sid.inpe.br/mtc-m19/2013/05.06.22.04-TDI

CLOUDSATCD: UMA ABORDAGEM DE e-ENGINEERING NO PROJETO CONCEITUAL DE SATÉLITE

Alex Barbosa Bastos

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelo Dr. Walter Abrahão dos Santos , aprovada em 08 de maio de 2013.

URL do documento original:
<http://urlib.net/8JMKD3MGP7W/3E3TJ58>

INPE
São José dos Campos
2013

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

sid.inpe.br/mtc-m19/2013/05.06.22.04-TDI

CLOUDSATCD: UMA ABORDAGEM DE e-ENGINEERING NO PROJETO CONCEITUAL DE SATÉLITE

Alex Barbosa Bastos

Dissertação de Mestrado do Curso de Pós-Graduação em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais, orientada pelo Dr. Walter Abrahão dos Santos , aprovada em 08 de maio de 2013.

URL do documento original:
<http://urlib.net/8JMKD3MGP7W/3E3TJ58>

INPE
São José dos Campos
2013

Dados Internacionais de Catalogação na Publicação (CIP)

Bastos, Alex Barbosa.
B297c CloudSatCD: uma abordagem de e-Engineering no projeto conceitual de satélite / Alex Barbosa Bastos. – São José dos Campos : INPE, 2013.
xxii + 182 p. ; (sid.inpe.br/mtc-m19/2013/05.06.22.04-TDI)

Dissertação (Mestrado em Engenharia e Tecnologia Espaciais/Engenharia e Gerenciamento de Sistemas Espaciais) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2013.
Orientador : Dr. Walter Abrahão dos Santos.

1. e-Engineering. 2. engenharia de sistemas. 3. nuvem. 4. colaboração. 5. projeto conceitual. I.Título.

CDU 621:658.5

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de ***Mestre*** em

Engenharia e Tecnologia Espaciais/Gerenciamento de Sistemas Espaciais

Dr. Germano de Souza Kienbaum

Presidente / INPE / SJCampos - SP

Dr. Walter Abrahão dos Santos

Orientador(a) / INPE / São José dos Campos - SP

Dr. Álvaro Augusto Neto

Convidado(a) / ITA / São José dos Campos - SP

Este trabalho foi aprovado por:

( ) maioria simples

( ) unanimidade

Aluno (a): ***Alex Barbosa Bastos***

São José dos Campos, 08 de Maio de 2013

AGRADECIMENTOS

A minha família por todo apoio e incentivo durante todo o período em que estive comprometido com o curso. Dentre estes, um agradecimento especial a meu pai, Roberto, pelo esforço incondicional e desmedido que fez para que eu pudesse estar aqui, desde o início dessa caminhada até o dia de hoje. E sei que continuará sempre sendo assim.

Ao Dr. Walter Abrahão dos Santos, orientador e professor, que acreditou na capacidade de um aluno que sequer tinha um projeto definido, apenas um objetivo e uma vaga ideia do que queria utilizar. Exigiu e corrigiu quando foi preciso, sem nunca deixar de se empenhar e ensinar. Conseguiu mostrar as melhores opções e possibilidades, além de compartilhar algo muito valioso: conhecimento.

Aos membros da Banca, cujas críticas e sugestões serviram sempre como forma construtiva para buscar o melhor.

Aos professores que conheci e com os quais tive a satisfação de aprender.

À Edleusa pela grande ajuda e presteza.

Ao INPE pela oportunidade de estudar e evoluir dentro de uma das instituições de pós-graduação mais conceituadas do país.

À “Dona” Fátima, Supervisora de Recursos Humanos da Prefeitura de Cachoeira Paulista, por ter sido amiga e também por ter me ajudado inúmeras vezes desde o dia em que comecei o mestrado, ajustando meus horários de forma que eu pudesse estudar e trabalhar.

As minhas três chefias imediatas também na Prefeitura de Cachoeira Paulista, Luciana, Margarete e Waléria, pela amizade, compreensão e apoio.

A todas as pessoas que contribuíram de alguma forma durante esse período importante.

A Deus, sempre, por estar comigo e me conduzir.

RESUMO

Este trabalho propõe uma arquitetura de *e-Engineering* e demonstra uma implementação de um ambiente, denominado CloudSatCD, aplicado à Engenharia de Sistemas Espaciais, expondo serviços numa nuvem privada/local para a fase de projeto conceitual de satélites. A solução estende dois sistemas legados, SatBudgets e SpaceESB, para automação de balanços de engenharia e disponibilização de serviços em barramento respectivamente, com um *groupware* e um gerenciador de requisitos. Espera-se que a solução possa promover a colaboração entre os *stakeholders* de projeto e a horizontalização da comunicação com a equipe de engenharia de sistemas, tornando, assim, o processo de desenvolvimento, através do apoio computacional, mais eficiente.

CLOUDSATCD: AN E-ENGINEERING APPROACH IN THE CONCEPTUAL DESIGN OF SATELLITES

ABSTRACT

This work proposes an architecture of e-Engineering and demonstrates an implementation of an environment applied for Space Systems Engineering, named CloudSatCD, in a private/local cloud for the provision of services to support the conceptual design phase of satellites. The solution extends two legacy systems, SatBudgets and SpaceESB, designed for engineering budgets automation and availability of services in bus respectively, with a groupware and a requirements manager. It is expected that the solution may promote collaboration among project stakeholders by means of communication with the systems engineering team, increasing the efficiency of the development process.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS E ABREVIATURAS

ACL	*Access Control List*
ACPI	*Advanced Configuration and Power Interface*
ALBATROs	*Automated Listing Budgeting And Tracing Repository*
API	*Application Programming Interface*
BSD	*Berkeley Software Distribution*
CAD	*Computer-Aided Design*
CCSDS	*Consultative Committee for Space Data Systems*
CDF	*Concurrent Design Facilities*
COTS	*Commercial Off-The-Shelf*
CRM	*Customer Relationship Management*
CRUD	*Create, Retrieve, Update, Delete*
CSCW	*Computer Supported Cooperative Work*
CWE	*Collaborative Work Environment*
Daemon	*Disk And Execution MONitor*
DEVICE	*Distributed Environment for Virtual Integrated Collaborative Engineering*
DSE	*Design Space Exploration*
EAP	Estrutura Analítica do Projeto
ESA	*European Space Agency*
ESB	*Entreprise Service Bus*
FAQ	*Frequently Asked Questions*
FEA	*Finite Element Analysis*
GB	*GigaByte*
GRE	*Generic Routing Encapsulation*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
HTTPS	*HyperText Transfer Protocol over SSL*
IaaS	*Infrastructure as a Service*
ICMP	*Internet Control Message Protocol*

ID	*Identity Document*
IMAP	*Internet Message Access Protocol*
INPE	Instituto Nacional de Pesquisas Espaciais
IP	*Internet Protocol*
KVM	*Kernel-based Virtual Machine*
NextGenRE	*Next Generation Requirements Engineering*
NFS	*Network File System*
OCDS	*Open Concurrent Design Server*
OCDT	*Open Concurrent Design Tool*
OSS	*Open Source Software*
PaaS	*Plataform as a Service*
PAE	*Physical Address Extension*
PDP	Processo de Desenvolvimento de Produto
PLM	*Product Lifecycle Management*
PPP	*Point-to-Point Protocol*
PPTP	*Point-to-Point Tunneling Protocol*
QA/PA	*Quality Assurance/Product Assurance*
REST	*Representational State Transfer*
RPC	*Remote Procedure Call*
RSS	*Really Simple Syndication*
RTH	*Requirements and Testing Hub*
SaaS	*Software as a Service*
SCP	*Secure Copy*
SEA	*Systems Engineering Area*
SLA	*Service Level Agreement*
SMTP	*Simple Mail Transfer Protocol*
SOA	*Service-Oriented Architecture*
SOAP	*Simple Object Access Protocol*
SOC	*Service-Oriented Computing*
SQL	*Structured Query Language*
SSH	*Secure Shell*

SSL	*Secure Sockets Layer*
SysML	*Systems Modeling Language*
TCP	*Transmission Control Protocol*
TI	Tecnologia da Informação
TLS	*Transport Layer Security*
UDDI	*Universal Description, Discovery and Integration*
UDP	*User Datagram Protocol*
URI	*Uniform Resource Identifier*
VFS	*Virtual File System*
VM	*Virtual Machine*
VNC	*Virtual Network Computing*
VPN	*Virtual Private Network*
VRML	*Virtual Reality Modeling Language*
XML	*eXtensible Markup Language*
WebDAV	*Web-based Distributed Authoring and Versioning*
WS-BPEL	*Web Services Business Process Execution Language*
WSDL	*Web Services Description Language*
YES	*Young Engineers' Satellite*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, o processo de desenvolvimento de sistemas está cada vez mais abrangente e complexo, devido às novas tecnologias e às constantes mudanças na área de TI (Tecnologia da Informação). Parte desta complexidade advém da reutilização e integração com sistemas legados, da necessidade de qualidade a custo acessível e da implementação de novos recursos para maiores funcionalidades e usabilidade ao usuário final (MEDEIROS, 2010).

A Engenharia de Sistemas é um campo interdisciplinar que se foca em como produtos complexos de Engenharia devem ser projetados e gerenciados ao longo de seu ciclo de vida, trabalhando com processos e ferramentas para lidar com estes projetos e utilizando-se de conceitos de outras áreas correlatas (FRIEDENTHAL et al., 2008).

Sistemas espaciais particularmente exigem a aplicação desta abordagem pela natural complexidade em lidar com vários segmentos de projeto e pelos seus requisitos de alta confiabilidade que devem ser tratados logo no início do ciclo de vida, na fase de projeto conceitual de sistemas (LARSON; WERTZ, 1999).

Por extensão, Engenharia de Sistemas Espaciais é uma abordagem que emprega práticas aceitas na engenharia de sistemas para planejar, definir, controlar, projetar, analisar, integrar, verificar e apoiar a produção, testes e operação de produtos de hardware, software, programas ou sistemas espaciais. Esta abordagem geralmente requer alta interação entre os engenheiros de sistemas uma vez que são integradas/descritas por meio de um *workflow*, cujos processos se encontram sob responsabilidade de diversos membros da equipe (LEONOR, 2010).

O desenvolvimento de produtos complexos, como satélites, por exemplo, vem sendo realizado de forma distribuída, com equipes de projeto geográfica e temporalmente dispersas. Essas barreiras adicionam complexidade ao esforço

de desenvolvimento do projeto. Problemas devidos à falta de coordenação, ineficientes infraestruturas de comunicações, diferenças culturais e de fuso horário, falta de interação e colaboração, são mencionados como os principais fatores de atraso de projeto e até mesmo seu fracasso (GRUDIN, 1988). Essa situação torna-se verdadeira mesmo quando as equipes estão co-localizadas ou próximas à organização principal.

Como contraponto a estes fatores sugere-se o emprego de *e-Engineering*, ou seja, um ambiente colaborativo web voltado para engenharia em um espaço virtual onde todos os participantes de um projeto (partes interessadas), mesmo se distribuídos pelo tempo ou espaço, podem negociar, realizar *brainstormings*[1], discutir, compartilhar o conhecimento, e trabalhar em conjunto para realização de alguma tarefa (BOOCH; BROWN, 2003), a fim de elucidar os problemas anteriormente citados.

Este trabalho propõe disponibilizar um ambiente colaborativo de *e-Engineering* que integra uma aplicação de *groupware* e um gerenciador de requisitos com dois sistemas legados: um aplicativo chamado SatBudgets (DOS-SANTOS et al., 2009) e sua extensão num barramento de serviços corporativos chamado SpaceESB (SOUZA; DOS-SANTOS, 2011). Ambos os sistemas visam auxiliar na fase de projeto conceitual dos projetos de satélites. A adaptação destes sistemas é realizada colocando em nuvem suas principais funcionalidades e provendo serviços de *e-Engineering*, a fim de auxiliar engenheiros de sistemas, gerentes e responsáveis pelas principais decisões durante a fase de projeto conceitual.

A ideia de aplicativos que usam serviços independentes espalhados pela rede constitui o paradigma denominado Computação Orientada a Serviços (SOC – *Service-Oriented Computing*) (GROSSI, 2005) e pode ser aplicado para solucionar problemas nas mais variadas áreas de conhecimento, incluindo *e-Engineering*. A abordagem é extremamente interessante, pois permite o uso de

[1] Atividade desenvolvida para explorar a potencialidade criativa de um grupo, visando a obtenção das melhores soluções para projetos.

ferramentas que auxiliem e gerenciem os processos de negócio, tal como o projeto conceitual de satélites.

O projeto conceitual de satélites visa avaliar possíveis alternativas de arquiteturas capazes de satisfazer os requisitos gerais e as principais restrições da missão (por exemplo, massa e custo). Os vários conceitos são então avaliados de acordo com um conjunto de critérios prioritários a fim de escolher projetos para desenvolver com maiores níveis de detalhes. O principal objetivo da avaliação é identificar os principais requisitos da missão e determinar se os conceitos selecionados cumprem estes requisitos (ECSS, 2008).

O projeto conceitual de satélites realizado no INPE é feito por um grupo de engenheiros de sistemas que utilizam planilhas e softwares dedicados às áreas específicas de seus subsistemas para verificar a viabilidade de uma determinada arquitetura candidata de projeto. Através de interações entre o grupo, uma arquitetura começa a maturar até que a mesma seja homologada por uma posterior revisão de projeto (SOUZA, 2011a).

O processo atual de interação possui baixo grau de automação e forte acoplamento, pois depende diretamente da comunicação entre os participantes do projeto. Mais complexidade é adicionada quando estão envolvidos parceiros externos, trazendo à tona mais morosidade ao processo, que demanda agilidade no desenvolvimento e na tomada de decisões.

Este trabalho propõe uma forma para que os envolvidos no projeto possam trabalhar conjuntamente, apoiado por um ambiente computacional de *e-Engineering*, minimizando as limitações impostas pelo cenário descrito anteriormente e contribuindo na agilidade e eficiência da fase de projeto conceitual. Especificamente, busca-se otimizar as tomadas decisões, a comunicação entre os participantes do projeto, estimular a colaboração entre as diferentes áreas do conhecimento envolvidas no projeto, proporcionar melhor coordenação ao projeto e, principalmente, otimizar os processos da fase de projeto conceitual.

1.1. Motivação

Os problemas anteriores citados que envolvem projetos conceituais de satélites com participantes distribuídos em tempo e espaço impõem barreiras, riscos e desafios ao setor de engenharia, exigindo maior interação entre profissionais de diversos domínios de conhecimento, ou seja, equipes multidisciplinares. Ao mesmo tempo isto requer também o uso de suporte computacional, visando maior qualidade, menores prazos e custo final de projeto.

O projeto conceitual torna-se a fase mais importante no ciclo de vida de um projeto espacial, pois é nela em que a avaliação de conceitos e a escolha da futura arquitetura do satélite são feitas e as decisões mais importantes são tomadas.

Nessa fase engenheiros de sistemas e pesquisadores de diferentes áreas de conhecimento atuam conjuntamente visando a criação de uma arquitetura constituída pela melhor relação dos componentes com as características requeridas do sistema final. O projeto conceitual é capaz de gerar especificações bem detalhadas do produto final, o satélite, antes mesmo de este começar a ser construído.

Entretanto, problemas como coordenação, colaboração e comunicação são intrínsecos à natureza dos projetos com participantes distribuídos. A colaboração entre os participantes torna-se extremamente difícil quando pessoas das diversas áreas envolvidas estão dispersas geográfica e temporalmente. Essa situação é agravada quando é preciso trabalhar com diferenças de idiomas, fuso horário, sistemas computacionais, sistemas de medidas e alguns outros fatores que impõem barreiras e restrições à comunicação e, consequentemente, aumento no tempo para tomada de decisões. Tais decisões são cruciais e podem determinar o sucesso ou falha de uma missão.

A fim de auxiliar os envolvidos no projeto durante essa fase tão importante do desenvolvimento de satélites, abordagens apropriadas de engenharia e, por conseguinte, ferramentas, são necessárias para que os problemas enunciados anteriormente sejam minimizados. O uso da engenharia simultânea, ou concorrente, em conjunto com ferramentas colaborativas, tem se mostrado extremamente eficiente, de tal forma que a ESA (*European Space Agency*), tem utilizado essa abordagem (ECSS, 2010), com sucesso, em seus projetos, através de suas instalações de projetos concorrentes (CDF – *Concurrent Design Facilities*), que proporciona a capacidade de usar modelos de engenharia de forma interativa para revisões e iterações de projeto. Em seus vários centros de projetos simultâneos, um modelo padrão com base na troca de dados permite colaboração em projetos de engenharia de maneira mais eficaz.

Atualmente diversas aplicações locais vêm sendo agregadas a outros sistemas em rede e em nuvem, bem como há um interesse crescente em ambientes colaborativos (ROSS et al., 2006). No INPE (Instituto Nacional de Pesquisas Espaciais), diferentemente da maneira como é feita na ESA, a equipe de engenharia de sistemas tem utilizado ferramentas baseadas em planilhas manualmente geradas além de um conjunto de aplicativos específicos para cada subsistema de satélites que geralmente não trabalham integradas. Adicionalmente, dados de projetos são trocados de maneira ainda manual e sequencial, deixando de explorar o caráter paralelo e distribuído de decisões de projeto conceitual (LEONOR, 2010).

Como a atuação dos profissionais envolvidos, especialmente durante a fase de projeto conceitual, não se dá individualmente, fica evidente a carência de ferramentas e de um ambiente que promova colaboração, comunicação e coordenação entre eles, de forma a minimizar os problemas inerentes aos projetos. Esta constitui a principal motivação para este trabalho.

1.2. Objetivo

O objetivo geral desta pesquisa é a proposta de uma arquitetura e o desenvolvimento/integração de um ambiente colaborativo web de *e-Engineering* com base nela, voltado para a engenharia de sistemas espaciais na fase de projeto conceitual, integrando e complementando duas ferramentas existentes para avaliação de modelos de arquiteturas de satélites, SatBudgets e SpaceESB. O conceito é demonstrado pela implementação de um protótipo da arquitetura, aqui denominado CloudSatCD, capaz de prover serviços em nuvem para apoiar durante a fase de projeto conceitual, onde existe a necessidade de interoperabilidade entre algumas das diversas ferramentas utilizadas para análise de cenários, gerenciamento, colaboração, comunicação e coordenação.

Por se tratar de um ambiente web, esses serviços poderão ser acessados pela equipe de engenharia de sistemas do INPE e eventuais parceiros envolvidos, independentemente de plataforma computacional e local, bastando apenas possuir um navegador (*browser*) e conexão com a Internet.

1.3. Metodologia

A metodologia utilizada para atingir os objetivos propostos neste trabalho compreende os passos a seguir.

Numa primeira etapa realizar uma revisão bibliográfica visando conhecer os principais aspectos de engenharia concorrente de sistemas, projeto conceitual de satélites, computação orientada a serviços, computação em nuvem, *e-Engineering*, trabalhos anteriores e trabalhos relacionados à proposta de ambiente colaborativo.

Numa segunda etapa investigar os conceitos relacionados às tecnologias e as ferramentas já existentes que serão integradas e estendidas para produção do ambiente proposto. Para isto fazer uso da literatura existente, buscando

identificar a melhor forma de conciliar essas tecnologias e ferramentas priorizando soluções de código aberto, de forma a serem alteradas e adaptadas conforme a necessidade.

