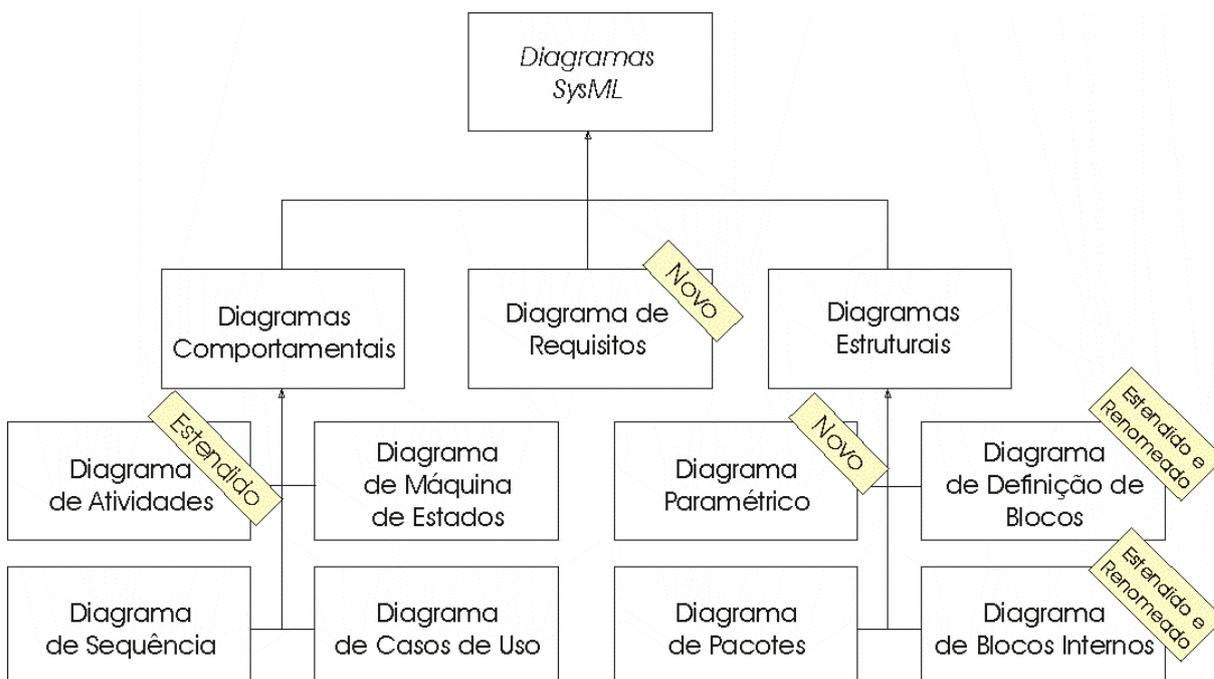


Figura 2.1 - Diagrama de Taxonomia SysML
 Fonte: Weikiens (2007)

2.2 Uso proposta da SysML

Um estudo de caso aplicado ao satélite ITASAT será descrito nesta seção, onde serão apresentados alguns exemplos dos diagramas SysML utilizados para modelagem do sistema. A linguagem adotada nestes diagramas foi o Inglês devido à documentação do projeto ITASAT assim requerer.

A Figura 2.2 representa um exemplo de diagrama de blocos de uma das versões arquiteturais do ITASAT. Nesta Figura são mostrados os blocos definidos para o ITASAT e os relacionamentos entre eles, além de representar a utilização de

pacotes, tal como o pacote do subsistema ACDH (do inglês, *Attitude Control and On-Board Data Handling*), que inclui outros blocos utilizados no sistema.

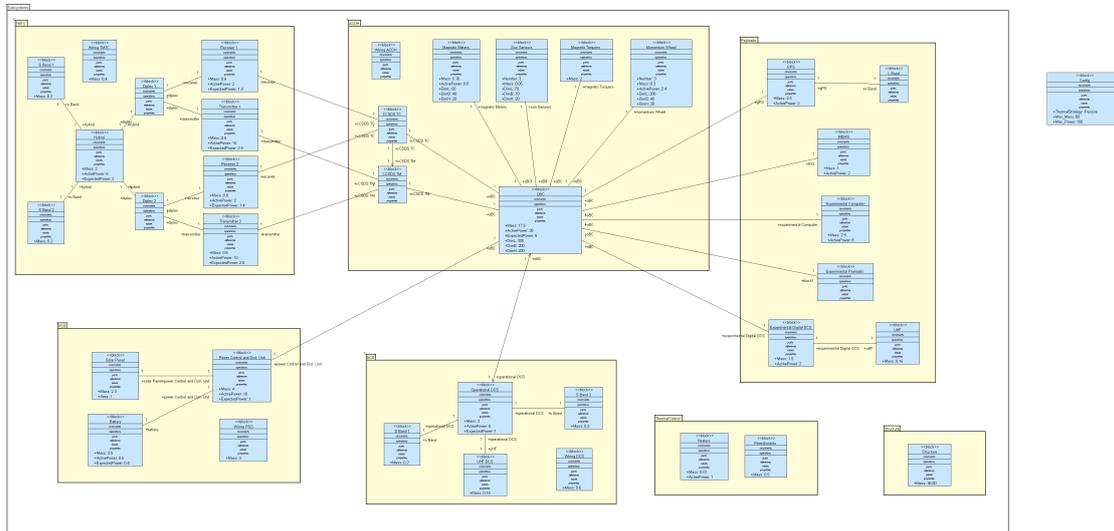


Figura 2.2 - Diagrama de Blocos do ITASAT

A Figura 2.3 exemplifica um diagrama de requisitos e mostra alguns relacionamentos possíveis dentro do diagrama. O diagrama apresentado é para o subsistema de potência do ITASAT. A Figura apresenta o relacionamento de requisitos com casos de uso, casos de teste, blocos e outros requisitos, além de apresentar uma maneira de rastrear requisitos.

Para este exemplo, pode ser descrito o relacionamento entre o requisito *Power Supply Requirements* e os casos de uso *Power Supply Functions* e *Test Power Supply Functions*. O caso de uso *Power Supply Functions* está relacionado com o requisito através de um relacionamento «satisfy», o que indica que este caso de uso deve satisfazer a implementação do requisito e o caso de uso *Test Power Supply Functions* está relacionado através de um relacionamento «verify», indicando que este caso de uso irá verificar se o requisito é satisfeito.