Na terceira etapa realizar a integração e customização das ferramentas para a formação do produto final do trabalho, CloudSatCD: o ambiente de *e-Engineering*, centrado no *groupware*.

Por fim, a última etapa documentar toda a implementação, testes e demonstração de sua aplicabilidade. Com os resultados obtidos, fazer considerações finais sobre a eficácia global da proposta e sugestões para trabalhos futuros envolvendo essa linha de pesquisa e o ambiente desenvolvido.

1.4. Estrutura da Dissertação

Este trabalho está organizado da forma descrita a seguir.

Este primeiro capítulo faz uma breve definição e contextualização do escopo e objetivos do trabalho.

O segundo capítulo introduz a fundamentação teórica do trabalho, apresentando conceitos, definições e aspectos do projeto conceitual de satélites e de ambientes de *e-Engineering.*

O terceiro capítulo aborda a fundamentação tecnológica utilizada para criação do ambiente proposto, através dos paradigmas da computação orientada a serviços e da computação em nuvem.

O quarto capítulo aborda os trabalhos relacionados, assim como outras ferramentas colaborativas e alguns casos de uso, além de apresentar os sistemas legados SatBudgets e SpaceESB, ambos utilizados para compor a arquitetura proposta.

O quinto capítulo apresenta a proposta deste trabalho, descrevendo a prototipação do ambiente de *e-Engineering* proposto, denominado CloudSatCD, bem como a integração entre os sistemas legados e a infraestrutura, seu funcionamento e também alguns testes de uso para os serviços disponibilizados.

Finalmente, o sexto capítulo encerra o trabalho fazendo as considerações finais e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os principais conceitos, técnicas e paradigmas que servem de embasamento teórico para o desenvolvimento do trabalho. O primeiro tópico inicia-se com a definição de aspectos do projeto conceitual de satélites, domínio que caracteriza a proposta de contribuição deste trabalho. O segundo tópico descreve a conceitualização de *e-Engineering* e descrição do ambiente colaborativo web voltado para engenharia que serve de base para a aplicação dos conceitos à pesquisa no sentido prático: o protótipo CloudSatCD.

2.1. Projeto Conceitual de Satélites

As atividades de diversas áreas de conhecimento e diferentes naturezas são oriundas geralmente de algum tipo de projeto, envolvendo planejamento, estratégia, controle e segurança.

Um projeto é definido como um esforço temporário, com início e final definidos, empreendido para criar um produto, serviço ou resultado exclusivo. O projeto compreende um grupo de atividades coordenadas e controladas a fim de atingir seus objetivos conforme requisitos específicos, incluindo limitações de tempo, custo e recursos (PMBOK, 2008).

Organizações que buscam qualidade de processos e produtos devem estabelecer formas de gerenciamento de atividades como tempo, custo e prazo, que são grandes causadores de falhas nos projetos, justamente pela falta de práticas padronizadas de gerenciamento e controle dos mesmos, provenientes da desinformação ou da falta de interesse em aplicar metodologias comprovadamente eficazes.

Com a modernização dos sistemas e o consequente aumento de complexidade, faz-se necessário cada vez mais que projetos conceituais sejam desenvolvidos, a fim de especificar, analisar, projetar e verificar sistemas sem a necessidade de maior retrabalho (DOS-SANTOS et al., 2009).

Neste sentido, um projeto espacial é um conjunto de atividades de estudo e desenvolvimento de um sistema espacial, geralmente realizado em fases (0, A, B, C, D, E e F) conforme ilustrado na Figura 2.1 (SOUZA, 2011c). O ciclo de vida é dividido de forma que cada fase compreenda um grupo de atividades, sendo: fase 0, análise da missão e identificação das necessidades; fase A, análise de viabilidade e especificação funcional; fase B, definição preliminar; fase C, definição detalhada; fase D, qualificação e produção; fase E, utilização (lançamento e operação); e fase F, descarte (ECSS, 2009c). Dessa forma, o projeto conceitual de satélites está localizado entre as fases 0 e A, compreendendo a pré-fase A, onde estão as atividades de concepção do sistema espacial após terem sido definidos os objetivos da missão.

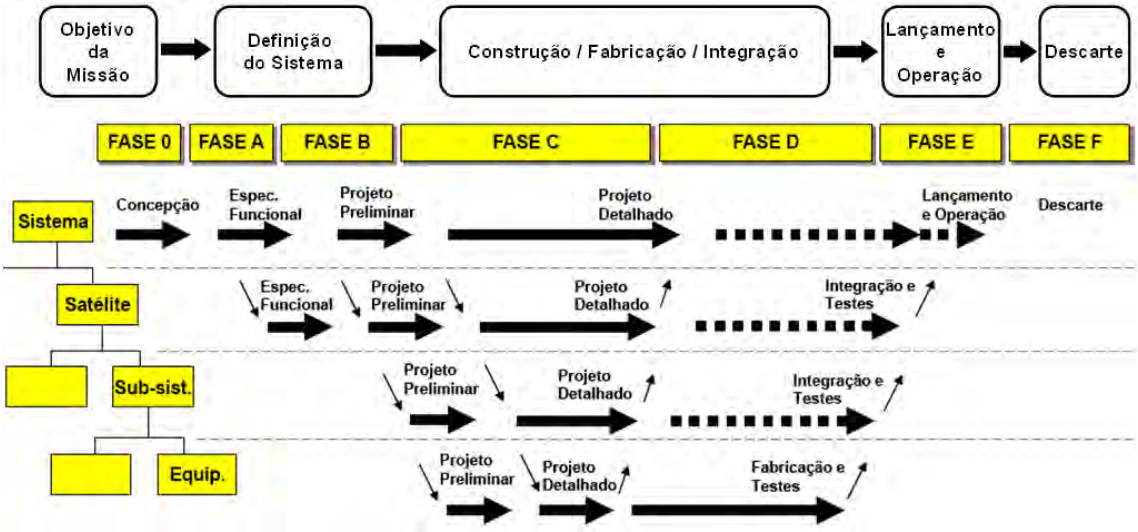


Figura 2.1 - O ciclo de vida de um projeto espacial
Fonte: Adaptado de Souza (2011c)

As fases 0 e A (e B em parte) focam principalmente (ECSS, 2009c): (1) elaboração dos requisitos técnicos e funcionais do sistema e identificação dos conceitos do sistema conforme os objetivos da missão, considerando as restrições técnicas e programáticas identificadas; (2) identificação de todas as atividades e recursos a serem utilizados no desenvolvimento do projeto; (3)

avaliações iniciais sobre riscos técnicos e programáticos; e (4) início de atividades de pré-desenvolvimento.

O estabelecimento dos requisitos técnicos durante a fase 0 inicia com a identificação e avaliação dos diferentes conceitos possíveis, através de uma avaliação inicial do projeto e dos resultados das especificações técnicas preliminares, conforme ilustrado na Figura 2.2, cujo objetivo é expressar as necessidades do cliente, os objetivos da missão, a restrição ambiental associada e elementos programáticos em termos de requisitos técnicos (ou seja, o problema a resolver). Este documento serve como uma base para iniciar a fase seguinte (ECSS, 2009b).

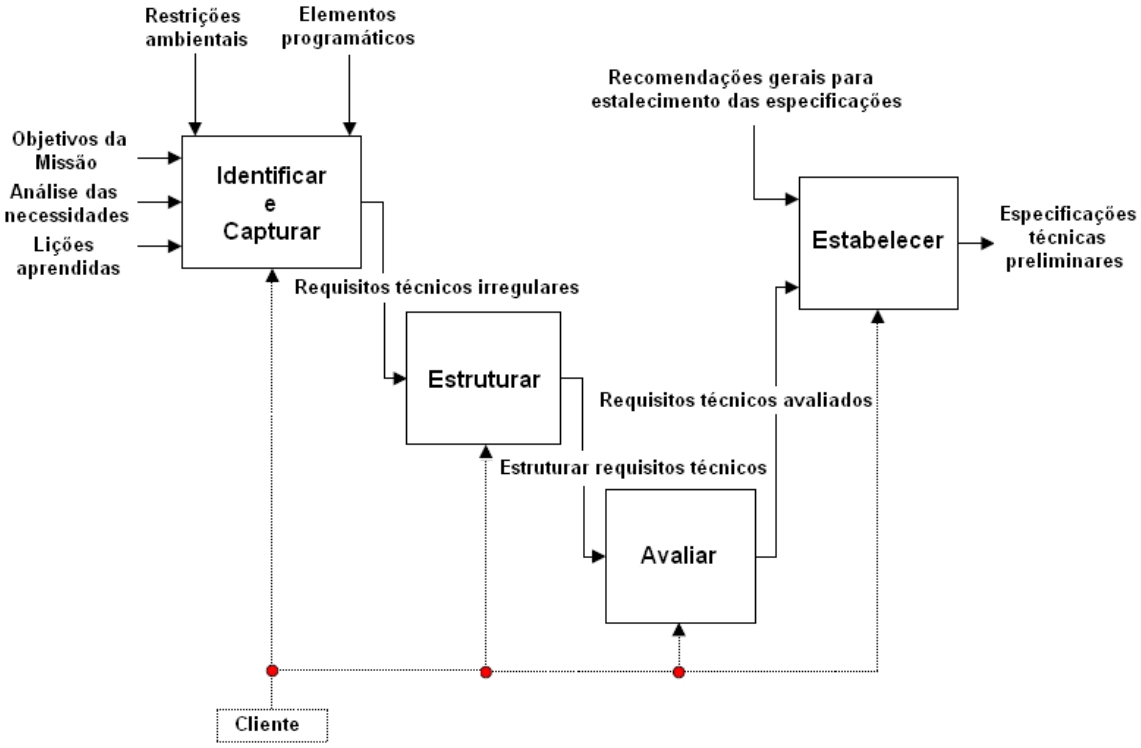


Figura 2.2 - Processo para estabelecer as especificações técnicas preliminares
Fonte: Adaptado de ECSS (2009b)

O processo tem início com a identificação e captura dos objetivos da missão ou necessidades e restrições ambientais associadas, expressos em termos de requisitos técnicos, sendo então classificados, justificados e estruturados em

requisitos técnicos individuais. A partir deste ponto é avaliado todo o conjunto de requisitos técnicos para correção, consistência e adequação ao uso pretendido, para por fim estabelecer as especificações técnicas preliminares e liberá-las (ECSS, 2009b).

O segundo passo consiste na exploração entre os diferentes conceitos possíveis, assegurando a conformidade com as necessidades definidas e, em seguida, a seleção de um conceito e resultados nas especificações técnicas. Esta versão é progressivamente elaborada a partir das especificações técnicas preliminares e considera as restrições induzidas a partir dos conceitos possíveis (ECSS, 2009b).

Assim, o projeto conceitual é a fase inicial do processo de projeto de um produto e exige a aplicação de técnicas especiais de engenharia de sistemas. Durante esta fase é necessária a avaliação de diversos cenários que antevêem a arquitetura mais adequada ao sistema final. Isto requer flexibilidade na execução de processos intrínsecos desta fase inicial, mas muito relevantes de projeto, tais como: análise de compromisso (*trade-offs*) e tomada de decisões (ALMEIDA, 2000).

O estudo de projeto conceitual é uma visão rápida sobre a viabilidade de construção um produto, cuja intenção é obter uma visão de alto nível do projeto, ao invés de se determinar os valores exatos de cada parâmetro de projeto. Num projeto em fase conceitual, os requisitos ainda não estão bem definidos e as especificações detalhadas não estão congeladas (*baseline*). A equipe ainda pode explorar outros cenários, alterando parâmetros a fim de buscar a melhor arquitetura. Muitas propostas para novas missões, se não a maioria, nunca vão além da fase de projeto conceitual, pois, normalmente, o custo da missão acaba por ser muito alto, ou o estudo expõe algum tipo deficiência no projeto (SMITH et al., 2001).

A finalidade destes estudos é avaliar a viabilidade de uma nova missão espacial do ponto de vista técnico, programático e econômico. Isto

normalmente é alcançado através da produção de um projeto conceitual preliminar da missão e sistema espaço, como indicado na Figura 2.3. O relatório do estudo resultante é utilizado como uma entrada para estudos de concepção na Fase A (BANDECCHI et al., 1999).

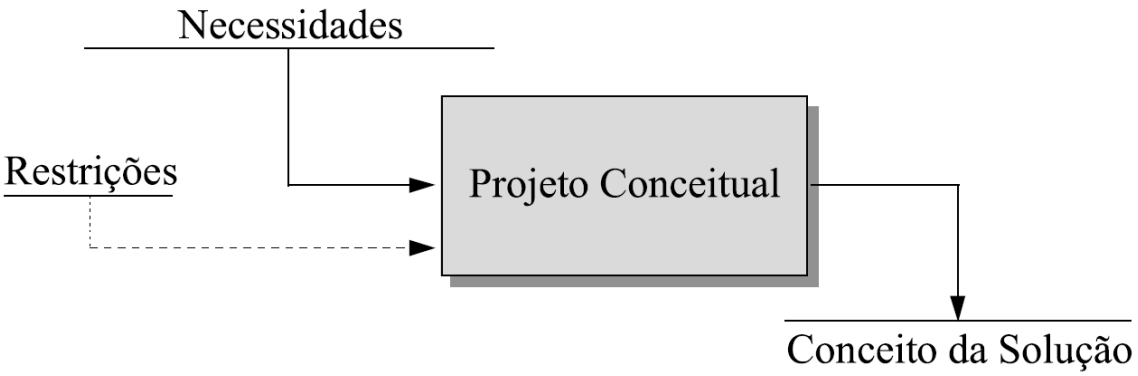


Figura 2.3 - Fase de projeto conceitual
Fonte: Almeida (2000)

De maneira geral, as entradas, tarefas e saídas obtidas durante a fase de projeto conceitual de satélites são demonstradas na Tabela 2.1.

Tabela 2.1 - Entradas, tarefas e saídas do projeto conceitual de satélites

Entradas	Tarefas	Saídas
Definição do problema Objetivos da missão Restrições Requisitos gerais Funções	Estabelecer especificações técnicas Gerar alternativas de projeto Avaliar conceitos de solução Gerar parâmetros/balanços (*budgets*) Gerar modelo de arquitetura funcional	Projeto conceitual *Baseline* de requisitos Especificações de projeto

Essa fase do projeto, portanto, deve ser sistematizada para que seja possibilitada sua integração com as demais fases do projeto do produto. Tal integração é implementada através de sistemas computacionais, devido ao grande volume de dados envolvidos, a sua organização e à necessidade de meios e sistemas de controle rápidos e eficientes sobre o processo como um todo (ALMEIDA, 2000).

Isto também ocorre em projetos de satélites, principalmente considerando a grande quantidade de recursos, humanos e financeiros, envolvidos em sua produção e também os benefícios trazidos pelos mesmos à população. Um bom planejamento é essencial, uma vez que evita erros que podem comprometer todo o empreendimento (DOS-SANTOS et al., 2009).

O projeto conceitual, o desenvolvimento e a produção de um satélite artificial são processos complexos de engenharia, resultantes da composição de uma série de subsistemas que, por sua vez, são compostos por vários equipamentos e componentes. As atividades de um projeto de satélite podem ser agrupadas em cinco grandes fases, como mostrado na Figura 2.4 (CUCO et al., 2010): Projeto Conceitual, Projeto Preliminar, Projeto Detalhado, Produção de Modelo de Voo e Testes de Aceitação. O projeto conceitual é a primeira – e mais importante – fase desse processo, pois as fases subsequentes são dependentes e oriundas dessa primeira.

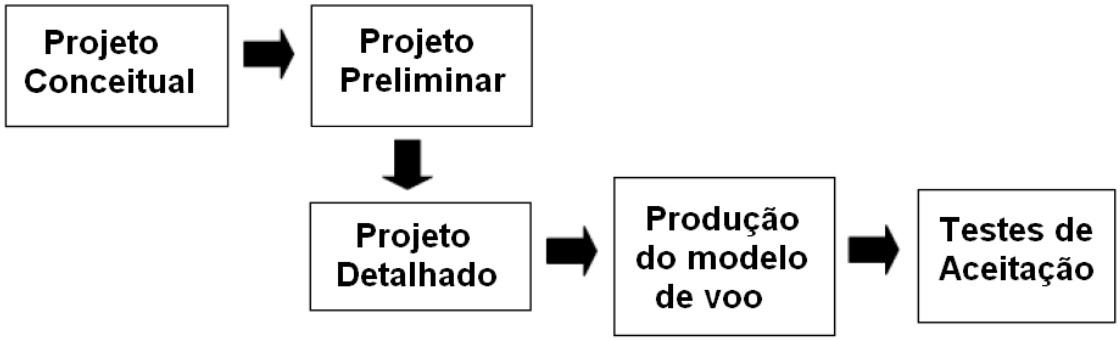


Figura 2.4 - Fases de desenvolvimento de um satélite artificial
Fonte: Cuco et al. (2010)

A modelagem conceitual de satélites envolve contribuições de várias disciplinas do conhecimento que devem ser corretamente coordenadas visando a uma solução que atenda aos requisitos da missão (SOUZA, 2011c). Por envolver várias linhas de raciocínio e uma grande quantidade de informações de diferentes áreas de conhecimento, o processo de projeto conceitual possui grande complexidade e interdisciplinaridade. Nele são manipuladas e tratadas

informações dispersas, muitas oriundas do raciocínio do próprio projetista e, portanto, não formalizadas (ALMEIDA, 2000).

O projeto conceitual de um satélite é desenvolvido normalmente através de uma dessas abordagens (CUCO et al., 2010): (1) reunião de uma equipe de especialistas dos diferentes subsistemas que, baseados na experiência em projetos anteriores, tentam conceber “manualmente” uma arquitetura que atenda aos requisitos de desempenho, custo e prazo exigidos para a missão, e (2) partindo-se de uma arquitetura já utilizada em outra missão similar, adaptando-a aos requisitos da nova missão.

Ambas as abordagens frequentemente geram configurações sub-ótimas. No primeiro caso, ainda que diferentes configurações possam ser concebidas e a mais apropriada seja escolhida, o tempo necessário para sua realização, e muitas vezes a dificuldade em coordenar eficientemente as contribuições individuais dos membros da equipe, faz com que este processo seja interrompido tão logo uma configuração que atenda aos requisitos seja obtida. No segundo caso, embora seja obtida uma solução rapidamente, assim como no processo “manual” de projeto, pode levar a uma arquitetura que não seja a melhor possível para a nova missão (CUCO et al., 2010).

As técnicas tradicionais para desenvolver o projeto conceitual de um novo satélite não exploram de maneira sistemática o espaço de projeto, limitando consequentemente o entendimento do mesmo e reduzindo assim as chances de que a melhor solução para o problema seja obtida (CUCO et al., 2010). Exploração de Espaço de Projeto (DSE – *Design Space Exploration*) é um processo para analisar várias alternativas "funcionalmente equivalentes" de uma implementação, que atendam a todas as restrições, a fim de identificar a escolha da solução mais adequada com base em métricas de qualidade, tais como custo, desempenho ou confiabilidade; vide Figura 2.5 para detalhes. Essa técnica é realizada durante o processo de projeto conceitual visando

encontrar arquiteturas ótimas ou durante a execução para auxiliar em reconfigurações dinâmicas (HEGEDÜS et al., 2010).

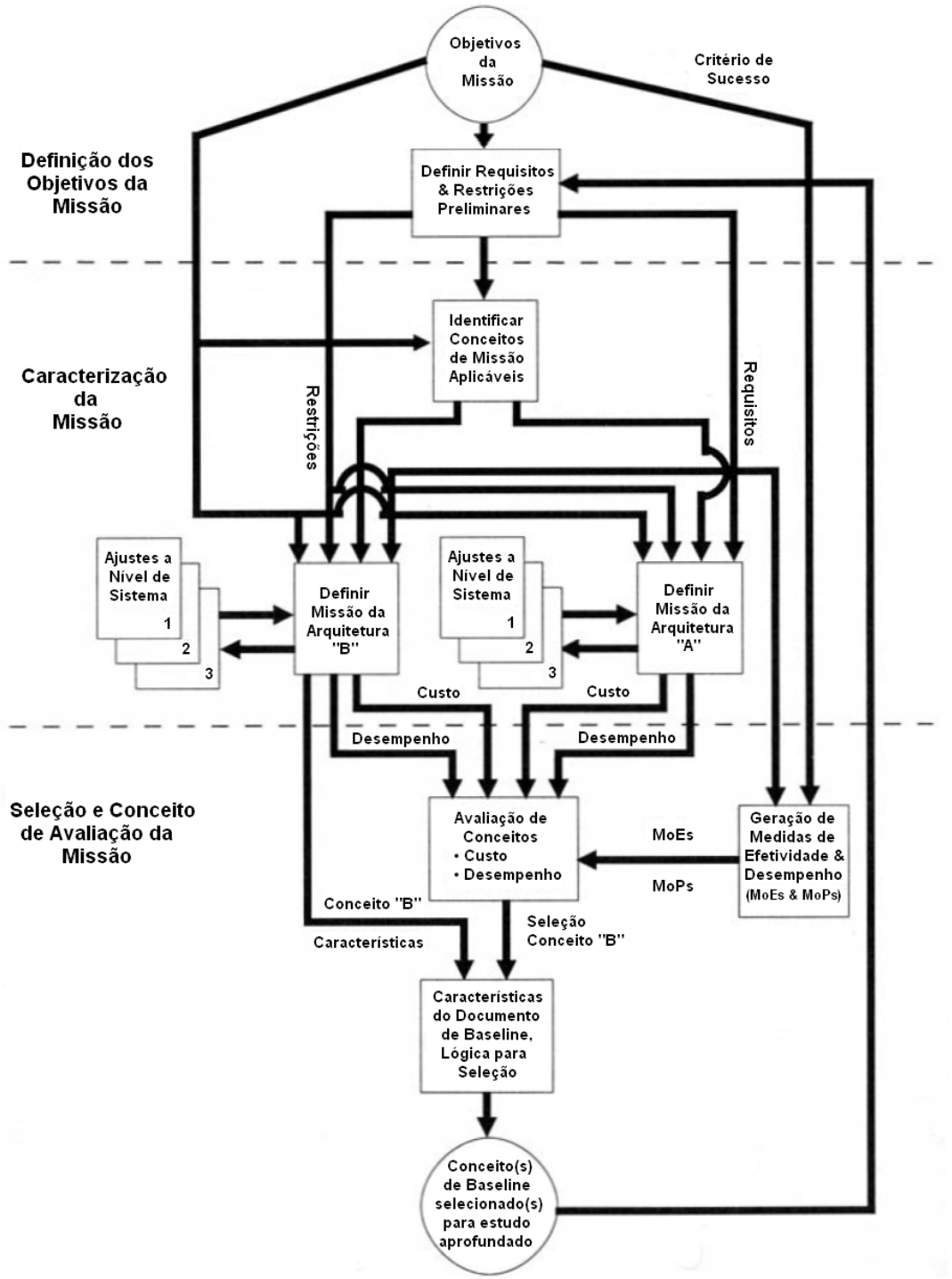


Figura 2.5 - Fluxo de exploração de conceitos

Fonte: Adaptado de Larson e Wertz (1999)

Os resultados do processo de elaboração da arquitetura e alocação dos requisitos de um satélite são: diagrama de blocos funcional do satélite; estimativas para a massa, volume e potência; configuração mecânica; balanço de massa e balanceamento dinâmico; balanço de potência (geração, consumo, profundidades de descarga das baterias); balanço de links de comunicação; balanço de apontamento; balanço de massa de propelente; balanço de capacidade de memória e de processamento (SOUZA, 2011c).

Um balanço de engenharia (*budget*) é uma lista numérica dos componentes de qualquer parâmetro do sistema global (LARSON; WERTZ, 1999) que define para cada parâmetro-chave de engenharia de um sistema ou produto a natureza deste parâmetro, sua medida, valor especificado, métricas de requisitos, valor computado e valor avaliado/estimado (ECSS, 2009a), a fim de conhecer suas exigências técnicas. Balanços são realizados para diversas disciplinas de projeto de satélites, conforme ilustrado na Figura 2.6. Como exemplo, o balanço de massa total de um satélite consistiria nas massas estimadas para os instrumentos da carga útil, os vários subsistemas, o propelente requerido, e tipicamente alguma margem para crescimento (LARSON; WERTZ, 1999).

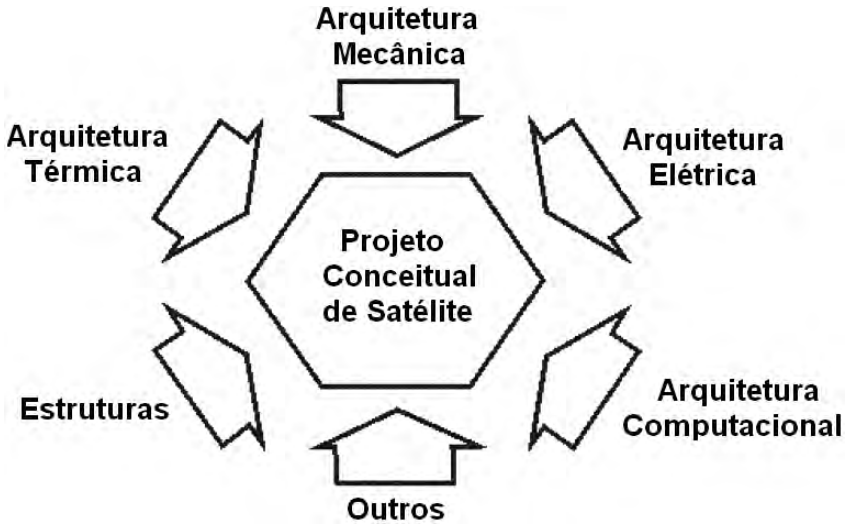


Figura 2.6 - Interdisciplinaridade no projeto conceitual de satélites
Fonte: Leonor (2010)

Uma vez que as características básicas do satélite tenham sido estabelecidas, inicia-se a fase de estabelecimento da arquitetura do satélite, que é um das fases mais importantes do projeto, pois é ela que estabelece a ligação entre a concepção geral do segmento espacial e sua efetiva materialização. A representação dos requisitos de projeto é um conceito chave para a prática da engenharia de sistemas e visa descrever, em modelos conceituais, as condições ou capacidades do sistema, tendo como objetivo principal propiciar uma melhor compreensão do domínio do problema e uma precisa definição do domínio da solução. Essa representação é estabelecida em um nível ainda abstrato de modelagem onde os modelos produzidos, textuais ou em notação gráfica, descrevem os aspectos relevantes do domínio do problema e relatam os componentes e comportamento esperado para o sistema (LEONOR, 2010).

O projeto de um satélite é, portanto, uma atividade multidisciplinar, e essa característica torna-se mais evidente durante a fase de projeto conceitual, pois é nesse momento em que a arquitetura básica do mesmo é definida, ou seja, onde é primeiramente feita a escolha dos tipos de componentes dos subsistemas e de como seus equipamentos serão distribuídos na estrutura do satélite (CUCO et al., 2010).

Portanto, ao automatizar o processo de projeto conceitual, obtém-se uma maior eficiência e velocidade de desenvolvimento, que são necessidades essenciais no mercado competitivo atual (LEONOR, 2010).

2.2. Ambiente para e-Engineering

Existe um relacionamento sinergético entre engenharia, tecnologia e ciência. Enquanto a ciência busca entender o mundo natural, e muitas vezes precisa de novas ferramentas para apoiar a descoberta de respostas, com base no conhecimento científico desenvolvido, a engenharia busca resolver problemas práticos em projetos, processos e produtos que atendam as necessidades da sociedade, através do desenvolvimento ou utilização de tecnologias

(MASS.GOV, 2001). A Figura 2.7 demonstra o relacionamento entre estes três pilares e a intersecção entre eles.

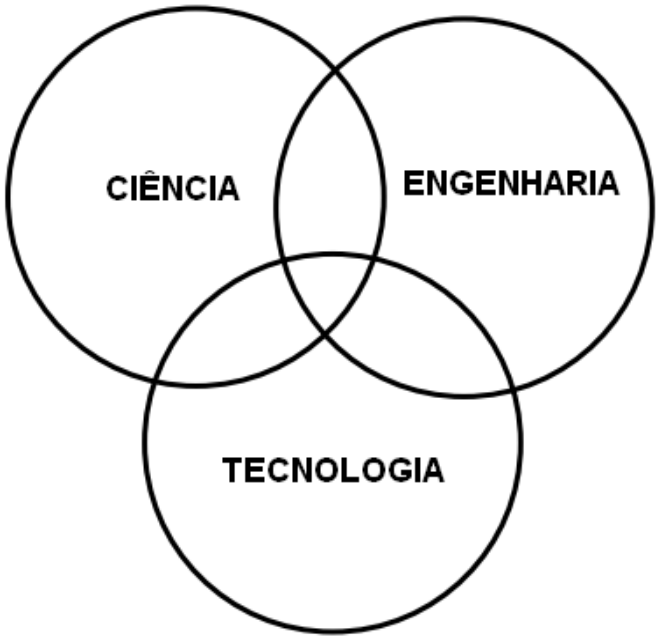


Figura 2.7 - Relacionamento entre engenharia, ciência e tecnologia
Fonte: Mass.gov (2001)

Tecnologia e engenharia trabalham em conjunto com a ciência para expandir a capacidade de compreensão. Tecnologias desenvolvidas através de engenharia incluem os sistemas que surgem como resultado de reconhecer uma necessidade ou problema (MASS.GOV, 2001).

Engenharia, Ciência e outras formas de produção de conhecimento estão se tornando mais colaborativas, distribuídas e de informação intensiva. Conforme ilustrado na Figura 2.8, as relações de sinergia entre ciência e engenharia estão sendo amplificadas através de pesquisa colaborativa e tecnologias colaborativas. Ao mesmo tempo, engenharia colaborativa, ciência colaborativa e tecnologias colaborativas estão apoiando e sendo apoiadas por *e-Engineering* e *e-Science*. (IMETI, 2012).

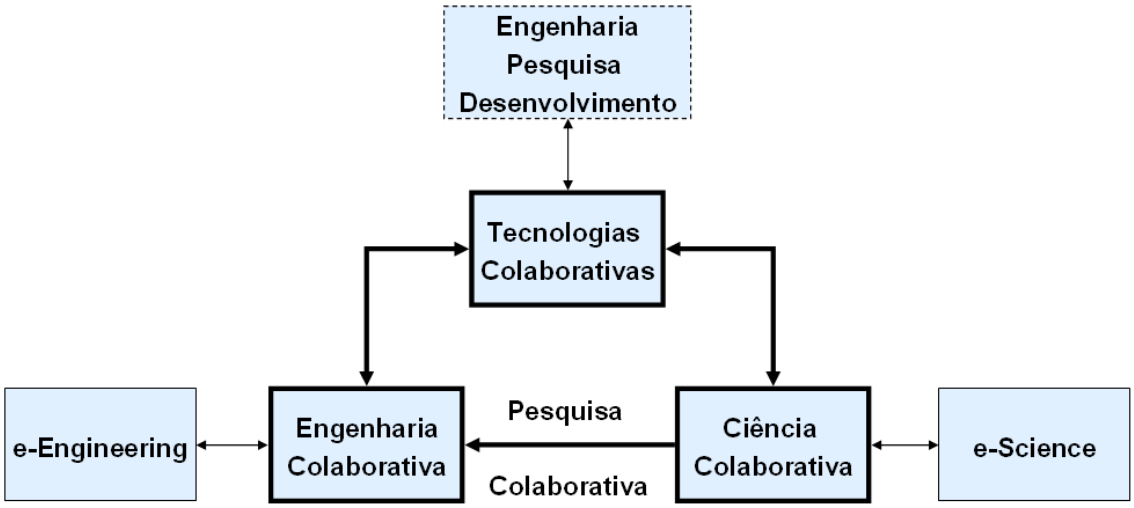


Figura 2.8 - Sinergia entre engenharia e ciência
Fonte: Adaptado de Imeti (2012)

A criação de ambientes de *e-Engineering* se apóia em diversos conceitos, abordados a seguir.

2.2.1. CSCW

CSCW (*Computer Supported Cooperative Work*), ou Trabalho Cooperativo Apoiado por Computador, é o estudo de como pessoas usam a tecnologia, com relação ao hardware e software, para trabalhar em conjunto em tempo e espaço compartilhados (GRUDIN, 1994).

Isto pode ser visto por dois pontos de vista distintos: (1) centrado na tecnologia, colocando ênfase na concepção de formas de projetar tecnologia computacional para melhorar o apoio às pessoas que trabalham juntas; (2) centrado no trabalho, colocando ênfase na compreensão dos processos de trabalho com o objetivo de melhorar o projeto de sistemas computacionais de modo a apoiar o trabalho de grupo (MILLS, 2003).

Sistemas CSCW incluem um ambiente compartilhado com funcionalidades relacionadas à cooperação, coordenação e comunicação para auxiliar grupos de usuários engajados em uma atividade ou objetivo comuns (KULESZA et al.,

2007), sendo seu principal objetivo descobrir maneiras de como a computação pode aumentar a realização de processos de trabalho em grupo face às dimensões de tempo e espaço (PFEIFER, 1995).

Conforme ilustrado na Figura 2.9, existem duas dimensões para CSCW: tempo e espaço. Essas dimensões podem ser divididas em quatro quadrantes, que por sua vez podem ser subdivididos em: síncrono e assíncrono para a dimensão tempo, e co-localizado (mesmo local) e distribuído para a dimensão espaço. A relação estabelecida entre as subdivisões delimita a forma que a informação é compartilhada.

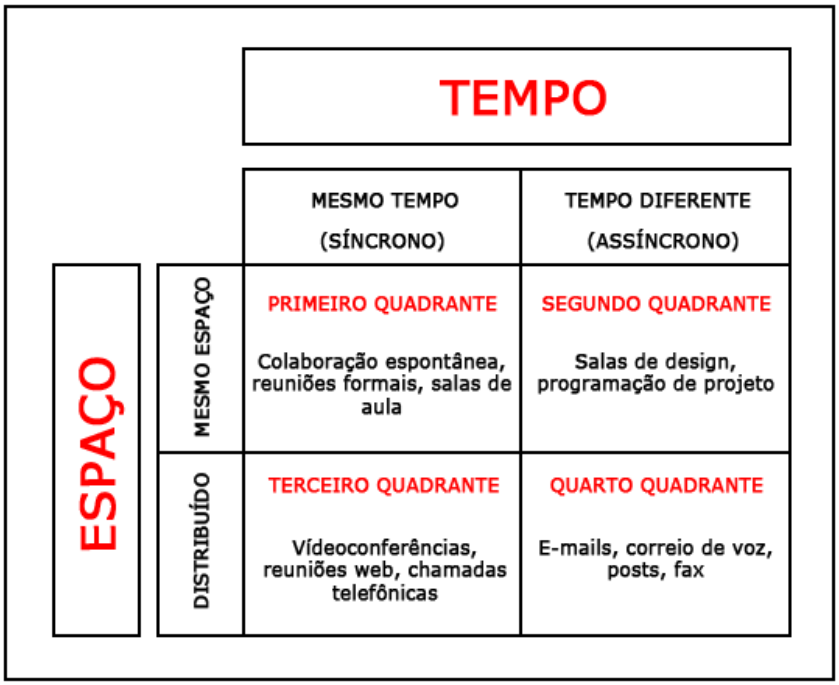


Figura 2.9 - Quadrantes de CSCW

Fonte: Rama e Bishop (2006)

CSCW envolve, portanto, um campo de pesquisa multidisciplinar que se concentra em ferramentas e técnicas para possibilitar que várias pessoas trabalhem em tarefas relacionadas, fornecendo a indivíduos e organizações o suporte à cooperação em grupo e orientação de tarefas em ambientes distribuídos ou em rede (ESERYEL et al., 2002).

2.2.2. Groupware

Groupware são sistemas baseados em tecnologias de computação e telecomunicações que auxiliam grupos de usuários a exercer uma atividade (SOUZA, 2011b). Basicamente, é um tipo de software para apoio ao trabalho cooperativo.

Uma ferramenta *groupware* é extremamente atrativa em ambiente de engenharia, pois, ao trabalhar em grupo, os participantes podem potencialmente produzir melhores resultados do que se atuassem individualmente. Isto ocorre porque há a complementação de capacidades, de conhecimentos e de esforços individuais, e a interação entre pessoas com interpretações, pontos de vista e habilidades complementares. Além disso há maior capacidade de gerar alternativas, levantar vantagens e desvantagens de cada uma, avaliar viabilidades e tomar decisões (FUKS et al., 2003).

Um *groupware* permite que muitos usuários simultâneos trabalhem no mesmo projeto. Enquanto um sistema monousuário se concentra no indivíduo, um *groupware* foca no grupo. Ao se trabalhar em um projeto onde a comunicação é essencial entre colaboradores, um *groupware* facilita de forma mais rápida e mais clara esse intercâmbio de informação. Seu objetivo é permitir múltiplas perspectivas, conhecimentos e assistência na resolução de problemas em conjunto, visando economizar tempo e custos na coordenação do trabalho de grupo (RAMA; BISHOP, 2006).

Um *groupware* utiliza ferramentas que auxiliam pessoas a trabalharem juntas mais fácil e eficazmente, nos seguintes aspectos (BOTELHO; VIDAL, 2005): (1) Comunicação - pessoas compartilham informações; (2) Coordenação - pessoas coordenam seus papéis pessoais com os outros; (3) Colaboração - pessoas trabalham juntas.

As trocas de informação ocorridas durante a comunicação geram compromissos que são gerenciados pela coordenação, que por sua vez

organiza e dispõe as tarefas que são executadas na cooperação. Ao cooperarem, os envolvidos têm necessidade de se comunicarem para renegociar e para tomar decisões. Tal comportamento mostra o aspecto cíclico da colaboração (FUKS et al., 2003).

Algumas características comuns às soluções de *groupware* são (LIMA et al., 2004): correio eletrônico; correio de voz; calendários (permitir a verificação automática do calendário eletrônico dos membros da equipe em busca de horários vagos); agendamentos e programação *on-line* (programar e notificar membros sobre reuniões agendadas); gerenciamento de projetos e atividades; gerenciamento do conhecimento (através da organização e compartilhamento de formulários de informações administrativas, formação de bibliotecas de dados, artigos, Intranets, Extranets, entre outros); mapeamento das áreas de conhecimento dos funcionários; fóruns de discussão; videoconferência; entre outros.