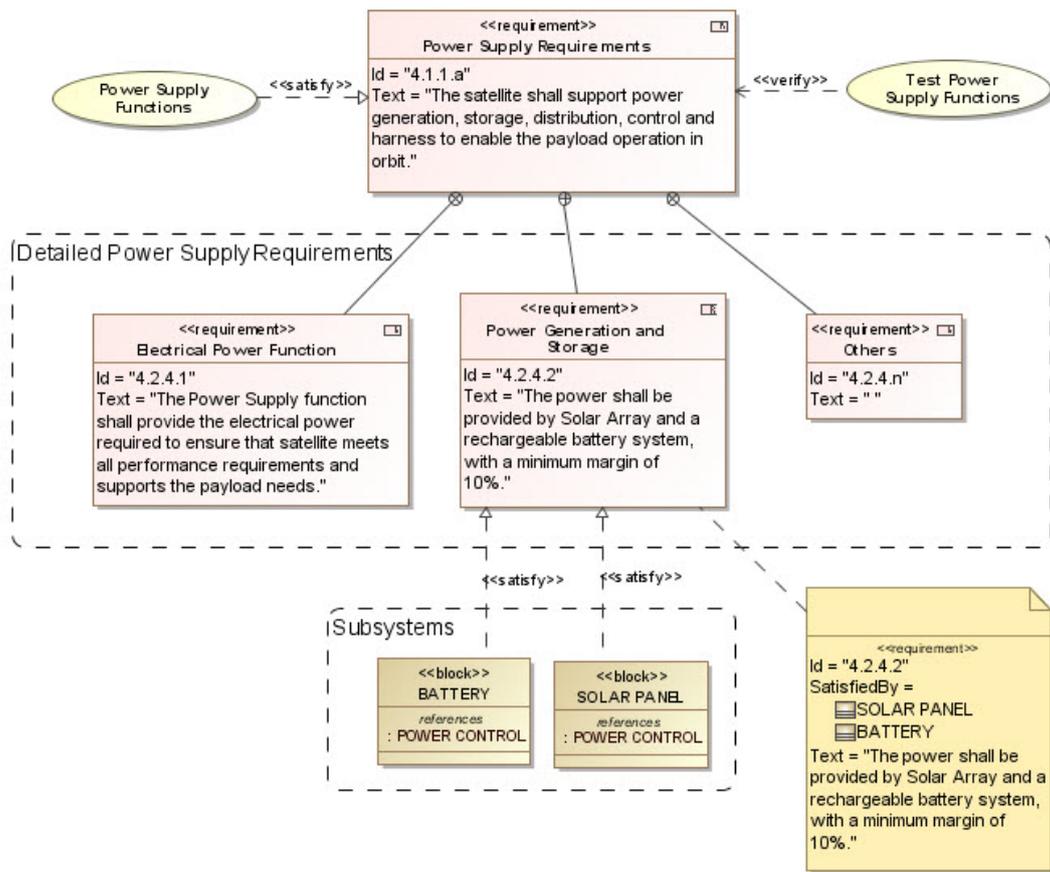


Figura 2.3 - Diagrama de Requisitos do Subistema de Potência

Outros tipos de relacionamentos em SysML que envolvem requisitos são: (1) Containment (\oplus); (2) Derivação ($\ll deriveReq \gg$); (3) Satisfação ($\ll satisfy \gg$); (4) Verificação ($\ll verify \gg$); (5) Refinamento ($\ll refine \gg$); (6) Rastro ($\ll trace \gg$) e (7) Cópia ($\ll copy \gg$).

Um exemplo de diagrama de casos de uso é apresentado na Figura 2.4 onde alguns atores, casos de uso e seus relacionamentos são listados. No diagrama apresentado, um exemplo pode ser descrito para o ator *Satellite Control Center* que por sua vez dispara vários casos de uso, entre eles o caso de uso *On-Board Management Functions*, que ao ser disparado irá disparar outros três casos devido ao seu tipo de relacionamento ($\ll include \gg$). Existe também a possibilidade do caso de uso *On-Board Management Functions* invés de ser disparado pelo ator *Satellite Control Center* ser disparado pelo caso de uso *Satellite Commis-*

oning and OrbOps através do relacionamento «extend».