Sistemas colaborativos permitem às pessoas se comunicarem umas com as outras, cooperando sobre projetos e compartilhando informações e conhecimento, independente da localização e do momento. Da mesma forma, facilitam a comunicação informal, a automatização e a redução do tempo na realização das tarefas, permitindo a realização do trabalho em equipe de maneira mais eficaz, eficiente e criativa (LIMA et al., 2004).

2.2.3. E-Science

Como esforços em *e-Engineering* surgiram da área de *e-Science*, esta é descrita sumariamente a seguir.

Segundo Taylor (1999), *e-Science* está relacionada com colaboração global em áreas-chave da ciência e com a próxima geração de infraestrutura computacional que irá possibilitá-la. Taylor viu que muitas áreas da ciência foram se tornando cada vez mais dependentes de novas formas de trabalho colaborativo e multidisciplinar (HEY; TREFETHEN, 2003).

Algumas áreas de pesquisas caracterizam-se pela conformação de um ambiente colaborativo, organizado em rede, e que faz uso intensivo de tecnologias computacionais de processamento, armazenamento e transmissão remota de quantidades massivas de dados. Essa infraestrutura atende às necessidades de pesquisas que, em sua maioria, são realizadas por equipes distribuídas pelo globo e pertencentes a áreas, laboratórios ou instituições distintas. (MANTOVANI; MOURA, 2009). A Figura 2.10 ilustra o atual estado da provisão de informação científica no âmbito de pesquisas e as direções para as quais essa provisão caminha, seguindo a escala evolutiva da *e-Science*.

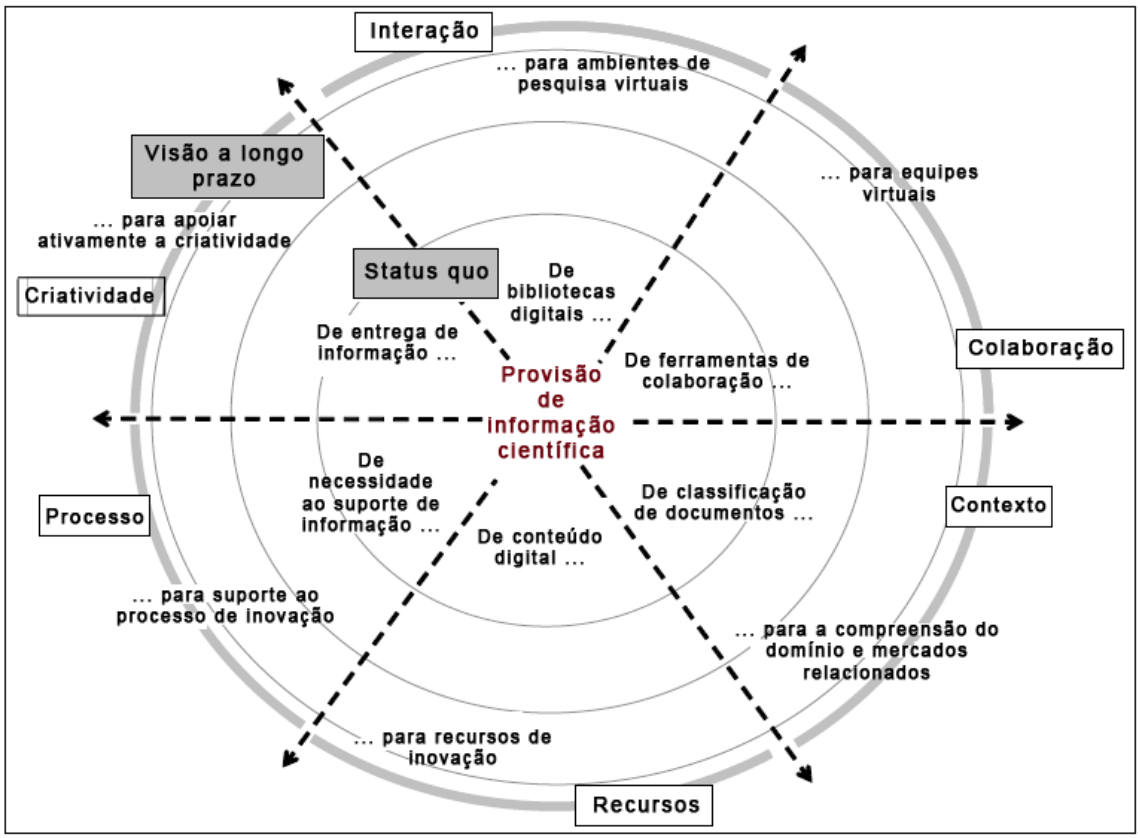


Figura 2.10 - Direções para a próxima geração de suporte a *e-Science*
Fonte: Niederée et al. (2007)

Implicitamente as aplicações de *e-Science* definem um conjunto de serviços computacionais e de dados que a infraestrutura de hardware e *middleware*

deve entregar para permitir essa revolução científica (HEY; TREFETHEN, 2003).

A introdução das tecnologias de informação e comunicação na ciência, em sua versão colaborativa, móvel e em rede, abre a possibilidade de vislumbrar as mudanças ocorridas na produção de conhecimento. A proposta de horizontalização da comunicação científica, ensejada pela *e-Science*, anuncia a possibilidade de incorporação de novos formatos para a difusão científica e exige modelos mais ágeis, levando os pesquisadores a estabelecer novos modelos de interação, conversação e produção em rede (MANTOVANI; MOURA, 2009). Semelhantemente, a *e-Science* é intrinsecamente interdisciplinar, permitindo e promovendo atividades sinergéticas entre diferentes disciplinas científicas ao invés de simplesmente uma única disciplina e Ciência da Computação.

Essa dinâmica de funcionamento em rede passa a exigir um novo tipo de organização da informação e a instituir uma nova discursividade na ciência. A *e-Science*, ao manifestar-se, essencialmente, pelos gêneros digitais no âmbito da comunicação científica, leva os pesquisadores a estabelecer novos modelos de interação, conversação e produção em rede (MANTOVANI; MOURA, 2009).

Em síntese, *e-Science* é o ponto onde a Tecnologia da Informação (TI) encontra cientistas e pesquisadores. É o contexto em que a computação se torna parte integrante e imprescindível para o sucesso na realização de pesquisas das mais variadas áreas; o ponto onde a ciência é realizada com o apoio computacional, tornando-se então, mais eficiente (SILVA et al., 2009).

2.2.4. E-Engineering

O conceito de *e-Engineering* está sendo usado frequentemente em indústrias específicas utilizando tecnologias baseadas em web para permitir desenvolvimento de produtos e resolução de problemas. As ferramentas de engenharia apoiadas pela Internet permitem que os engenheiros trabalhem

diariamente em tempo real mesmo se estiverem geograficamente dispersos (CULLER; GARCÍA, 2004).

Um ambiente de *e-Engineering* é um ambiente computacional que permite às pessoas colaborar e interagir no desenvolvimento de um novo projeto, independentemente de sua localização geográfica e meios de interação. Com base nesta abordagem, ambientes colaborativos devem promover parcerias globais de engenharia bem sucedidas através do uso de tecnologias de informação e comunicação (SHEN, 2003).

Da mesma forma, um *framework* de *e-Engineering* provê integração e automação flexíveis para processos de engenharia em um ambiente distribuído (KUK et al., 2008), objetivando integrar serviços de engenharia e mediá-los, promovendo assim a colaboração no contexto de desenvolvimento do produto (LEE et al., 2010).

Em um projeto *e-Engineering* deve-se, portanto, estudar a interação requerida pelas equipes interdisciplinares a fim de completar o projeto conceitual, o protótipo funcional e atividades de planejamento e produção para entregar o produto final ao mercado (CULLER; GARCÍA, 2004).

O emprego de uma plataforma colaborativa para o planejamento de projetos tem sido identificado como uma ferramenta-chave no gerenciamento do ciclo de vida de produtos de engenharia, principalmente durante o projeto conceitual, pois amplia as capacidades de parceiros de negócios com a junção de conhecimentos de engenharia e outros recursos.

Ferramentas de *e-Engineering* permitem a duas (ou mais) partes apoiar e executar projetos de engenharia durante seu ciclo de vida (MEJÍA et al., 2005). Esta atividade requer a solução de problemas de natureza e escopo diferentes, lida com o desenvolvimento em rede e formas de ser utilizada para satisfazer as necessidades e requisitos da sociedade da informação no contexto da engenharia (KRZEMINSKI; JOZWIAK, 2012).

3 FUNDAMENTAÇÃO TECNOLÓGICA

Este capítulo apresenta conceitos e técnicas que servem de embasamento tecnológico para o desenvolvimento do trabalho. O primeiro tópico descreve as principais características do paradigma da computação orientada a serviços, utilizado para interligar e prover serviços legados aos usuários. O segundo tópico aborda o paradigma da computação em nuvem, utilizado para a criação da nuvem privada e disponibilização dos serviços de software via nuvem.

3.1. Computação Orientada a Serviços

A computação orientada a serviço (SOC – *Service-Oriented Computing*) representa uma nova plataforma de computação distribuída que inclui princípios e padrões de projeto, modelo arquitetural distinto, além de conceitos e tecnologias relacionados. SOC permite a construção de ambientes distribuídos através da integração entre plataformas já existentes e soma de novas camadas, considerações de governança e um vasto conjunto de tecnologias de implementação, normalmente baseado em serviços web (ERL, 2008). É um paradigma para o desenvolvimento e integração de aplicações distribuídas, cujo objetivo de apoiar a interoperabilidade entre componentes em execução sobre plataformas diversas.

Esse paradigma promove a ideia de montar os componentes da aplicação em uma rede de serviços que pode ser fracamente acoplada para criar processos de negócios dinâmicos, flexíveis e aplicações ágeis que abrangem organizações e plataformas de computação, onde funcionalidades ou componentes de aplicativos podem ser reutilizados e transformados em serviços de rede (*web services*) disponíveis (PAPAZOGLOU et al., 2007).

A abordagem “orientada a serviço” é baseada no conceito de composição de aplicações através da descoberta e invocação de serviços disponíveis para realizar alguma tarefa, sendo independente de linguagens de programação ou sistemas operacionais, permitindo que as organizações exponham suas

principais competências numa rede, como por exemplo, a Internet, usando o padrão de linguagens baseadas em XML (*eXtensible Markup Language*) e protocolos, além de uma interface autodescritiva (PAPAZOGLOU et al., 2007).

Essa orientação a serviço almeja a criação de soluções lógicas que estão individualmente moldadas de forma que possam trabalhar coletivamente a fim de auxiliar a realização das metas estratégicas específicas (ERL, 2008). Assim, a computação orientada a serviços permite que desenvolvedores aumentem rápida e dinamicamente as aplicações através da (PAPAZOGLOU et al., 2007): criação de soluções compostas que utilizam ativos de softwares internos da organização, incluindo informações empresariais e sistemas legados; e combinação dessas soluções com componentes externos, possivelmente residentes em redes remotas.

A chave para a realização dessa visão é a Arquitetura Orientada a Serviços (SOA – *Service-Oriented Architecture*), que é uma maneira lógica de concepção de um sistema de software para fornecer serviços tanto para aplicativos de usuário final quanto para outros serviços distribuídos em uma rede, por meio de interfaces publicadas e detectáveis (PAPAZOGLOU et al., 2007).

SOA representa um modelo arquitetônico que objetiva aumentar a agilidade e custo-benefício de um empreendimento enquanto reduz a carga de TI na organização. Isso é feito posicionando serviços como os meios primários pelos quais a lógica de solução é representada. SOA apóia a orientação a serviço na realização das metas estratégicas associada com a computação orientada a serviço (ERL, 2008).

Entre as principais características da computação orientada a serviço estão (TEIXEIRA, 2011): evolução das arquiteturas distribuídas; flexibilidade na organização de seus elementos (serviços); ênfase na interoperabilidade entre componentes heterogêneos, sobre uma base de protocolos padrão abertos e universalmente aceitos, tais como SOAP (*Simple Object Access Protocol*) para

transmissão de dados, WSDL (*Web Services Description Language*) para definição de serviços, UDDI (*Universal Description, Discovery and Integration*) para descoberta de serviços e WS-BPEL (*Web Services Business Process Execution Language*) para orquestração de serviços; independência do mecanismo de comunicação (síncrono ou assíncrono); atenção especial a aspectos de negócio; domínio da tecnologia de *web services*.

Em suma, SOA é uma abordagem arquitetural que visa construir sistemas a partir de um conjunto de componentes fracamente acoplados (serviços, geralmente web) que podem ser combinados dinamicamente. SOA define três papéis, três operações e três padrões de serviços web (ERL, 2004), sendo suas relações demonstradas na Figura 3.1:

- Papéis: *Provider* (Provedor do serviço), *Requestor* (Solicitante do serviço) e *Registry* (Catálogo de serviços).
- Operações: *Publish* (Registro), *Find* (Busca) e *Bind* (Interação/ligação).
- Padrões: SOAP (Comunicação), WSDL (Descrição das interfaces dos serviços) e UDDI (Descoberta de serviços).

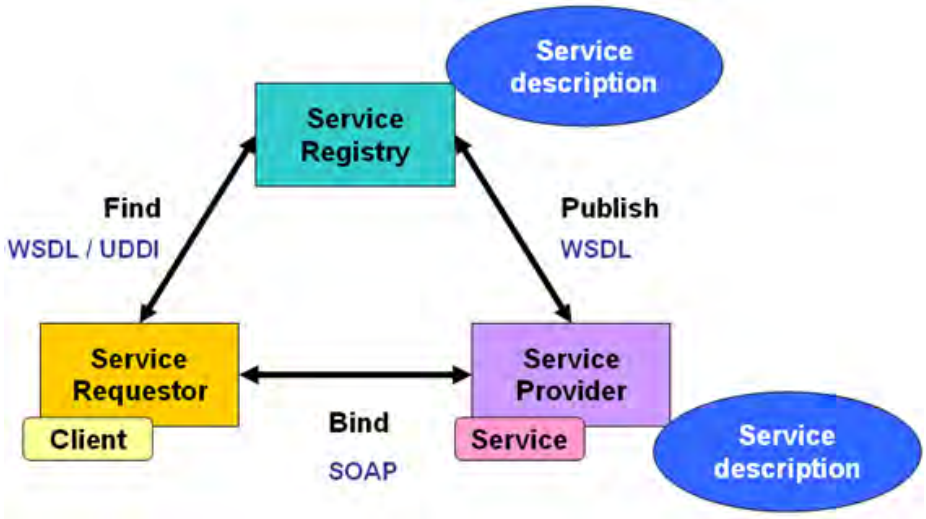


Figura 3.1 - Interações de serviços web

Fonte: Erl (2004)

Como uma forma de arquitetura tecnológica, uma implementação de SOA pode consistir em uma combinação de tecnologias, produtos, APIs (*Application Programming Interface*), apoiando extensões de infraestrutura, e várias outras partes, caracterizando-se tipicamente pela introdução de tecnologias novas e plataformas que especificamente apóiam a criação, execução, e evolução de soluções orientadas a serviço (ERL, 2008).

Outra forma de atingir a orientação a serviço é através do uso de REST (*Representational State Transfer*), um conjunto coordenado de restrições arquiteturais que tenta minimizar latência e comunicação de rede ao mesmo tempo em que maximiza a independência e escalabilidade de implementações de componente (FIELDING, 2000), provendo meios de construir sistemas distribuídos baseado na noção de recursos (partes de informação), que são unicamente identificados por URIs (*Uniform Resource Identifier*). Serviços REST (ou *RESTful services*) são programas projetados com uma ênfase em simplicidade, escalabilidade e usabilidade (ERL, 2008).

Como alternativa aos *web services* que utilizam SOAP, a motivação para o uso de REST está em aplicar as características da Web à Arquitetura Orientada a Serviços (SOA). A Tabela 3.1 ressalta algumas diferenças entre as duas abordagens.

Tabela 3.1 - RESTful web wervices versus SOAP web services

	REST	**SOAP**
Formato da Mensagem	XML	XML dentro de envelope SOAP
Definição da Interface	Nenhuma	WSDL
Transporte	HTTP	HTTP, FTP, MIME, SMTP, outros

Fonte: Hansen (2007)

REST expõe recursos em forma de documentos XML acessíveis por URIs que usam uma interface uniforme do protocolo HTTP (*HyperText Transfer Protocol*) (SZEPIELAK, 2006). Por não haver linguagem de definição para as interfaces RESTful, o acesso aos serviços pode ser feito de duas maneiras, conforme demonstrado na Figura 3.2 (CHAPPEL, 2009): (1) aplicação-cliente escreve chamadas HTTP *raw* (formato bruto), ou (2) o serviço RESTful provê uma biblioteca-cliente; cliente vê métodos com parâmetros.

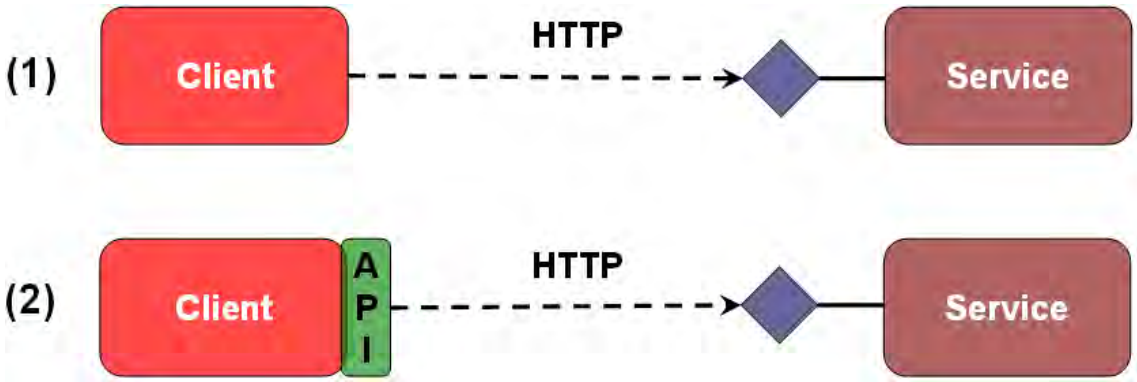


Figura 3.2 - Acesso via REST
Fonte: Chappel (2009)

Este conceito é fundamentalmente diferente do procedimento baseado em RPC (*Remote Procedure Call*) que encapsula a idéia de chamar um procedimento no servidor remoto e normalmente contêm informações sobre o procedimento a ser invocado ou ação a ser tomada. Estas informações são referidas como verbos em um pedido de serviço web (HANSEN, 2007).

Cada operação pode ser mapeada para um dos verbos (métodos) de HTTP, sendo o *POST* para criar (*Create*), o *GET* por recuperar seus detalhes (*Read*), o *PUT* para atualizar (*Update*), e o *DELETE* para apagar (*Delete*), conjunto frequentemente referido pelo acrônimo CRUD (*Create, Read, Update, Delete*), e documentos XML podem ser usados para prover uma representação de recurso uniforme (WEBBER et al., 2010).

Ao usar HTTP como protocolo de aplicação ao invés de um protocolo de transporte, é possível perceber que a Web é realmente um grande *framework* para construção de sistemas distribuídos. Quando mesclados com códigos de estado apropriados e alguns padrões de senso comum, HTTP pode prover uma boa plataforma para domínios de CRUD, usado para manipular os recursos pela rede, resultando em arquiteturas realmente simples (WEBBER et al., 2010).

REST descreve a Web como uma aplicação de hipermídia distribuída cujos recursos "linkados" se comunicam trocando recursos de representações de estado (WEBBER et al., 2010). Por seu estilo arquitetônico ser uma abordagem centrada em recurso, é simples e pode ser mapeado de forma natural (SZEPIELAK, 2006).

A computação orientada a serviços baseia-se então em arquiteturas de software orientadas a serviços, sendo uma maneira de reorganizar aplicações e infraestruturas de software em um conjunto de serviços interativos. Esse estilo arquitetônico de fazer software promove o fraco acoplamento entre os componentes e a independência de tecnologias específicas para que eles sejam reutilizáveis, tendo como principais vantagens a adaptação das aplicações a tecnologias em constante evolução, a fácil integração com outros sistemas, o reaproveitamento de investimentos em sistemas legados e a criação prática e rápida de novos processos a partir de serviços existentes (GROSSI, 2005).

3.1.1. ESB - Enterprise Service Bus

Um ESB (*Entreprise Service Bus*), ou Barramento de Serviços Corporativos, é a infraestrutura que possibilita aos consumidores invocar os serviços oferecidos pelos fornecedores, provendo interoperabilidade (JOSUTTIS, 2008). Representa um ambiente projetado para promover interconectividade sofisticada entre serviços, estabelecendo uma camada intermediária de

processamento que pode ajudar a superar problemas comuns associados com confiança, escalabilidade e disparidade de comunicações (ERL, 2008).

O ESB é uma camada de software que permite que programas diferentes ”conversem” uns com os outros em uma maneira padronizada, mesmo se estiverem executando em sistemas operacionais diferentes e escritos em linguagens de programação diferentes (HURWITZ et al., 2009).

Segundo JOSUTTIS (2008), dependendo das abordagens técnicas e organizacionais para se implementar um ESB, este pode envolver as seguintes responsabilidades (entre outras): prover conectividade; transformação de dados; roteamento (inteligente); lidar com segurança; lidar com confiabilidade; gerenciamento de serviços; monitoramento e *logging;* entre outros. Um exemplo típico de um ESB é mostrado na Figura 3.3, onde a separação lógica em camadas e a exposição de funcionalidades são exibidas.

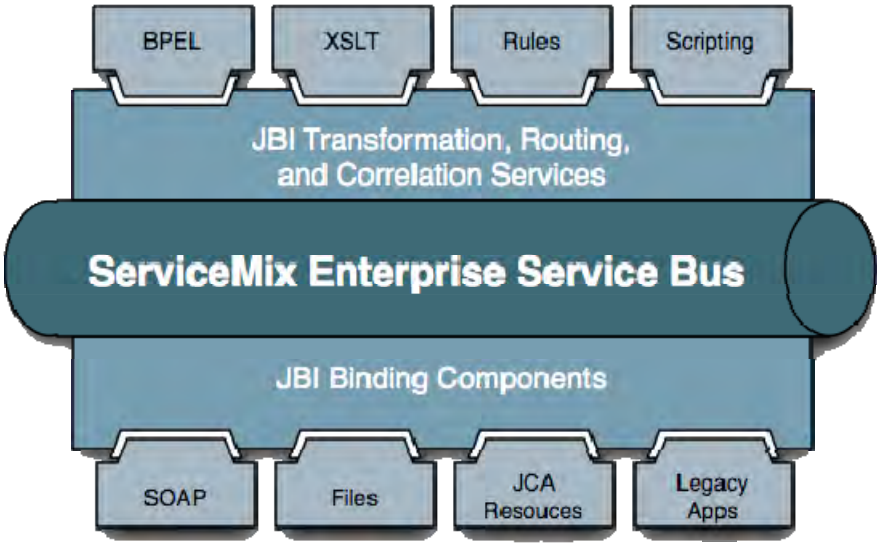


Figura 3.3 - Exemplo de um ESB: Apache ServiceMix
Fonte: Apache (2011)

Um exemplo de caso de uso para o ESB é agir como uma camada intermediária entre um portal de servidor e as fontes de dados com as quais o

portal do servidor deve interagir, sendo que este é um ponto de agregação voltado para o usuário, contendo vários recursos representados como serviços (PAPAZOGLOU et al., 2007).

3.2. Computação em Nuvem

O termo nuvem (*cloud*) e sua imagem vieram originalmente da telefonia e posteriormente foram adotados como metáfora para descrever a Internet nos diagramas de rede. As linhas desenhadas nos diagramas cruzavam por dentro da nuvem para indicar que o fluxo de dados simplesmente passava pela Internet (TAURION, 2009).

Atualmente, com a Computação em Nuvem (*Cloud Computing*), a nuvem deixou de representar algo teoricamente inatingível e passou a ser um ambiente de computação, onde aplicações podem usar recursos da nuvem ou podem se executadas a partir da mesma (TAURION, 2009).

Este ambiente de computação é baseado em uma rede de servidores, sejam físicos ou virtuais, sendo então um conjunto de recursos como capacidade de processamento, armazenamento, conectividade, plataformas, aplicações e serviços disponibilizados na Internet (TAURION, 2009).

A virtualização é a simulação do software e/ou hardware sobre a qual outro software é executado. Este ambiente simulado é chamado de VM (*Virtual Machine*) ou simplesmente máquina virtual (SCARFONE et al., 2011). A tecnologia de virtualização provê o software *hypervisor* que permite que múltiplos sistemas operacionais convidados (*guest*) executem simultaneamente no mesmo servidor físico. Esta capacidade habilita a capacidade multi-inquilino (*multi-tenancy*), que é o compartilhamento de recursos físicos visando melhor utilização de servidor (PIZETTE; RAINES, 2010).

Através desta infraestrutura é possível utilizar as mais variadas aplicações via Internet, em qualquer lugar e independente de plataforma, com a mesma

facilidade de tê-las instaladas no computador pessoal, podendo se tornar um *datacenter* remoto, centro de dados ou conjunto de servidores onde o aplicativo é armazenado (VELTE et al., 2011). Este paradigma da computação é composto basicamente por quatro modelos de implantação (tipos de nuvens) e por três modelos de serviços, abordados em detalhes a seguir.

3.2.1. Modelos de Implantação

Modelos de implantação caracterizam a gestão e disposição dos recursos computacionais para prestação de serviços para os consumidores, bem como a diferenciação entre as classes de consumidores (JANSEN; GRANCE, 2011). Em geral, existem quatro tipos de nuvens ou modelos de implantação: Nuvem Pública (*Public Cloud*), Nuvem Privada (*Private Cloud*), Nuvem Comunitária (*Community Cloud*) e Nuvem Híbrida (*Hybrid Cloud*).

3.2.1.1. Nuvem Pública

Uma nuvem pública (*public cloud*) é um modelo em que a infraestrutura e recursos computacionais que a compreende são postos à disposição do público em geral através da Internet. Ele pertence e é operada por um provedor de nuvem que oferece serviços em nuvem para os consumidores e, por definição, é externo às organizações dos consumidores (JANSEN; GRANCE, 2011).

Nesse modelo de implantação a infraestrutura de nuvem é disponibilizada para o público em geral, sendo acessado por qualquer usuário que conheça a localização do serviço, não podendo ser aplicadas restrições de acesso quanto ao gerenciamento de redes, e menos ainda, utilizar técnicas para autenticação e autorização (SOUSA et al., 2009), sendo estas responsabilidades do fornecedor. Os recursos são fornecidos dinamicamente pela Internet, através de aplicações e/ou *web services*, geridos e fornecidos por uma organização externa.

As nuvens públicas fornecem uma maneira para reduzir o risco e o custo para o cliente, fornecendo uma extensão flexível, ainda que temporária, para infraestrutura corporativa. Se uma nuvem pública é implementada com o desempenho, segurança e localização de dados em mente, a existência de outros aplicativos executados na nuvem deve ser transparente para ambos os arquitetos de nuvens e usuários finais. De fato, um dos benefícios de nuvens públicas é que elas podem ser muito maiores que nuvens privadas, oferecendo a capacidade de escalar para cima ou para baixo na demanda e transferindo os riscos de infraestrutura da empresa para o provedor de nuvem, ainda que apenas temporariamente (SUN, 2009).

3.2.1.2. Nuvem Privada

Uma nuvem privada (*private cloud*) é aquela em que o ambiente de computação é operado exclusivamente para uma única organização, podendo ser gerido por ela própria ou por terceiros, e pode ser hospedado no centro de dados próprio (local) ou fora dela (remota). Esse modelo tem o potencial de dar à organização maior controle sobre os consumidores da infraestrutura, recursos computacionais do que poderia uma nuvem pública (JANSEN; GRANCE, 2011). As técnicas utilizadas para prover tais características podem ser em nível de gerenciamento de redes, configurações dos provedores de serviços e a utilização de tecnologias de autenticação e autorização (SOUSA et al., 2009), sendo uma opção considerável às organizações que necessitam de segurança e confidencialidade dos dados.

Nuvens privadas são construídas para o uso exclusivo de um cliente, proporcionando o máximo controle sobre os dados, segurança e qualidade de serviço. A organização possui a infraestrutura e tem controle sobre como os aplicativos são implantados sobre ela, podendo ser implantadas em um *datacenter* empresarial ou podem ser implantadas localmente (SUN, 2009).

Entretanto, o modelo de computação em nuvem não significa sempre que os clientes têm que atravessar a Internet para chegar ao conteúdo. Uma nuvem

local, também conhecida como virtualização de apresentação, ignora o componente fornecedor de serviços e permite gerenciar todos os conteúdos no próprio centro de dados, sem o compromisso de terceirização e trazendo a nuvem tão próxima quanto possível (VELTE et al., 2011). Nesse sentido, quando a infraestrutura subjacente está dentro da própria organização e é gerida por esta, tem-se o conceito de Nuvem Privada Local.

Com uma nuvem local, é possível manter o servidor “em casa” e os clientes se conectam a ele. Começa-se a oferecer recursos de computação para os usuários como um utilitário. Ironicamente, algumas organizações usam a computação em nuvem para fornecer Intranet corporativa. Intranets normalmente são utilizadas dentro de uma organização e não são acessíveis ao público. Ou seja, um servidor web é mantido *in-house* e informações sobre a empresa são mantidas na empresa para que possam ter acesso. No entanto, agora, Intranets estão sendo mantidas na nuvem; e para ter acesso às informações locais privadas da organização, os usuários têm que fazer *logon* na Intranet indo a um local público seguro (VELTE et al., 2011).

Trata-se, então, de uma infraestrutura de nuvem dentro dos limites físicos de uma organização (KOOPMANS, 2010), sendo um ambiente que é capaz de executar e implementar características da computação em nuvem como virtualização e serviços em camadas sobre a rede, mas ao mesmo tempo também se aplicam políticas mais rígidas e requisitos como segurança, latências, acordos de nível de serviço (SLA – *Service Level Agreement*) e também o uso de recursos de *datacenters* existentes. Isso proporciona eficiência e flexibilidade ao mesmo tempo em que reduz os custos.

Uma nuvem privada oferece muitos dos benefícios de um ambiente público de computação em nuvem. A diferença entre uma nuvem privada e uma nuvem pública é que em um serviço baseado em nuvem privada, os dados e processos são gerenciados dentro da organização, sem as restrições de largura de banda de rede, exposições de segurança e requisitos legais que o

uso de serviços de nuvem pública pode acarretar. Além disso, os serviços em nuvem privadas oferecem ao provedor e ao usuário maior controle da infraestrutura de nuvem, melhorando a segurança e resiliência, pois o acesso do usuário e as redes utilizadas são restritos e designados (BADGER et al., 2011).

3.2.1.3. Nuvem Comunitária

Uma nuvem comunitária (*community cloud*) situa-se entre as nuvens públicas e privadas no que diz respeito ao conjunto-alvo de consumidores. É algo semelhante a uma nuvem privada, mas a infraestrutura e recursos computacionais são exclusivos para duas ou mais organizações que têm privacidade, de segurança, e considerações regulatórias, ao invés de uma única organização (JANSEN; GRANCE, 2011).

Nesse modelo de implantação ocorre o compartilhamento por diversas empresas de uma nuvem, sendo esta suportada por uma comunidade específica que partilhou seus interesses, tais como missão, requisitos de segurança, política e considerações sobre flexibilidade. Este tipo de modelo pode existir local ou remotamente, e geralmente é administrado por alguma empresa da comunidade ou por terceiros (SOUSA et al., 2009).

3.2.1.4. Nuvem Híbrida

A nuvem híbrida (*hybrid cloud*) é mais complexa do que os outros modelos de implantação, pois envolve uma composição de duas ou mais nuvens (comunitária, privada ou pública). Cada membro continua sendo uma entidade única, mas está ligado aos outros através da tecnologia padronizada ou proprietária que permite a portabilidade de aplicações e de dados entre eles (JANSEN; GRANCE, 2011).

Uma nuvem híbrida pode também ser usada para lidar com picos de carga de trabalho previstos, ou seja, uma nuvem pública pode ser usada para executar

tarefas periódicas que podem ser implementadas facilmente. Essas nuvens introduzem a complexidade da determinação de como distribuir aplicações em ambas as nuvens, pública e privada (SUN, 2009).

3.2.2. Modelos de Serviços

O modelo de serviço com o qual uma nuvem está em conformidade determina o âmbito e o controle sobre o ambiente computacional de uma organização. Este modelo caracteriza um nível de abstração para sua utilização, podendo ser utilizado sob qualquer um dos modelos de implantação (JANSEN; GRANCE, 2011).

Nesse sentido, três modelos de serviços bem conhecidos e frequentemente utilizados são abordados: IaaS (*Infrastructure as a Service*), PaaS (*Plataform as a Service*) e SaaS (*Software as a Service*), conforme exibido na Figura 3.4.

Figura 3.4 - Pilha da computação em nuvem
Fonte: Adaptado de Kepes (2011)

3.2.2.1. IaaS

Infraestrutura como um serviço (IaaS – *Infrastructure as a Service*) é a capacidade que o provedor tem de oferecer uma infraestrutura de processamento e armazenamento de forma transparente para o cliente. Os usuários não têm o controle da infraestrutura física, mas, através de

mecanismos de virtualização, possuem controle sobre as máquinas virtuais, armazenamento, aplicativos instalados e possivelmente um controle limitado dos recursos de rede (VERAS, 2012).

Neste cenário, utilizam-se as máquinas fornecedoras de computação em nuvem, utilizando um servidor virtualizado e executando o software nele (VELTE et al., 2011). É um modelo de prestação de serviços através do qual a infraestrutura de computação básica de servidores, software e equipamentos de rede é fornecido como um serviço sob demanda sobre a qual uma plataforma para desenvolver e executar aplicações pode ser estabelecida. Sua principal finalidade é evitar a compra, hospedagem e gestão do hardware básico e componentes de software, obtendo esses recursos como objetos virtualizados controláveis através de uma interface de serviço. O consumidor geralmente tem ampla liberdade para escolher o sistema operacional e ambiente de desenvolvimento para ser hospedado (JANSEN; GRANCE, 2011).

IaaS é basicamente deixar de adquirir hardware e software básico e passar a desenvolver a aplicação em uma infraestrutura virtual baseada na Internet e adquirida na forma de serviço, tratando da utilização da infraestrutura, normalmente de terceiros, como serviço, sendo que os requisitos de banda, latência, poder de processamento são substituídos por garantias do nível de serviço trazidos por disponibilidade e desempenho adequados (VERAS, 2012).

A implantação de infraestrutura na forma de serviço tem se mostrado propícia para muitas organizações iniciando na criação de nuvem privada pelo fato da infraestrutura residir no próprio cliente. Por fornecer um ambiente de computação genérico, permite criar e implantar qualquer tipo de aplicação, tanto nova quanto legada, portada para executar sobre ela.

3.2.2.2. PaaS

Plataforma como um serviço (PaaS – *Plataform as a Service*) tem seu conceito vinculado ao uso de ferramentas de desenvolvimento de software oferecidas

por provedores de serviços, onde os desenvolvedores criam as aplicações e as desenvolvem utilizando a Internet como meio de acesso, ou seja, o fornecedor oferta a plataforma de desenvolvimento (VERAS, 2012).

PaaS é uma forma de construir aplicações e tê-las hospedadas em um provedor de nuvem. Este modelo permite implantar aplicações sem ter custos para comprar servidores e armazená-los (VELTE et al., 2011). Dessa forma, é um modelo de prestação de serviços através do qual a plataforma de computação é fornecida como um serviço sob demanda na qual as aplicações podem ser desenvolvidas e implementadas. Seu principal objetivo é reduzir o custo e a complexidade de compra, hospedagem e gestão do hardware subjacente e componentes da plataforma de software, incluindo qualquer programa necessário e ferramentas de desenvolvimento de banco de dados. O ambiente de desenvolvimento é tipicamente para um propósito específico, determinado pelo provedor de nuvem e adaptados ao projeto e arquitetura da plataforma desejada. O consumidor tem controle sobre os aplicativos e configurações de aplicativos do ambiente da plataforma (JANSEN; GRANCE, 2011).

O modelo de plataforma como um serviço, então, se propõe a criar uma plataforma para o desenvolvimento de aplicações já voltadas para computação em nuvem, tendo a seguinte definição: uma plataforma para criar e operar aplicações, incluindo ferramentas de desenvolvimento, administração e gerenciamento, além de serviços *runtime*, tudo na modalidade SaaS (TAURION, 2009).

Na plataforma ofertada rodam os aplicativos e se armazenam os dados. A grande diferença em relação a um modelo convencional de terceirização é que a plataforma roda em *datacenters* (centro de dados) de provedores externos e é acessada via Internet, sendo que os desenvolvedores estão do outro lado da rede (VERAS, 2012).

3.2.2.3. SaaS

Software como um serviço (SaaS – *Software as a Service*) é uma modalidade de serviços em nuvem onde os aplicativos de interesse para uma grande quantidade de usuários passam a ser hospedados na nuvem como uma alternativa ao processamento e armazenamento local. Os aplicativos são oferecidos como serviços por provedores e acessados pelos clientes por aplicações como o *browser*. Todo controle e gerenciamento da rede, sistemas operacionais, servidores e armazenamento é feito pelo provedor de serviço (VERAS, 2012).

SaaS é simplesmente um provedor de nuvem que fornece uma determinada parte de software que deseja-se utilizar, em seus servidores. Ao contrário do PaaS, em que os usuários desenvolvem sua própria aplicação, no SaaS essa aplicação é fornecida pelo provedor (VELTE et al., 2011). Assim, o software é disponibilizado através da rede, como um serviço, conforme ilustrado na Figura 3.5, chegando ao usuário final via navegador web.

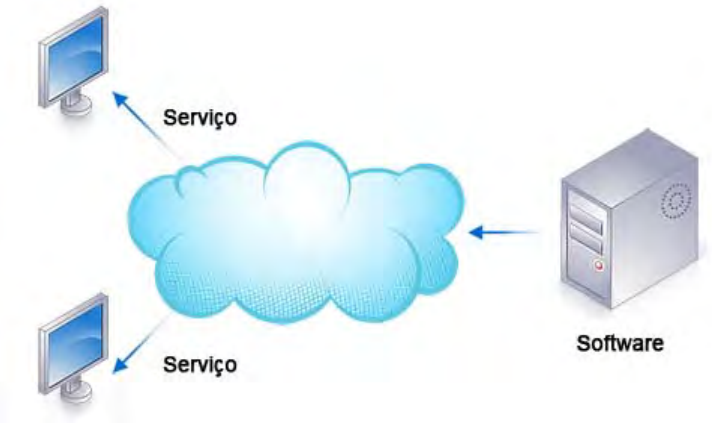


Figura 3.5 - Software como um serviço

SaaS é uma espécie de evolução do conceito de ASP (*Application Service Provider*), que forneciam aplicativos empacotados aos usuários pela Internet, havendo, então, mais pontos em comum com os aplicativos tradicionais *on-premise* (instalados localmente), como licenciamento e arquitetura, do que com

a proposta dos novos aplicativos baseados em software como serviço (VERAS, 2012).