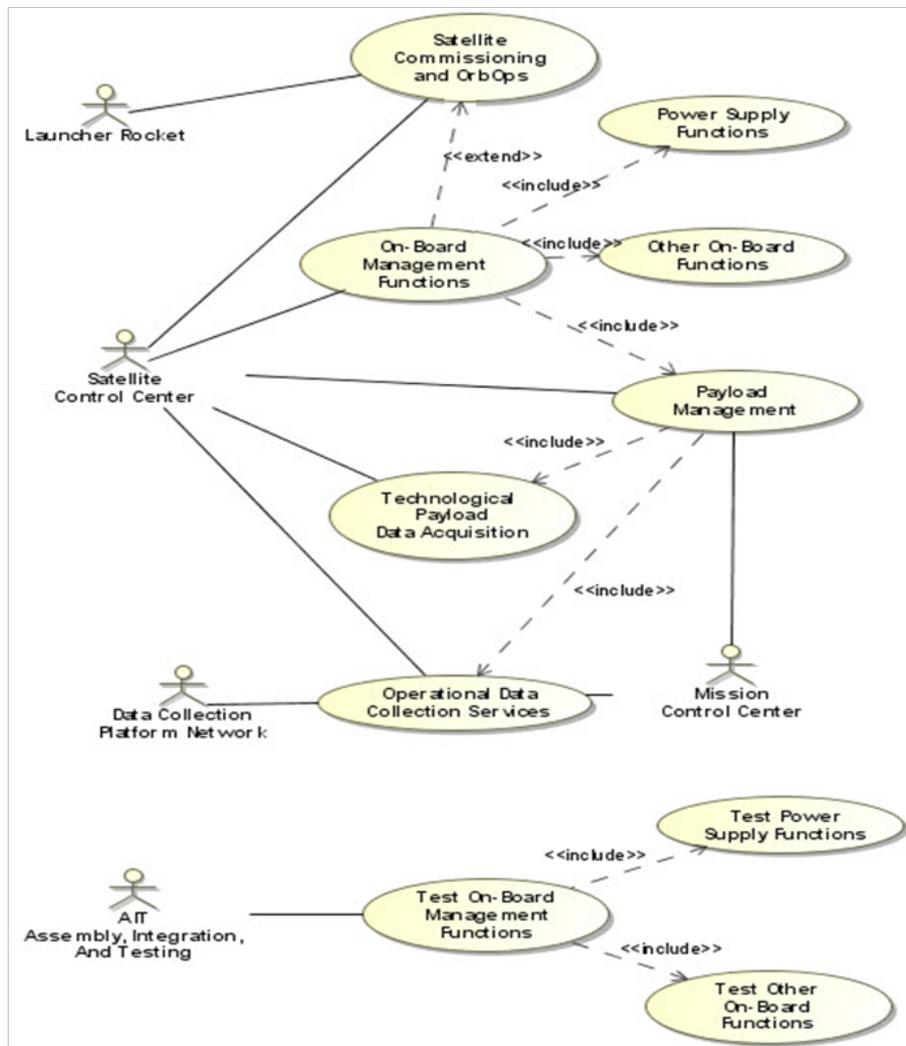


Figura 2.4 - Diagrama Simplificado de Casos de Uso do ITASAT

3 Ferramentas Utilizadas

O protótipo da ferramenta SatBudgets foi desenvolvido na linguagem Java, fazendo uso somente de ferramentas e APIs (*Application Programming Interface*) de livre distribuição, sendo este um dos requisitos para o desenvolvimento da ferramenta.

Existem outras alternativas às ferramentas escolhidas, sendo estas tomadas por serem as mais convenientes para o projeto. O conhecimento prévio de utilização e grande comunidade de usuários foram alguns dos fatores que levaram a estas escolhas, a saber:

- (1) TopCased - diagramação SysML;
- (2) Eclipse - IDE de desenvolvimento;
- (3) JDOM - manipulação de arquivos XML;
- (4) DROOLS - gerenciador de regras de negócio;
- (5) JasperReports - API para geração de relatório,
- (6) iReports - designer de relatório e
- (7) JFreeChart - API para geração de gráficos.

As ferramentas escolhidas serão descritas e alguns exemplos de utilização serão apresentados quando possível.

3.1 Modelagem

Para a modelagem do sistema foi escolhida a ferramenta TOPCASED. Esta ferramenta possibilita a modelagem de sistemas fazendo uso da SysML (OMG, 2009a), uma linguagem de descrição de arquiteturas (ADL) de uso geral que possibilita a modelagem de qualquer tipo de sistema, desde os mais simples até os mais complexos. A SysML é uma extensão da UML (*Unified Modeling Language*) sugerida pela OMG (*Object Management Group*) para possibilitar a modelagem de qualquer tipo de sistema, seja ele hardware ou software.

TOPCASED baseia-se no Galileo (Eclipse 3.5) e requer uma JVM 1.5 (*Java Virtual Machine*). Seu download pode ser feito como um aplicativo autônomo ou pode ser instalado diretamente através do gerenciador de instalação do Eclipse (LESCOT, 2009).

A Figura 3.1 apresenta a tela principal da ferramenta TOPCASED, onde é de-

monstrada a modelagem de um diagrama de blocos para o projeto do satélite ITASAT.

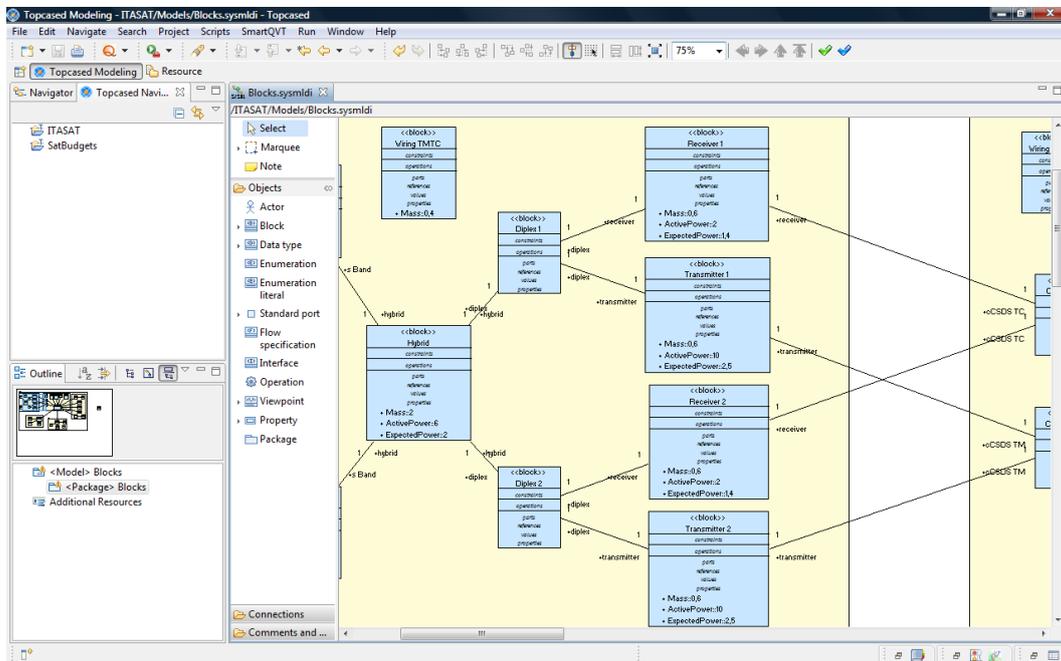


Figura 3.1 - Tela Principal TOPCASED

3.2 Desenvolvimento

3.2.1 Ambiente de Desenvolvimento - Eclipse

O Eclipse (BURNETTE, 2005) é uma IDE (*Integrated Development Environment*) desenvolvida em Java, com código aberto para a construção de programas de computador. O projeto Eclipse foi iniciado na IBM que desenvolveu a primeira versão do produto e doou-o como software livre para a comunidade.

A Figura 3.2 apresenta a tela principal da IDE Eclipse com trecho de código da implementação da ferramenta SatBudgets.