Esse é um modelo de prestação de serviços pelo qual uma ou mais aplicações e os recursos computacionais para executá-las são fornecidos para uso sob demanda como um serviço. Seu principal objetivo é reduzir o custo total de desenvolvimento de software e hardware, manutenção e operações. O consumidor não administra ou controla a infraestrutura subjacente ou aplicativos individuais, exceto para as seleções de preferência e limitadas configurações administrativas dos aplicativos (JANSEN; GRANCE, 2011).

Diferentemente do modelo de licenciamento único, normalmente usado para software instalado no local, o acesso ao aplicativo SaaS é feito pela Internet, via navegador, ou seja, o provedor hospeda o aplicativo, os dados e implanta *patches* e atualizações de modo centralizado e transparente, possibilitando o acesso. Esses aplicativos que são de interesse para uma grande quantidade de usuários/clientes, passam a ser hospedados na nuvem como uma alternativa ao processamento local, sendo oferecidos como serviços. (VERAS, 2012).

Quando utilizado no modelo de nuvem privada, o usuário pode executar aplicativos sob demanda e os dados são mantidos “em casa”. Tal combinação é extremamente atrativa quando: (1) é preciso trabalhar com informações restritas e há a preocupação de que fornecedores externos de SaaS não sejam seguros; (2) há a necessidade de integração com sistemas legados restritos e conexões com sistemas remotos ou aberturas no *firewall* para acesso externo não são desejadas; (3) a transferência de arquivos grandes gera latência e os atrasos resultantes das velocidades de conectividade implicam em mais custos; (4) o pagamento de taxas de uso de SaaS não reflete benefícios em relação a viabilidade financeira a longo e médio prazo.

Nessa situação combinada os riscos são mínimos porque se os requisitos e restrições iniciais não forem atendidos, pode-se simplesmente cancelá-la.

3.2.3. Comparação entre os modelos da Computação em Nuvem

Uma comparação entre os modelos de implantação da computação em nuvem em relação a gerenciamento, infraestrutura, localização e acesso é descrita na Tabela 3.2.

Tabela 3.2 - Comparação entre os modelos de implantação

	Gerenciada por	Infraestrutura da	Infraestrutura localizada	Acessível e consumida por
Pública	Provedor terceiro	Provedor terceiro	Off-Premise	Consumidor não confiável
Privada/ Comunitária	Organização Provedor terceiro	Organização Provedor terceiro	On-Premise Off-Premise	Consumidor confiável
Híbrida	Ambos: Organização & Provedor terceiro	Ambos: Organização & Provedor terceiro	Ambos: On-Premise & Off-Premise	Consumidor confiável & Consumidor não confiável

Fonte: Rational Survivabilit (2009)

O gerenciamento inclui governança, operações, segurança, políticas, e outros aspectos administrativos. A infraestrutura implica estrutura física básica como instalações, computadores, redes e equipamentos de armazenamento. A localização da infraestrutura pode ser física e/ou relativa sob o aspecto do gerenciamento da organização, confrontando propriedade com controle (RATIONAL SURVIVABILIT, 2009).

Consumidores confiáveis dos serviços são aqueles considerados parte de uma organização sob o aspecto legal/contratual/político, incluindo funcionários, contratados e parceiros de negócios, ao passo que consumidores não confiáveis são aqueles que podem ser autorizados a consumir alguns/todos

serviços, mas não são extensões lógicas da organização (RATIONAL SURVIVABILIT, 2009).

A Figura 3.6 descreve o relacionamento entre os papéis existentes no modelo de serviços de computação em nuvem.

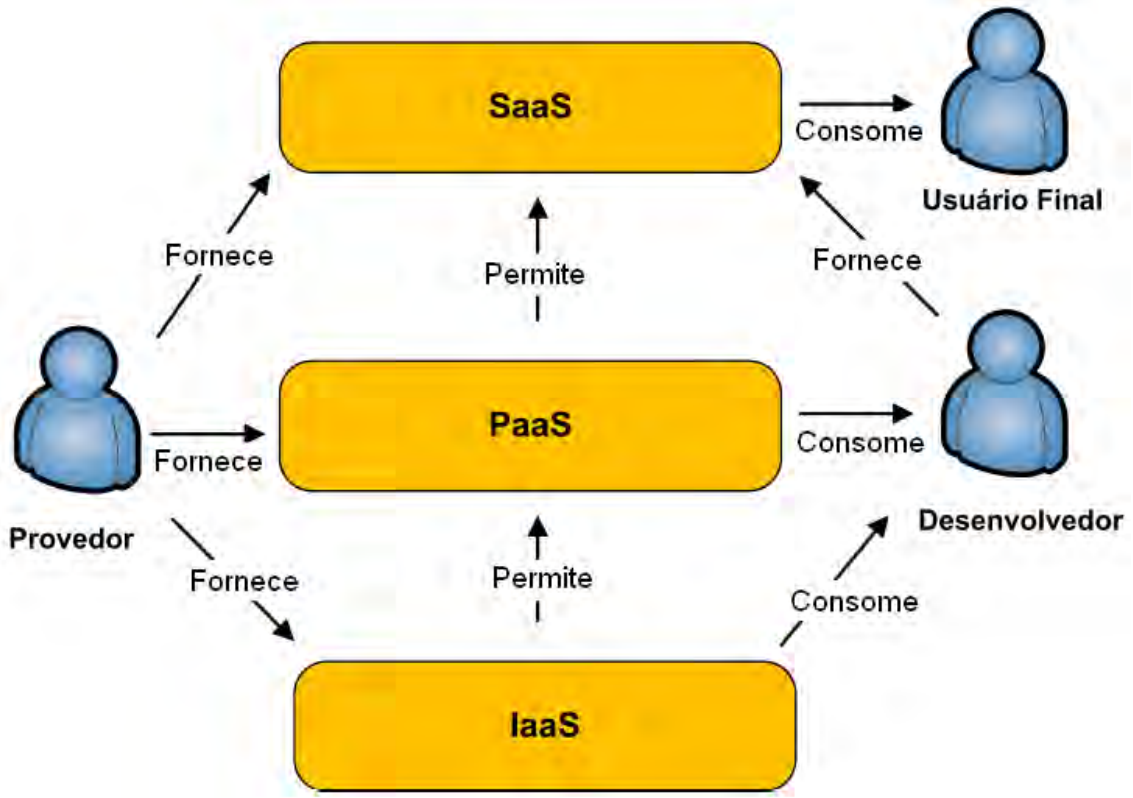


Figura 3.6 - Papéis na computação em nuvem
Fonte: Adaptado de Sousa (2009)

IaaS fornece recursos computacionais, seja de hardware ou software, para PaaS, que por sua vez fornece recursos, tecnologias e ferramentas para desenvolvimento e execução dos serviços implementados a serem disponibilizados como SaaS. É importante, nesse aspecto, ressaltar que uma organização provedora de serviços não precisa obrigatoriamente disponibilizar os três modelos (VERAS, 2012).

A Figura 3.7 descreve os componentes dos serviços de computação em nuvem e as responsabilidades de cliente/usuário e provedor, seguindo os modelos de serviços.

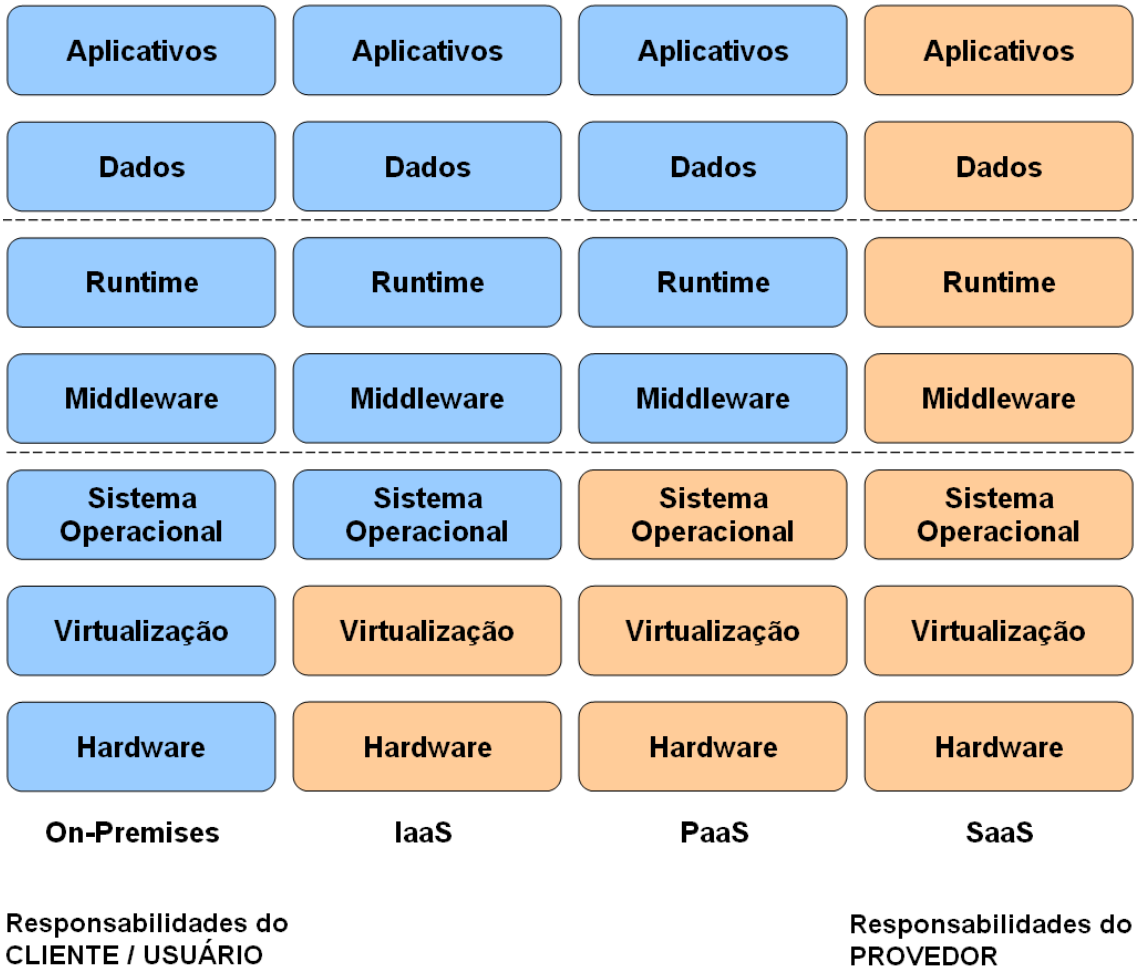


Figura 3.7 - Componentes dos serviços na computação em nuvem
Fonte: Veras (2012)

Nesse contexto, *runtime* é o software responsável pela execução dos programas e *middleware* é a designação genérica utilizada para se referir aos softwares que são executados entre as aplicações e os sistemas operacionais, cujo objetivo é facilitar o desenvolvimento de aplicações, tipicamente as distribuídas, assim como facilitar a integração de sistemas legados ou desenvolvidos de forma não integrada (VERAS, 2012).

4 TRABALHOS RELACIONADOS

Objetivando o trabalho colaborativo em engenharia de sistemas, diversas iniciativas de aplicações vêm sendo desenvolvidas, incluindo a área espacial, conforme citado a seguir. Também são abordados os sistemas legados, SatBudgets e SpaceESB, ferramentas para geração de balanços de disciplinas de engenharia de sistemas espaciais, utilizadas para compor a solução final proposta neste trabalho e prover seus respectivos serviços aos usuários.

4.1. Trabalhos anteriores com ferramentas colaborativas

DEVICE (*Distributed Environment for Virtual Integrated Collaborative Engineering*), desenvolvida pela Thales Alenia Space para apoiar as equipes de engenharia em projetos, resenhas, investigação de anomalia, integração, planejamento, testes e avaliações, é baseada em COTS (*Commercial Off-The-Shelf*), OSS (*Open Source Software*) e aplicações *in-house*, proporcionando interação dos usuários com produtos virtuais, ambientes virtuais, realidade virtual imersiva, dispositivos de interação, simulação física, metodologias de sistemas de engenharia baseados em modelos, modelagem de sistema de dados, gestão do conhecimento, mineração de dados, modelagem baseada em SysML/UML, compartilhamento de dados, plataforma de troca de dados, persistência de dados de projetos, análise de impacto multi-nível, entre outros (GIORGIO et al., 2012).

ALBATRos (*Automated Listing Budgeting And Tracing Repository*) é uma ferramenta baseada na web desenvolvida durante o segundo programa do projeto YES2 (*Young Engineers' Satellite*) da ESA (*European Space Agency*) e que apóia a engenharia de sistemas e QA/PA (*Quality Assurance/Product Assurance)* no projeto de satélites de forma integrada e intuitiva, juntando todas as funções do sistema de engenharia necessárias e elementos em uma aplicação, acessível em todos os momentos para todos os membros da equipe. Entre suas *features* estão: consulta e busca de dados; entrada de dados e recuperação, incluindo listagem de relações entre itens e registro de

alterações; árvore gráfica de relações (forma visual de apresentar as informações no banco de dados); compartilhamento de dados; saída de documentos automatizada; entre outros (HAMBLOCH at al., 2007).

NextGenRE (*Next Generation Requirements Engineering*) é uma ferramenta também patrocinada pela ESA que desenvolveu uma abordagem inovadora com o uso da tecnologia *wiki* semântico para permitir que os usuários especifiquem requisitos estruturados e semanticamente ricos associados a projetos de sistemas em SysML. A reutilização de requisitos através de modelos também é permitida, assim como o raciocínio semântico para verificar propriedades como consistência e coerência entre os requisitos. A abordagem é inteiramente baseada em modelo, permitindo integração da engenharia de requisitos e processos de design, aplicação de verificação de modelos, técnicas de simulação; gestão de requisitos; compartilhamento e troca de dados; gestão do conhecimento; alocação e rastreabilidade de requisitos; gestão de documentos; *templates* de requisitos; reutilização de requisitos; extensões semânticas em *wiki*; acoplamento entre requisitos e design através de *gateway* para SysML; entre outros (DE KONING et al., 2012).

TeamProject é uma aplicação para utilização de tecnologias Web para engenharia de sistemas em projetos mecânicos e campos de análise estrutural baseada em software open source em um conjunto de scripts em XML e PHP para estruturar e armazenar dados. Um dos valores agregados oferecidos por essa versão web é a tradução de dados CAD (*Computer Aided Design*) e FEA (*Finite Element Analysis*) em formato VRML (*Virtual Reality Modeling Language*), além de estender a revisão de dados 3D de produtos para todos os membros da equipe envolvidos em um projeto, provendo um acesso simples e eficiente de desenho mecânico e análise de dados estruturais em uma abordagem de engenharia colaborativa via navegador. Suas principais funcionalidades são: compartilhamento de arquivos e mensagens; gestão de projetos; gestão de documentos; gestão de tarefas; visualização de dados

CAD, FEA e VRML; invocação de processamento e transformação de dados no servidor; entre outros (EYNARD et al., 2005).

GEPPnet Conceitual é uma aplicação baseada numa versão anterior, GEPPnet Informacional, estendida com ferramentas gratuitas como o servidor de aplicações Apache Tomcat, MySQL e linguagem Java, a fim de apoiar o trabalho distribuído durante a fase de projeto conceitual do PDP (Processo de Desenvolvimento de Produto). Entre suas funcionalidades destacam-se: controle de acesso; grupos de usuários; *blogs* dentro do próprio sistema; gestão de projetos; histórico de alterações e atividades; gestão do conhecimento; quadro de mensagens; gestão de documentos; notas; calendário; ferramenta de modelagem conceitual como Matriz Morfológica, Estruturação Funcional, Fluxogramas Funcionais, Matriz de Decisão, entre outras; status de projetos; avaliação e seleção de estrutura funcional; diagramas de Gantt; fórum; compartilhamento de dados; entre outras (GUERRA, 2007).

Engineering Cockpit é uma plataforma de colaboração estilo-rede-social para automação de projetos de engenharia de sistemas voltada para gerentes e engenheiros, proporcionando um papel específico de ponto de entrada único para acompanhamento, monitoramento, colaboração e gestão de projetos. Como resultado, aumenta a percepção da equipe de engenheiros e fornece informações específicas de projeto além das fronteiras das disciplinas de engenharia. Possui suporte a dois tipos de papéis (gerentes e desenvolvedores); funções e capacidades específicas do contexto de interação; definição de marcos (*milestones*); criação de novas questões para solicitações de recursos; informações de monitoramento refletindo o estado atual do projeto; calendário; e-mail; *chat*; notas; compartilhamento de dados; entre outros (MOSER et al., 2011).

Em MACEDO (2008) é apresentado o desenvolvimento de um protótipo com base no levantamento dos requisitos da plataforma colaborativa para gestão de

projetos baseada no Plone, um sistema de gestão de conteúdos escrito na linguagem Python e executa em um servidor com aplicações Zope. Entre suas funcionalidades pode-se destacar: sistema de permissões; grupos de usuários; gestão de projetos; calendário; *plugin* Skype; sessão de perguntas mais frequentes (FAQ - *Frequently Asked Questions*); *newsletters*; *templates*; *chat*; *bookmarks*; gestão de eventos; e-mail; gestão de documentos; flash vídeo *player*; boletins informativos; lista de contatos; progresso e status de projetos, incluindo *milestones*; diagramas Gantt; *blogs* pessoais; *wiki*; visualização de páginas web dentro da plataforma; *feeds* RSS; fórum de discussão; gestão de tarefas; histórico; uso de ACLs; entre outros.

CWE (*Collaborative Work Environment*) é um conjunto de diversos *frameworks* para trabalho colaborativo do CCSDS (*Consultative Committee for Space Data Systems*) dentre os quais se destaca o SEA (*Systems Engineering Area*), onde diversas ferramentas são utilizadas para apoiar projetos, como calendário; anúncios; fóruns de discussões; *bookmarks*; lista de membros; gestão de documentos; *frameworks*; entre outros (CWE, 2012).

OCDS (*Open Concurrent Design Server*) é uma iniciativa da ESA que provê apoio à engenharia simultânea, colaborativa e distribuída para a indústria espacial, utilizando modelos de informação de padrões abertos e bibliotecas de referência. Este sistema teve anunciada sua migração de maneira transparente para o OCDT (*Open Concurrent Design Tool*), também desenvolvida pela ESA. O sistema atual dispõe de documentos; área para download e upload; fórum; informações para desenvolvedores; *reports* de *bugs*; links; manuais; área de instalação de clientes; área de informações técnicas para engenheiros; compartilhamento de informação e experiências; entre outros (OCDS, 2012).

4.2. Alguns casos de uso de e-Engineering

Por ser uma área em pleno desenvolvimento e que aplica conceitos relativamente novos, ainda há pouca divulgação de casos de uso *e-*

Engineering, principalmente utilizando abordagens de Computação Orientada a Serviços e Computação em Nuvem.