3.2.2 Manipulação do Modelo SysML - JDOM

Para a manipulação do modelo SysML foi utilizada a API JDOM (JDOM, 2009), uma API de código aberto baseado em Java que cria um modelo de objeto do documento XML (*Extensible Markup Language*). Esta API fornece uma forma

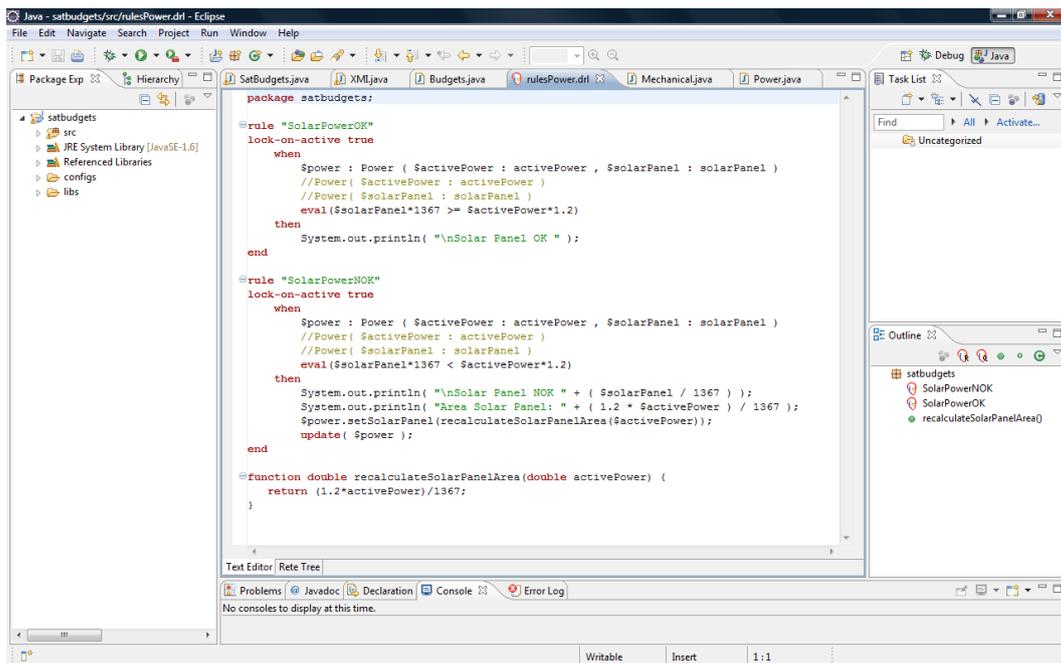


Figura 3.2 - IDE Eclipse

fácil e eficiente de representar este documento para manipulação.

Utilizando as classes da JDOM foram criadas as classes XMI, Parser e Dispatcher. Estas classes são responsáveis pela manipulação do modelo SysML do satélite dentro da ferramenta SatBudgets.

A classe XMI é responsável pela abertura e leitura do modelo. Já a classe Parser é responsável por extrair as propriedades modeladas através da busca por tokens definidos na linguagem SysML. A classe Dispatcher realiza a atribuição das propriedades extraídas do modelo para processamento das regras.

3.2.3 Regras de Negócio - DROOLS

Para o processamento das regras de negócio foi adotado o framework Drools. Drools é um mecanismo baseado em regras que foi lançado pela Codehaus e posteriormente adotado como um projeto do JBoss, conhecido também como JBoss Rules (JBoss, 2009).

É um sistema de gerenciamento de regras de negócios (BRMS - *Business Rules Management System*) que tem como um de seus principais objetivos separar as

regras de negócios dos códigos da aplicação. Com isso, Drools permite atualizações das regras sem a necessidade de recompilação do sistema.

Em Drools, o domínio é representado através das classes do sistema e as necessidades de uso do sistema geram as regras. Uma regra tem uma ou mais condições (ou fatos), que levam a uma ou mais ações (ou conseqüências). As regras não podem ser pensadas como um comando de IF-THEN-ELSE, pois Drools trabalha apenas com condições verdadeiras. Uma regra Drools é tratada em uma memória interna chamada de WorkingMemory. A Figura 3.3 representa a estrutura de uma regra Drools. A regra deve ter um nome, único dentro de um pacote de regras. Atributos são opcionais. O LHS da regra segue a palavra-chave `when`, o RHS segue a palavra-chave `then`. Uma regra é finalizada com a palavra-chave `end`.

```
rule "<nome>"
  <atributo>
when
  <condições>
then
  <ações>
end
```

Figura 3.3 - Estrutura de uma Regra DROOLS

3.2.3.1 Atributos da Regra

Atributos da regra fornecem uma forma declarativa para influenciar o seu comportamento. A Figura 3.4 apresenta um diagrama com os possíveis atributos que podem ser utilizados em uma regra.

3.2.3.2 LHS (when) - Elementos Condicionais

LHS (do inglês *Left Hand Side*) é o nome comum para a parte condicional da regra. Ela consiste de zero ou mais elementos condicionais. Se o LHS é deixado vazio, ele é re-escrito como `eval(true)`, o que significa que essa condição é sempre verdadeira.

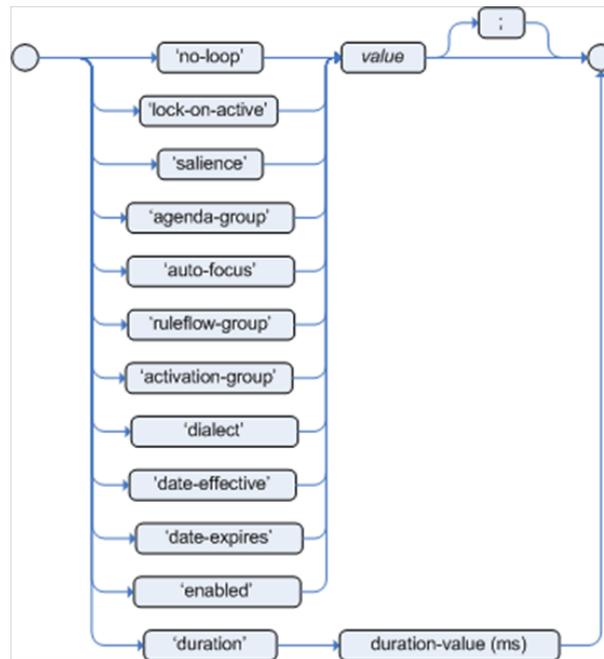


Figura 3.4 - Atributos de uma Regra DROOLS

3.2.3.3 RHS (then) - Ações

RHS (do inglês *Right Hand Side*) é o nome comum para a parte de ação ou consequência da regra, esta parte pode conter uma lista de ações para serem executadas. A principal proposta da RHS é inserir e modificar as informações existentes na **WorkingMemory**.

O trecho a seguir descreve um exemplo de regra a ser convertida para Drools.