Entretanto, ainda que timidamente, à medida que esses conceitos ganham maturidade novos casos de sucesso começam a despontar no mercado, principalmente entre grandes organizações, normalmente tendo na vanguarda as que atuam no ramo computacional. A seguir são citados alguns casos de uso de *e-Engineering*.

4.2.1. Fujitsu

A Fujitsu anunciou em meados de 2011 o conceito de "*Engineering Cloud*", um ambiente de manufatura de última geração oferecido na forma de serviços baseados em nuvem a partir dos *datacenters* da empresa. Esse ambiente surgiu para apoiar o setor industrial com uma combinação do próprio software de apoio à engenharia da Fujitsu – CAD (*Computer-Aided Design*) e software analítico, bem como partes de software de banco de dados – com um conjunto de novos serviços para transformar o processo de fabricação (FUJITSU, 2011).

A Nuvem de Engenharia oferece aos clientes acesso a esses serviços em qualquer lugar e a qualquer momento, sem restrições de ambientes computacionais e sem necessidade de qualquer configuração especial prévia, reduzindo consideravelmente os custos de produção e tempos de desenvolvimento. Para os clientes que têm operações de desenvolvimento global e fabricação, a capacidade de compartilhar dados sem ser limitado pelo tempo ou lugar, ajudando a colaborar (YASUDA, 2012).

Com base na solução de PLM (*Product Lifecycle Management*) já provida aos seus clientes no setor de manufatura, a Fujitsu vê sua nova Nuvem de Engenharia como forma de contribuir para tornar o desenvolvimento mais criativo e espaços de produção que geram novo valor (FUJITSU, 2011).

Essa Nuvem de Engenharia compreende dois serviços: *Engineering Cloud/SaaS*, que oferece aos clientes no setor industrial com uma variedade de soluções como parte de um ambiente de desenvolvimento de produto; *Engineering Cloud/PaaS*, que proporciona um ambiente de alta velocidade a *thin clients* para suportar a próxima geração de fabricação através de aplicações personalizadas para as necessidades de cada empresa (YOSHIDA; FUJITA, 2012).

4.2.2. AMD

Em abril de 2009 a AMD iniciou um projeto para transformar seus pacotes isolados de servidores e armazenamento que ela tinha implantados em todo o mundo em uma nuvem privada. A estratégia global, internamente chamada de "*Compute Anywhere*", rapidamente tomou forma. O plano tático era centralizar os centros de dados, fornecer ferramentas de conectividade apropriadas, e assim estabelecer uma nuvem privada para abastecer todos os projetos de design da AMD em todo o mundo (AMD, 2011).

Em uma fase inicial, a TI da AMD substituiu seus processadores antigos em servidores de toda a organização pelos mais recentes e mais energeticamente eficientes. Como resultado, a AMD foi capaz de dobrar sua capacidade de computação, aproveitando sua infraestrutura existente e reduzindo custos (AMD, 2011).

Em paralelo, iniciou um projeto para padronizar software e versões de software ao longo de sua nuvem emergente. Após um estudo de padrões de uso e recursos de software, a TI da AMD também padronizou a plataforma de computação para gerenciamento de carga de trabalho e software para automação de design eletrônico (AMD, 2011).

Essa migração para uma infraestrutura de nuvem proporcionou os seguintes benefícios:

- Todos os projetos de engenharia da AMD residem numa nuvem privada;

- A equipe TI da AMD é capaz de responder dinamicamente às necessidades de projeto de equipes de engenharia, não mais sendo restringidos pela geografia e localização dos dados que necessitam, diminuindo diretamente *time-to-market* e reduzindo custos;

- Com a padronização e flexibilidade, a AMD pode utilizar melhor seu hardware e reduzir os custos do centro de dados, associados às instalações de hardware de apoio;

- Com flexibilidade de recursos de computação otimizada, a TI da AMD implantou a tecnologia *thin client*, fornecendo acesso de alta velocidade a partir de locais remotos para a nuvem;

- Com os dados já disponíveis em todos os lugares, cópias redundantes locais de conjuntos de dados não são mais necessárias. A AMD é capaz agora de alavancar engenheiros disponíveis, independentemente da sua localização, a participar em qualquer projeto, conforme necessário, utilizando conjuntos de dados consolidados e aumentando a produtividade;

- A nuvem permite à equipe de TI da AMD responder às necessidades dinâmicas, possibilitando recursos em todo o mundo, e mantendo altos níveis de prestação de serviços, além de permitir que qualquer projeto de engenharia seja facilmente movido através da nuvem (AMD, 2011).

4.3. Sistemas Legados

Normalmente, muitas tarefas durante o projeto conceitual são desenvolvidas usando algumas aplicações computacionais distribuídas através de uma organização.

A integração de sistemas legados e o crescimento constante do mercado para ambientes colaborativos tornam os serviços envolvidos e prestados por essas aplicações um tanto quanto atraentes, permitindo aos *stakeholders* (participantes) interagir e otimizar o gerenciamento de projetos. Assim, duas aplicações computacionais legadas são brevemente abordadas, SatBudgets e SpaceESB, uma vez que ambas constituem a solução proposta.

4.3.1. SatBudgets

SatBudgets é uma ferramenta para automação de cálculos de balanços de disciplinas da Engenharia de Sistemas de satélites como, por exemplo, mecânico, elétrico e computação de bordo, e baseia-se no modelo de satélite escrito SysML (*Systems Modeling Language*), uma linguagem gráfica de modelagem que apóia a análise, especificação, projeto, verificação e validação de sistemas complexos (FRIEDENTHAL et al., 2008). O diagrama de blocos SysML é, então, o diagrama principalmente explorado onde a arquitetura do satélite é descrita e os relacionamentos entre os blocos são definidos, como os pacotes dos subsistemas.

Esse software extrai informações paramétricas de projeto do modelo SysML que expressa a arquitetura do satélite que está sendo unificada a partir de contribuições dos especialistas das diversas disciplinas de engenharia de sistemas. Além disso, possui um mecanismo baseado em regras que impõe regras de negócios neste domínio que devem ser seguidas e que permite acionar, conforme necessário, o processamento de balanços (DOS-SANTOS et al., 2009).

Através de um sistema modelado em SysML, durante a fase de projeto conceitual, como entrada, o software gera balanços de diversas áreas para um dado projeto de satélite, onde constam os blocos definidos para um dado satélite e os relacionamentos entre eles, além da representação da utilização de pacotes, tais como os pacotes dos subsistemas existentes na arquitetura modelada (DOS-SANTOS et al., 2009).

O fluxo de trabalho da ferramenta desde a modelagem SysML do satélite, as etapas realizadas pela ferramenta SatBudgets até a fase final da ferramenta com a geração do relatório é mostrado na Figura 4.1 (LEONOR, 2010).

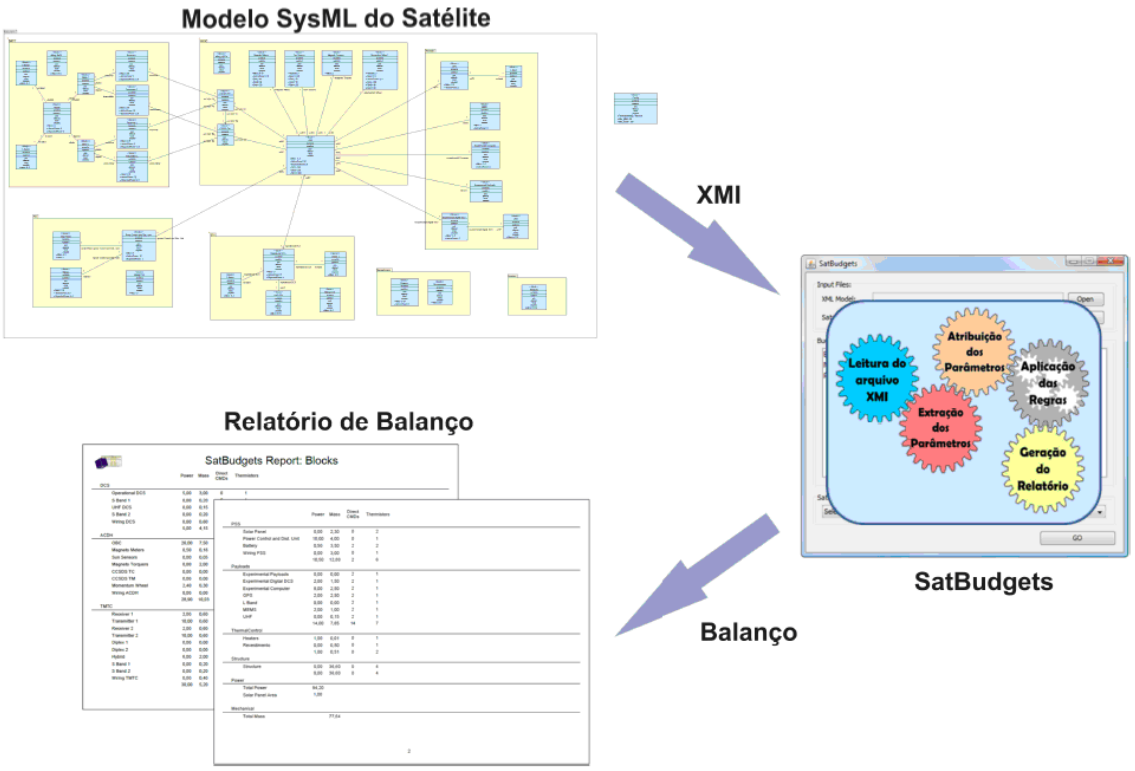


Figura 4.1 - Fluxo de trabalho do SatBudgets
Fonte: Leonor (2010)

Após entrar com o diagrama de blocos do satélite, o software gera saídas como relatórios e gráficos, possibilitando que exploração do espaço de projeto (DSE – *Design Space Exploration*) seja realizada a fim de se obter uma arquitetura de projeto viável com maior retorno (DOS-SANTOS et al., 2009).

4.3.2. SpaceESB

SpaceESB é uma extensão de um barramento de serviços corporativos (ESB – *Enterprise Service Bus*) para balanços que pode ser invocado remotamente utilizando o paradigma da arquitetura orientada a serviços (SOA – *Service-Oriented Architecture*) (SOUZA; DOS-SANTOS, 2011).

Esse ESB customizado provê acesso a alguns serviços para cálculo de balanços distribuídos e permite que alguns fluxos de trabalho de engenharia sejam suportados. Isso é particularmente interessante, pois atualmente se lida com alguns parceiros de projeto externos e essa solução pode trabalhar de forma transparente com questões de acoplamento do projeto. Assim, a abordagem facilita a integração, o compartilhamento de informações, a independência de plataforma e a interoperabilidade dos sistemas computacionais. Adicionalmente, permite que o impacto de quaisquer decisões mais importantes de projeto tomadas pela equipe de design seja rapidamente avaliado e incorporado, tendendo a reduzir os riscos concepções errôneas (SOUZA; DOS-SANTOS, 2011).

Nesse sistema a troca de dados é realizada através de mensagens SOAP (*Simple Object Access Protocol*), encapsuladas em XML (*eXtensible Markup Language*), permitindo que os sistemas-cliente possam consumir serviços através da comunicação com o SpaceESB, que por sua vez gera um pedido de serviço e retorna para o consumidor (SOUZA; DOS-SANTOS, 2011).

O SpaceESB possui uma interface junto ao SatBudgets, de forma que este consegue acessar os *web services* providos pelo SpaceESB, gerando um ambiente colaborativo independente de plataforma e implementando algumas das atividades pertencentes aos processos da fase de projeto conceitual (SOUZA; DOS-SANTOS, 2011). O fluxo de informações e a integração entre essas duas ferramentas são expostos na Figura 4.2.

Dessa forma, para cada modelo SysML de satélite fornecido, seus balanços podem ser selecionados na interface e processados localmente (LEONOR, 2010) ou remotamente via *web services* (SOUZA; DOS-SANTOS, 2011).

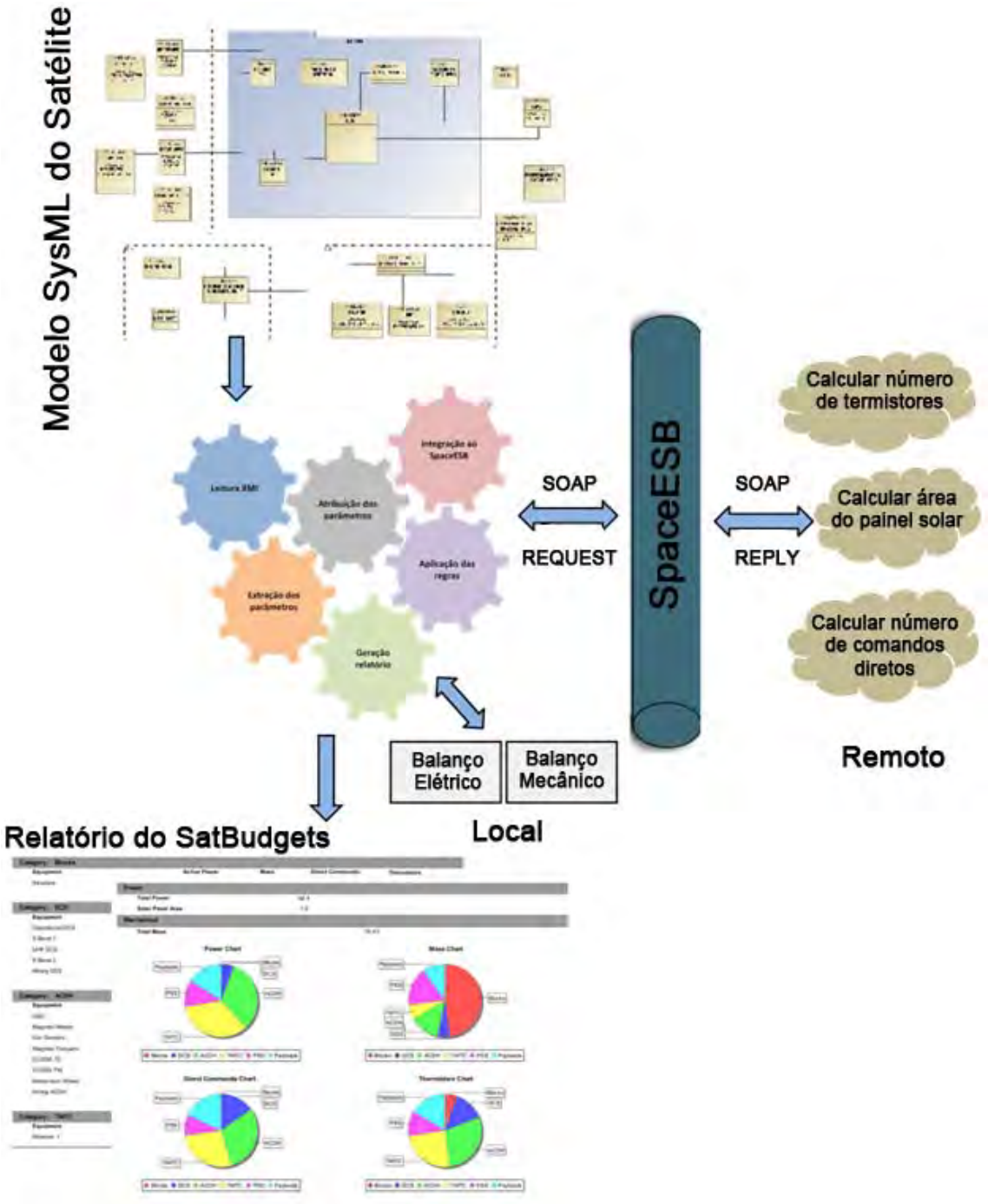


Figura 4.2 - Integração e execução entre SatBudgets e SpaceESB

Fonte: Souza (2011a)

5 CLOUDSATCD

Um ambiente colaborativo para o projeto conceitual de satélites parte da necessidade de automatizar os processos de colaboração, coordenação e comunicação entre a equipe de projeto da Engenharia e Tecnologia Espacial do INPE. Durante esta importante fase de projeto, os diversos profissionais envolvidos precisam trabalhar em conjunto, e faz-se necessária a utilização de ferramentas para auxiliar este processo.

Atualmente, no INPE, essa fase de projeto depende da utilização de ferramentas proprietárias para gerenciamento do ciclo de vida de produtos (PLM) e fluxo de trabalho (*workflow*), além do uso de planilhas para cálculos de balanços de disciplinas das diferentes áreas do conhecimento envolvidas em um projeto espacial.

Estas soluções requerem o pagamento de licenças para utilização e não contam com ferramentas como gerenciamento de projetos, e-mails, gestão de requisitos ou gestão do conhecimento, todas integradas, bem como não podem ser alteradas, devido às questões de direitos autorais, dificultando ou até mesmo impossibilitando a integração com soluções já existentes, por vezes fundamental, já que tais soluções não são especificamente voltadas para a área espacial.

De maneira análoga, a disponibilização de tais ferramentas a parceiros é praticamente inexistente, uma vez que depende que os envolvidos possuam a mesma solução proprietária para que haja interoperabilidade entre os participantes.

A solução proposta, denominada CloudSatCD, permite melhorar as capacidades de trabalho em grupo através de diversos serviços de software voltados para engenharia e disponibilizados para os usuários através de um ambiente virtual web, de forma que a equipe de projeto e seus parceiros possam ter acesso a uma variedade de ferramentas uniforme para apoiar os

processos e atividades do projeto conceitual de satélites, minimizando suas restrições inerentes.

Os requisitos para a formação deste ambiente podem ser expressos, de maneira geral, na Tabela 5.1.

Tabela 5.1 - Requisitos de CloudSatCD

ID	REQUISITOS
RF1	CloudSatCD deve prover ferramenta CSCW.
RF1.1	CloudSatCD deve prover ferramentas para colaboração.
RF1.2	CloudSatCD deve prover ferramentas para coordenação.
RF1.3	CloudSatCD deve prover ferramentas para comunicação.
RF2	CloudSatCD deve prover acesso às ferramentas através da web.
RF3	CloudSatCD deve prover serviços de softwares para equipe de projeto.
RF4	CloudSatCD deve prover acesso aos serviços de softwares a parceiros de projeto.
RF5	CloudSatCD deve prover a execução de softwares com maior flexibilidade.
RF6	CloudSatCD deve possibilitar a gestão de projetos.
RF7	CloudSatCD deve possibilitar a gestão do conhecimento.
RF8	CloudSatCD deve possibilitar a gestão do tempo e agendamento de eventos.
RF9	CloudSatCD deve possibilitar envio e recebimento de e-mails.
RF10	CloudSatCD deve possibilitar a produção de documentos colaborativos.
RF11	CloudSatCD deve possibilitar a gerenciamento de recursos em projetos.
RF12	CloudSatCD deve possibilitar a gestão de incidentes.
RF13	CloudSatCD deve possibilitar a gestão de conteúdo.
RF14	CloudSatCD deve possibilitar a gestão de tarefas.
RF15	CloudSatCD deve possibilitar a gestão de requisitos.
RF16	CloudSatCD deve possibilitar a gestão de *baselines*.
RF17	CloudSatCD deve possibilitar a rastreabilidade de falhas.
RF18	CloudSatCD deve possibilitar a avaliação de balanços de engenharia espacial.
RF19	CloudSatCD deve possibilitar agregar balanços disponibilizados por parceiros.