Rule Description:

1. FOR each equipment E_j in the first column and its power consumption P_j DO:
 - TotalPower = summation of $(E_j * P_j)$ for j in 1 ..n
2. SolarPower = (Solar Panel Area) * 1.367Watt m2
3. IF SolarPower greater or equal to $(1.2 * \text{TotalPower})$ THEN
 - SolarPowerOK is TRUE
 - ELSE
 - SolarPowerOK is FALSE
4. IF SolarPowerOK is FALSE THEN
 - SolarPanelArea = $(1.2 * \text{TotalPower}) / 1367\text{Watt m2}$

5. Return (SolarPanelArea) , (SolarPowerOK)

A regra citada é referente ao subsistema de energia do satélite e diz que o painel solar do satélite deve gerar uma energia superior a 20% da energia consumida pelo satélite. Caso o painel solar não seja capaz de gerar essa energia, uma nova área deve ser calculada para que o painel solar possa atender a este requisito. A Figura 3.5 apresenta a mesma regra convertida para DROOLS.

```
package satbudgets;

rule "SolarPowerOK"
lock-on-active true
when
    $power : Power ( $activePower : activePower , $solarPanel : solarPanel )
    eval($solarPanel*1367 >= $activePower*1.2)
then
    System.out.println( "\nSolar Panel OK " );
end

rule "SolarPowerNOK"
lock-on-active true
when
    $power : Power ( $activePower : activePower , $solarPanel : solarPanel )
    eval($solarPanel*1367 < $activePower*1.2)
then
    System.out.println( "\nSolar Panel NOK " + ( $solarPanel / 1367 ) );
    System.out.println( "Area Solar Panel: " + ( 1.2 * $activePower ) / 1367 );
    $power.setSolarPanel(recalculateSolarPanelArea($activePower));
    update( $power );
end

function double recalculateSolarPanelArea(double activePower) {
    return (1.2*activePower)/1367;
}
```

Figura 3.5 - Regra Convertida para DROOLS

3.2.4 Geração do Relatório

Para a geração do relatório de balanço foram utilizadas as APIs JasperReports e JFreeChart, sendo o relatório projetado na ferramenta iReport.

3.2.4.1 JasperReports

A biblioteca JasperReports é uma poderosa e flexível ferramenta de geração de relatório que tem a capacidade de oferecer rico conteúdo sobre a tela, para a impressora ou em PDF, HTML, CSV, XLS, RTF ou arquivos XML (DANCIU, 2002).

O JasperReports organiza dados de acordo com o *design* do relatório definido em um arquivo XML. Estes dados podem vir de várias fontes de dados, incluindo dados relacionais, coleções ou matrizes de objetos Java.

O relatório é gerado através da criação de um modelo XML que contém o layout do relatório. Este modelo XML do relatório é compilado utilizando a API JasperReports, o que irá resultar em um arquivo com a extensão .jasper que contém toda a formatação do relatório e como as colunas se relacionam com os dados. O ciclo para geração do relatório é concluído com a compilação dos dados que irão compor o relatório com o arquivo .jasper criado pelo modelo do relatório. A Figura 3.6 apresenta este fluxo.

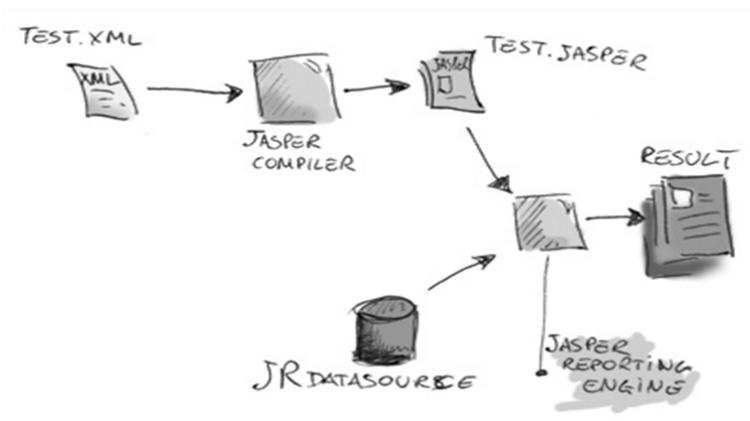


Figura 3.6 - Fluxo de funcionamento da API JasperReports

3.2.4.2 iReport

O iReport é um *designer* de relatório de código aberto para o JasperReports, disponível para todos os principais sistemas operacionais sob a GNU *General Public License*. O iReport possibilita a criação de layouts complexos contendo gráficos, imagens, sub-relatórios, tabelas de referência cruzada e muito mais (JASPERSOFT, 2009).

A Figura 3.7 apresenta a tela principal de ferramenta iReport onde está sendo projetado o relatório que será utilizado na ferramenta SatBudgets.

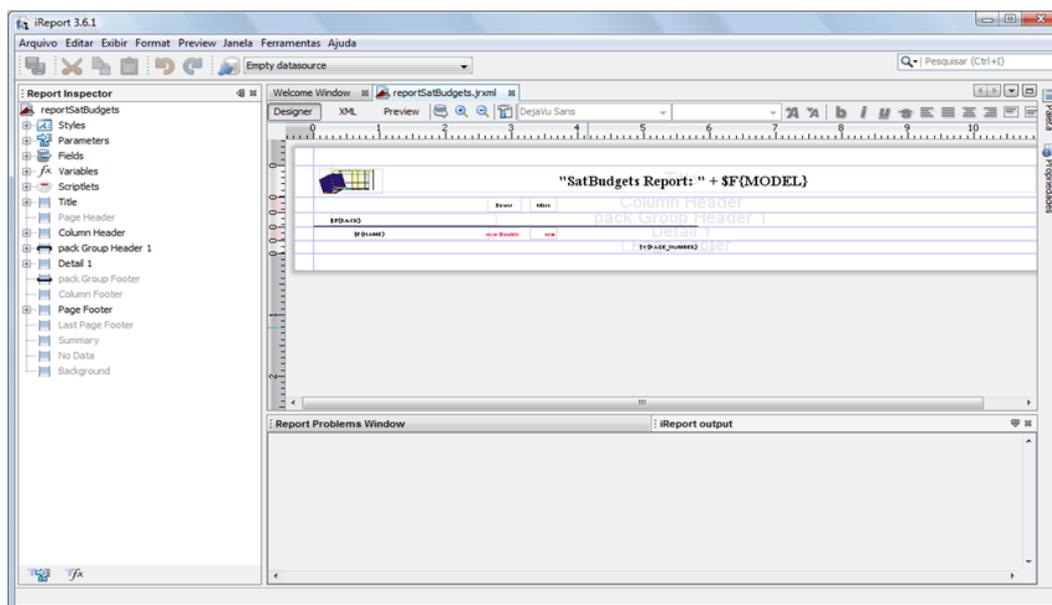


Figura 3.7 - Ferramenta iReport

3.2.4.3 JFreeChart

JFreeChart é uma consistente e bem documentada API para criação de gráficos que suporta uma ampla gama de tipos de gráfico (GILBERT, 2004).

Suporta muitos tipos de saída, incluindo componentes Swing, arquivos de imagem (JPEG e PNG), e gráficos vetoriais (PDF, EPS e SVG)

Um exemplo de relatório gerado utilizando as API JasperReports e JFreeChart é apresentado na Figura 4.7.

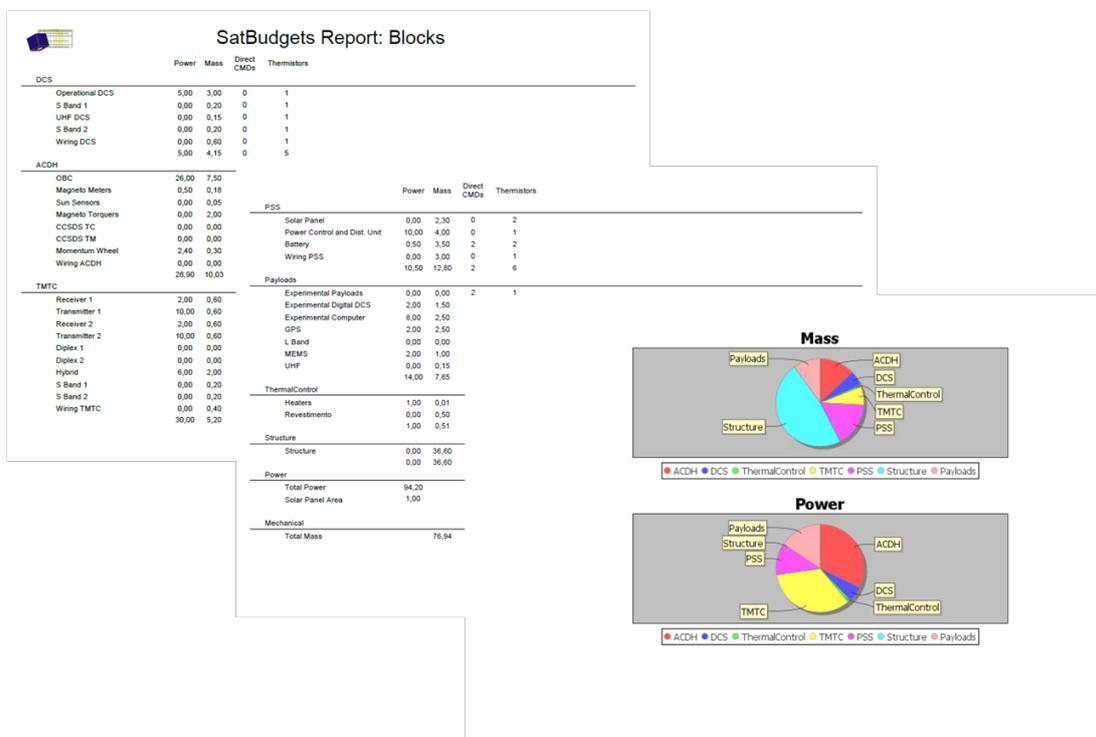


Figura 3.8 - Exemplo de Relatório

4 SatBudgets

SatBudgets é o protótipo de um software baseado em conhecimento concebido para auxiliar na fase conceitual de um projeto de satélite gerando automaticamente balanços (mecânico, elétrico, entre outros) para o modelo SysML referente a uma dada configuração de arquitetura de satélite. Um estudo de caso foi aplicado fazendo uso de dados do projeto do satélite ITASAT.

O ITASAT é um satélite universitário iniciado no ano de 2005, atualmente sendo desenvolvido pelo INPE (Instituto Nacional de Pesquisas Espaciais), ITA (Instituto Tecnológico de Aeronáutica) e algumas universidades públicas brasileiras. Além de funcionalidades experimentais o ITASAT contará com um transmissor para transmissão de dados coletados das PCDs (Plataformas de Coleta de Dados) distribuídas por todo o país, representando um importante reforço para o Sistema Nacional de Coleta de Dados.

A Figura 4.1, apresenta a tela principal do protótipo da ferramenta SatBudgets. No campo Input Files é especificado qual o modelo que servirá de entrada para o processamento, no campo Budgets, devem ser selecionados quais balanços devem ser calculados e, no campo Output File Format, o usuário seleciona qual a forma de apresentação do relatório de balanços, que pode ser o visualizador da ferramenta ou salvo no formato PDF, XLS ou HTML. Existe ainda uma barra de progresso que indica o andamento das tarefas e um botão GO para dar início ao processamento.

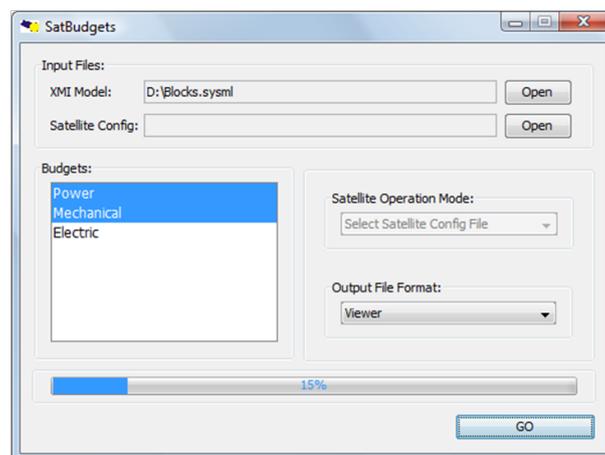


Figura 4.1 - Tela Principal da SatBudgets

A ferramenta SatBudgets foi desenvolvida na linguagem Java, fazendo uso somente de ferramentas e APIs (Application Programming Interface) de livre distribuição, sendo este um dos requisitos para o desenvolvimento da ferramenta.

A Figura 4.2, apresenta o fluxo de funcionamento da ferramenta SatBudgets. O modelo SysML (Systems Modeling Language), mais especificamente o diagrama de blocos, do satélite servirá de entrada para a ferramenta SatBudgets. A ferramenta irá abrir o arquivo do modelo através da classe OPEN. A classe PARSER é responsável por encontrar dentro da estrutura do arquivo os tokens definidos e extrair seu valor. A classe DISPATCHER irá atribuir cada um desses valores extraídos ao processamento. Estas classes foram implementadas fazendo uso da API JDOM, uma API Java para manipulação de arquivos XML. As regras de negócio serão aplicadas sobre os parâmetros extraídos da configuração de arquitetura de satélite e será gerado um relatório contendo o balanço para a dada configuração.

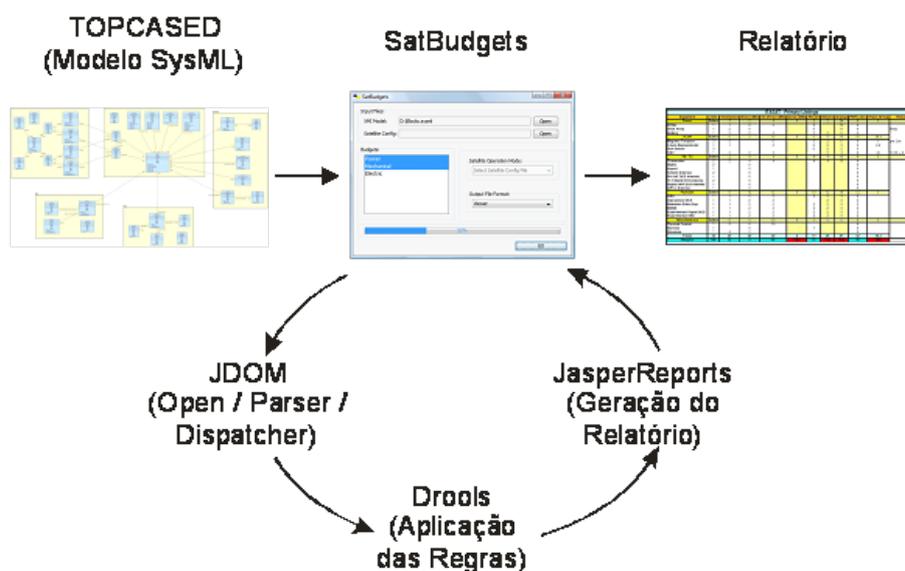


Figura 4.2 - Fluxo de funcionamento da SatBudgets

4.1 Uso da ferramenta SatBudgets

O uso da ferramenta SatBudgets é simples, nesta seção serão descritas as etapas para a criação do relatório utilizando a ferramenta.

O primeiro passo para a criação do relatório é identificar o modelo SysML ao qual serão aplicadas as regras, como mostrado na Figura 4.3.

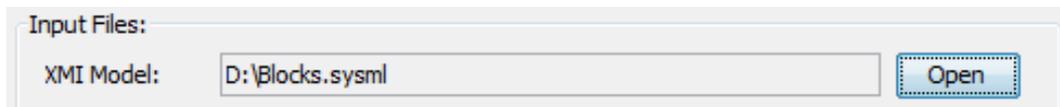


Figura 4.3 - Seleção do modelo SysML

O segundo passo para a geração do relatório é selecionar quais os balanços serão calculados para o modelo SysML escolhido. A Figura 4.4 mostra um exemplo deste passo.

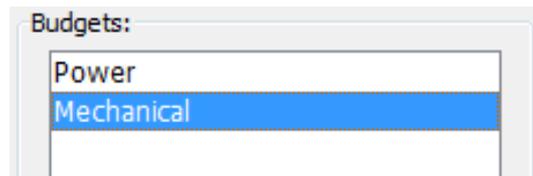


Figura 4.4 - Selecionando os balanços

O último passo antes de dar início a criação do relatório é selecionar o modo de exibição do relatório, como demonstrado na Figura 4.5. O relatório pode ser apresentado na própria aplicação (modo Viewer), ou salvo como .PDF, .XLS ou .HTML, sendo Viewer o default. Quando é escolhida uma das opções para salvar o relatório, o mesmo será salvo no mesmo caminho onde se encontra o modelo SysML informado no primeiro passo.

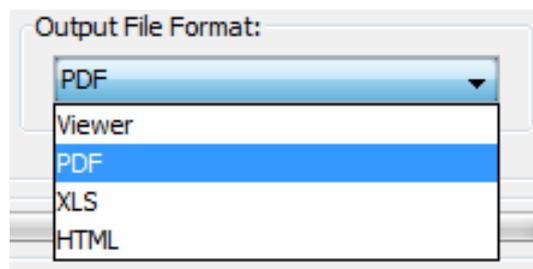


Figura 4.5 - Modos de visualização

Após ser iniciado o processamento do modelo, pressionando o botão "GO", o andamento da operação pode ser acompanhado pela barra de progresso existente na parte inferior da ferramenta, como mostrado na Figura 4.6.



Figura 4.6 - Barra de progresso

A operação é finalizada com a exibição do relatório criado pela ferramenta, como o exemplo apresentado na Figura 4.7.

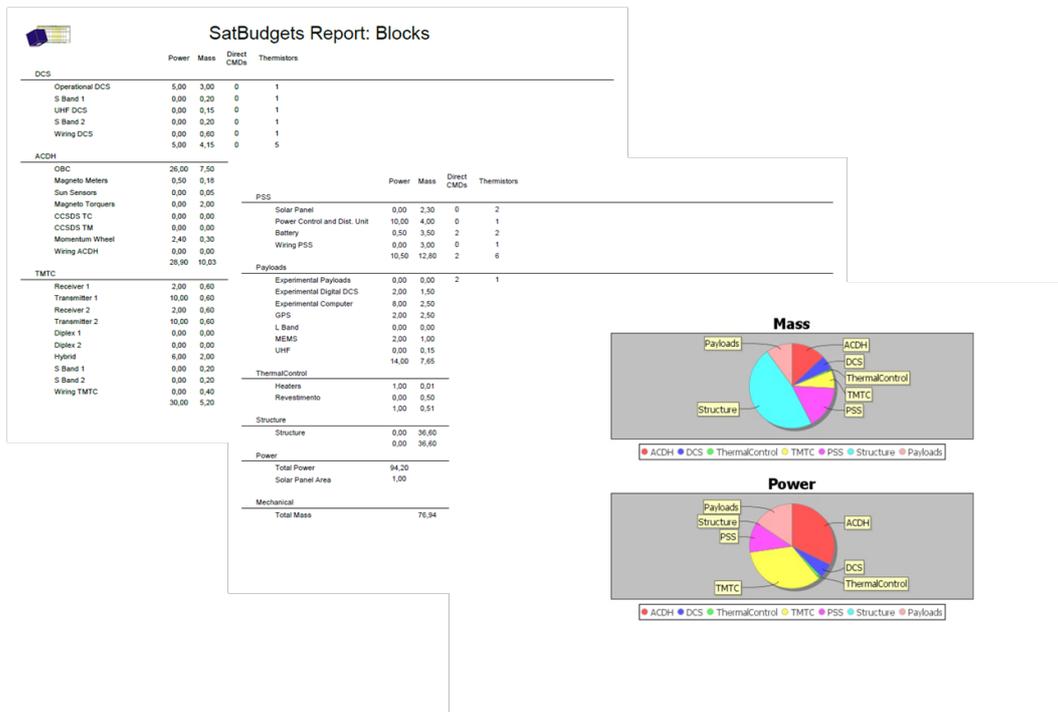


Figura 4.7 - Relatório gerado

5 Dicionário de Dados

A ferramenta SatBudgets reconhece as seguintes propriedades no modelo SysML (Diagrama de Blocos) do satélite, o que não restringe a modelagem de outras propriedades dentro dos Blocks:

- Mass;
- ActivePower; e
- ExpectedPower

Algumas propriedades devem ser obrigatoriamente modeladas para que a ferramenta consiga executar com êxito as regras definidas pelos especialistas. As propriedades obrigatórias são as seguintes:

- Block **SolarPanel** deve conter a propriedade **Area**; e
- Block **Config** deve conter a propriedade **ThermalStrategy**, que descreve qual o modelo de controle térmico será utilizado no satélite, se **Active** ou **Passive**.

As propriedades devem ser inseridas nos Blocks seguindo a seguinte sintaxe:

<Propriedade>::<Valor>

6 Conclusão

Os sistemas espaciais requerem cada vez mais bons princípios de engenharia de sistemas para lidarem com as questões de crescente complexidade do sistema. Neste sentido, foi adotada neste trabalho uma nova abordagem para projeto conceitual de satélites empregando MDE e visando atender melhor os aspectos de engenharia de requisitos de projeto. MDE permite a reutilização de informações em diferentes projetos de satélites, bem como facilita a integração e gestão das atividades de engenharia de sistemas, focando a criação de modelos ou abstrações maximizando a compatibilidade entre sistemas.

A aplicação desta abordagem neste trabalho foi baseada no uso da linguagem SysML para modelagem de satélites e na ferramenta proposta SatBudgets para dar apoio à fase de projeto conceitual.

Partindo da modelagem SysML, a proposta da ferramenta SatBudgets é gerar automaticamente balanços (mecânico, elétrico, entre outros) de projeto de satélite, demonstrando desta forma a viabilidade de construção do sistema ou a necessidade de alterações no projeto. Um estudo de caso foi aplicado ao projeto do satélite universitário ITASAT.

No proposto contexto, MDE e SatBudgets mostraram-se adequadas para serem adotadas em projetos futuros de satélites, podendo a ideia de desenvolvimento da ferramenta SatBudgets ser aplicada em outras áreas, por exemplo em projetos de veículos lançadores como já demonstrado com o IAE (Instituto de Atividades Espaciais).

REFERÊNCIAS BIBLIOGRÁFICAS

ALMEIDA, F. J. Research and choice of a methodology for the conceptual design. **Revista de Ciência & Tecnologia**, v. 8, n. 16, p. 31–42, Dez. 2000. 1

BURNETTE, E. **Eclipse IDE: pocket guide**. 1. ed. Sebastopol: O'Reilly Media, 2005. 128 p. ISBN 9780596100650. 9

DANCIU, T. **The JasperReports ultimate guide**. 2002. Disponível em: <<http://www.jaspersoft.com/jasperreports-ultimate-guide>>. Acesso em: set. de 2009. 13

FRIEDENTHAL, S.; MOORE, A.; STEINER, R. **A practical guide to SysML**. 1. ed. Oxford: Morgan Kaufmann, 2008. 560 p. ISBN 978-0-12-374379-4. 3

GILBERT, D. **The JFreeChart Developer Guide**. Object Refinery Limited, 2004. Disponível em: <<http://www.jfree.org/jfreechart/devguide.html>>. Acesso em: dez. de 2009. 15

INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING. **INCOSE Foundation**. 2009. Disponível em: <<http://www.incose.org>>. Acesso em: jul. de 2009. 3

JASPERSOFT. **iReports Tutorials & Help**. JasperSoft Corp., 2009. Disponível em: <http://jasperforge.org//website/ireportwebsite/IR%20Website/ir_documentation.html?header=project&target=ireport>. Acesso em: nov. de 2009. 14

JBOSS. **Drools Expert User Guide**. JBoss Community, 2009. Disponível em: <https://hudson.jboss.org/hudson/job/drools/lastSuccessfulBuild/artifact/trunk/target/docs/drools-expert/html_single/index.html#d0e3717>. Acesso em: dez. de 2009. 10

JDOM. **JDOM Documentation**. 2009. Disponível em: <<http://www.jdom.org/downloads/docs.html>>. Acesso em: out. de 2009. 9

LEONOR, B. B. F. **Um enfoque baseado em conhecimento e dirigido a modelos de engenharia de requisitos para projeto conceitual de satélites**. (INPE-16670-TDI/1621). Dissertação (Mestrado) — Instituto Nacional de

Pesquisas Espaciais – Laboratório Associado de Computação e Matemática Aplicada, São José dos Campos, SP, Brasil, 2010. 2

LEONOR, B. B. F.; SANTOS, W. A. dos; STEPHANY, S. A model-driven requirements engineering approach to conceptual satellite design. In: BRAZILIAN SYMPOSIUM ON AEROSPACE ENGINEERING AND APPLICATIONS / CTA DLR WORKSHOP ON DATA ANALYSIS AND FLIGHT CONTROL, 3., 2009, São José dos Campos, SP, Brasil. **Anais...** São José dos Campos: CTA/ITA/AAB/DLR, 2009. p. 10. 1

LESCOT, J. **TOPCASED v3**: installation guide. Sep. 2009. 8

OMG. **SysML**. 2009. Disponível em: <<http://www.omg.sysml.org/>>. Acesso em: abr. de 2009. 1, 8

_____. **UML**. 2009. Disponível em: <<http://www.uml.org/>>. Acesso em: abr. de 2009. 2

SANTOS, W. A. dos; LEONOR, B. B. F.; STEPHANY, S. A knowledge-based and model-driven requirements engineering approach to conceptual satellite design. **Lecture Notes in Computer Science - Conceptual Modeling - ER 2009**, v. 1, p. 487–500, 2009. 1

SCHMIDT, D. C. Model-driven engineering. **IEEE Computer Society**, p. 25–31, Feb. 2006. 1

WEILKIENS, T. **Systems engineering with SysML/UML**: Modeling, analysis, design. 1. ed. Oxford: Morgan Kaufmann, 2007. 307 p. ISBN 978-0-12-374274-2. 4