Um ambiente de *e-Engineering* é capaz de prover uma plataforma colaborativa para a interação necessária no desenvolvimento de projetos, promovendo o

trabalho em conjunto ao mesmo tempo em que permite automatizar processos do projeto conceitual de satélites em um ambiente distribuído, através de integração de serviços de engenharia e outros recursos. Quando disponibilizado em nuvem, estes serviços podem ser acessados por uma gama maior de usuários, sejam membros da equipe ou parceiros, além de ser expansíveis.

A arquitetura proposta provê funções e serviços de *groupware* e de requisitos, formando uma infraestrutura de computação em nuvem, através de um ecossistema *mashup*[2] de *e-Engineering*, conforme ilustrado na Figura 5.1.

Figura 5.1 - Ecossistema *mashup* para a infraestrutura de *e-Engineering*

Nesse sentido, o paradigma da Computação Orientada a Serviços aplicado à infraestrutura de *e-Engineering* pode ser estendido eficientemente à

[2] Aplicação web que usa conteúdo de mais de uma fonte para criar um novo serviço e/ou aplicação.

Engenharia Concorrente de Sistemas Espaciais na fase de projeto conceitual e pode contribuir na agilidade de execução de processos internos da área de Engenharia e Tecnologia Espacial do INPE.

A solução, descrita na Figura 5.2, é baseada em computação em nuvem (*cloud computing*) privada/local, trabalhando num regime de software como um serviço (SaaS), e utiliza uma abordagem baseada em serviços para oferecer uma combinação de computação de infraestrutura, gerenciamento, armazenamento e serviços de software, sendo que esta infraestrutura é transparente ao usuário (HURWITZ et al., 2009).

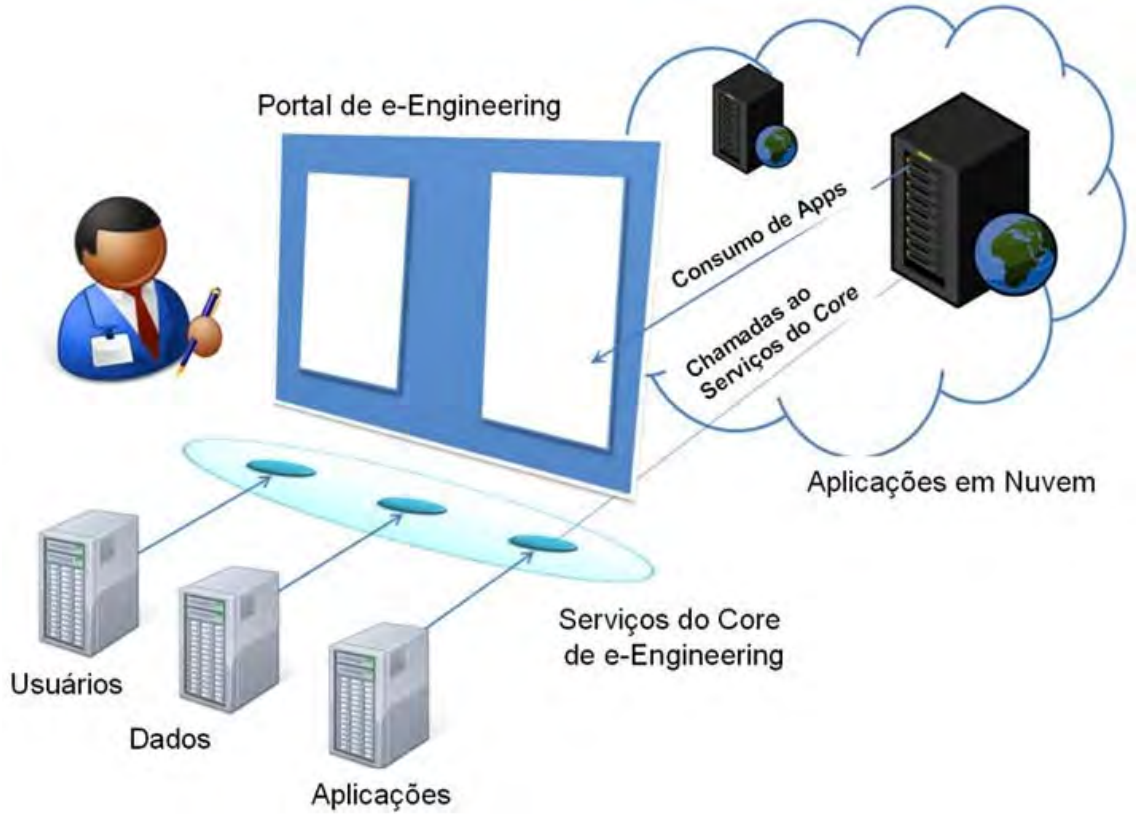


Figura 5.2 - Uma extensão de *e-Engineering* para infraestrutura padrão de *e-Science*

Como representado a seguir na Figura 5.3 em alto nível de abstração e em camadas, o ambiente colaborativo proposto pode ser representado da seguinte forma.

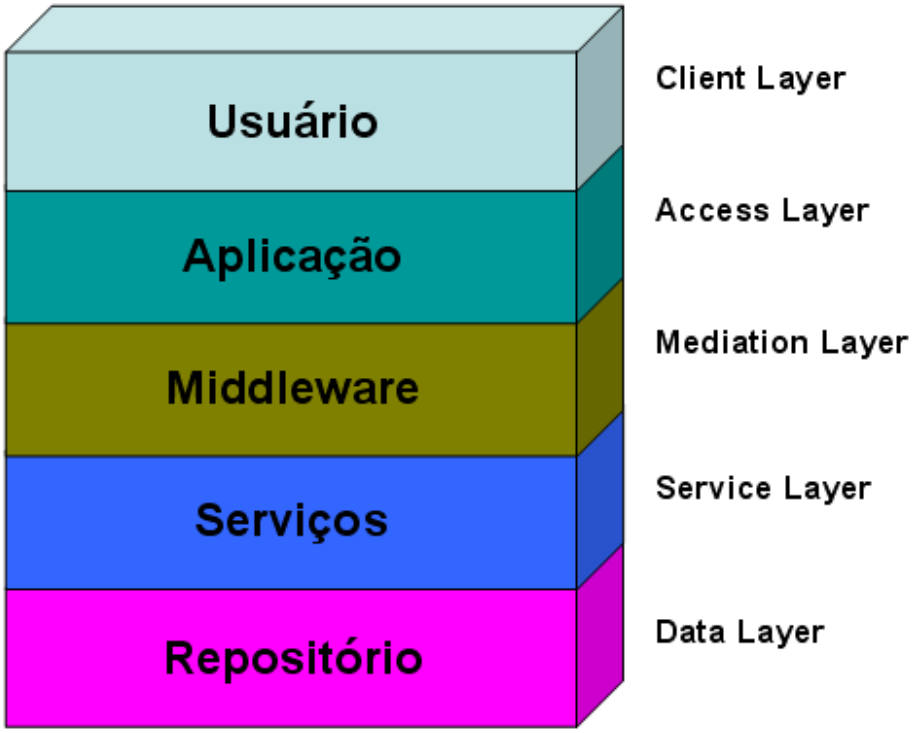


Figura 5.3 - Camadas da arquitetura proposta de *e-Engineering*

Client Layer

É a camada onde estão os usuários, ou seja, os consumidores dos serviços providos pela infraestrutura de *e-Engineering*.

Access Layer

É a camada que implementa uma interface para interação entre os usuários e a arquitetura subjacente, onde estão os recursos providos por ela. É responsável por: autenticação de usuários; monitoração de trabalhos; submissão de dados; visualização de dados; colaboração entre usuários; acesso aos serviços de *groupware*; serviços de requisitos e testes; interface para o SatBudgets como serviço. Basicamente, é composta pelo portal que dá acesso às opções de serviços, conforme ilustrado na Figura 5.4.

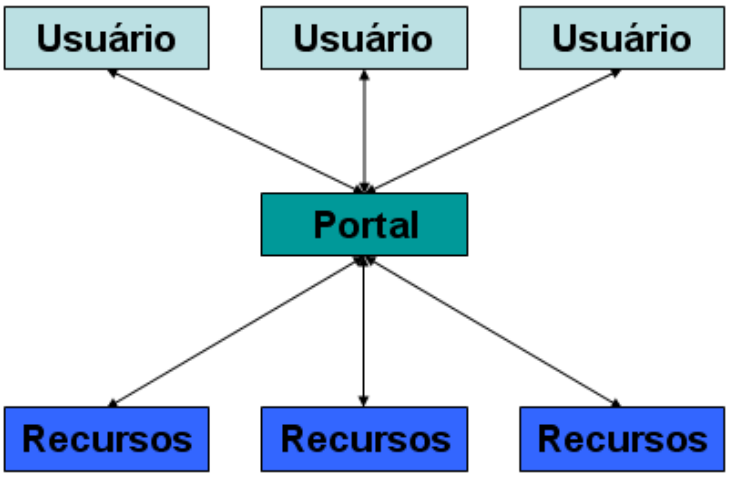


Figura 5.4 - Portal de *e-Engineering* de CloudSatCD

Mediation Layer

É a camada de mediação, sendo responsável por intermediar os serviços providos e a camada de acesso, além de realizar: interação entre aplicação e SatBudgets; abstrair a presença do SpaceESB da aplicação; armazenamento, busca, descoberta e registro de serviços web; codificação dos parâmetros e execução do serviço web. Esta camada atua como um *middleware*, e que também pode ser usada para a criação de infraestruturas e aplicações distribuídas, permitindo o compartilhamento de serviços, base de dados e outras ferramentas de maneira segura.

Service Layer

Esta camada é composta pelos serviços providos pela *e-infraestrutura*[3], onde estão os serviços de *Groupware*, Requisitos, SatBudgets e SpaceESB, que recebem os dados e executam os serviços solicitados, vinculados aos respectivos interessados, para posteriormente enviar uma resposta à solicitação feita.

[3] Termo utilizado para definir tecnologia/ferramenta estratégica para fomentar a inovação colaborativa global a pesquisadores. Abrange redes, grades, centros de dados e ambientes colaborativos, entre outros, pois é a integração destes que define uma *e-infraestrutura*.

Data Layer

Esta camada é um repositório que contém os dados dos usuários e de seus serviços, além de outros dados e documentos relacionados ao projeto conceitual de satélites, conforme ilustrado na Figura 5.5.

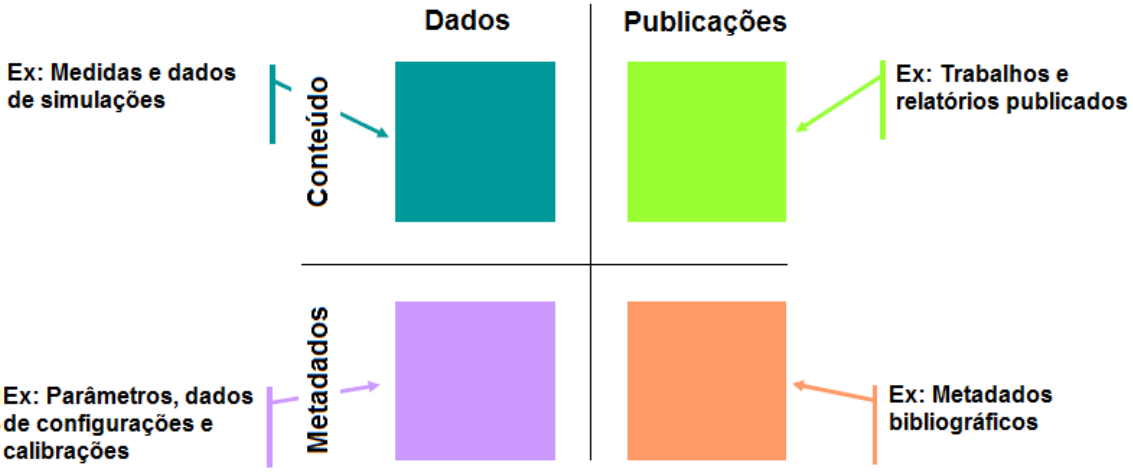


Figura 5.5 - Tipos de dados no repositório
Fonte: Adaptado de eSciDR (2008)

Através da criação da arquitetura descrita é possível atender aos requisitos mapeados, disponibilizando em nuvem um ambiente computacional com diversos serviços de software para apoiar o projeto conceitual de satélites, através de uma solução de *groupware* e um gerenciador de requisitos, agregando de forma transparente os sistemas legados.

Inicialmente, a arquitetura proposta trabalhará num ambiente web ainda voltado para a rede interna (Intranet) e/ou aberta à colaboração de parceiros (Extranet), formando uma infraestrutura de computação em nuvem local que provê software como serviço (SaaS). Tal restrição se dá até por questões de segurança e confidencialidade das informações dos projetos envolvidos, mas a estrutura deve ser capaz de se comunicar com outras redes, fazendo a descoberta de serviços por elas disponibilizados.

A solução em nuvem proposta, ilustrada na Figura 5.6 e denominada de CloudSatCD, é baseada num esquema de SaaS (*Software as a Service*) e agrega os aplicativos SatBudgets e SpaceESB, para geração de balanços, local ou remoto, respectivamente, bem como um *groupware* e um gerenciador de requisitos. Esta estrutura de *e-Engineering* é, portanto, um *mashup* proveniente da junção do SatBudgets, SpaceESB, *Groupware* e Gestor de Requisitos, todos disponibilizados em nuvem via navegador web.

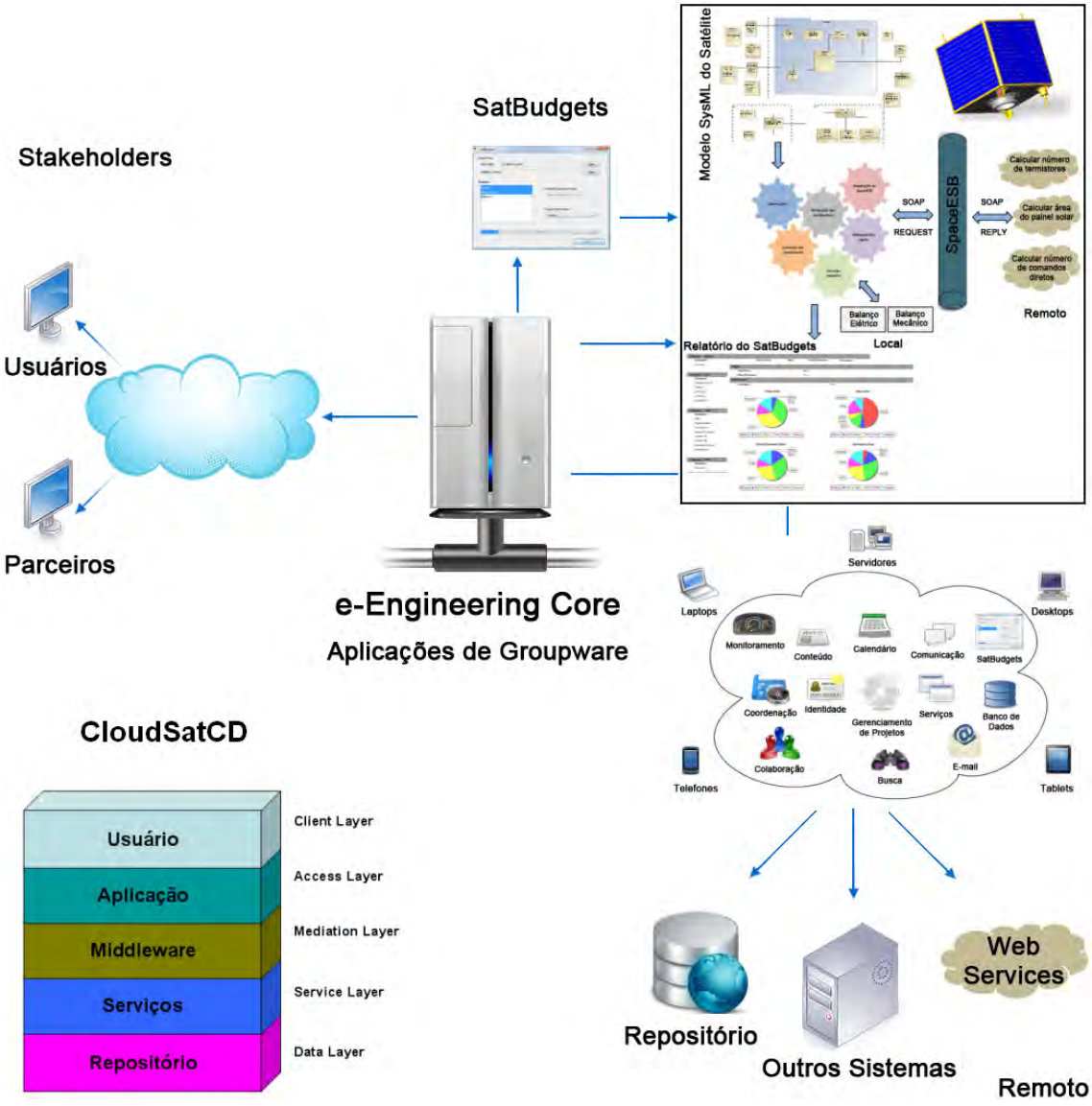


Figura 5.6 - Integração do SatBudgets e SpaceESB à nuvem de *e-Engineering*

Tal solução permite uma variedade de clientes que vão desde *desktops* até dispositivos móveis, oferecendo uma combinação de infraestrutura de computação, gestão, armazenamento e serviços de software transparentes para o usuário através do portal *mashup*. Por ter capacidade de *groupware*, os membros da equipe de projeto podem também ter acesso a uma variedade de aplicações que residem na nuvem.

Essa arquitetura, assim, vai de encontro à proposta de *e-Science*, ou seja, a convergência de TI e ambiente de pesquisas do INPE, priorizando a fase mais importante de um projeto espacial: o projeto conceitual de satélites.

A nuvem computacional proposta CloudSatCD pode ser dividida em duas partes: (1) **ambiente administrativo**, base da infraestrutura da nuvem privada, formado somente por máquinas físicas que executam as máquinas virtuais, provedor de infraestrutura como serviços (IaaS), cujas operações e uso estão limitados ao administrador da infraestrutura, que controla todos os recursos da nuvem; e (2) **ambiente operacional**, base da infraestrutura de *e-Engineering*, formado somente por máquinas virtuais criadas na nuvem, provedor de software como serviço (SaaS), onde estão as aplicações providas cujo objetivo é disponibilizar software para o usuário final, onde este faz uso dos recursos providos.

As principais ferramentas utilizadas para a prototipação da infraestrutura são listadas e descritas no Apêndice A, enquanto a materialização dos ambientes administrativo e operacional, por se tratar de conteúdo de maior complexidade técnica para a criação da nuvem privada, instalação e configuração, está descrita detalhadamente nos Apêndices B e C, respectivamente.

O ambiente operacional, núcleo da infraestrutura proposta, pode ser dividido em dois grupos, alocados separadamente nas duas máquinas virtuais de CloudSatCD: (1) **Concentrador de Serviços**, na máquina virtual 1, núcleo da solução, onde estão os serviços providos pelas aplicações de *groupware*, gerenciador de requisitos e SatBudgets, este último via interface e interligado

ao SpaceESB; e (2) **Gerenciador de VPN**, na máquina virtual 2, responsável pelo gerenciamento de VPN (*Virtual Private Network*) entre a arquitetura e sistemas remotos, provendo acesso seguro e criptografado, além de disponibilizar redirecionamentos de chamadas HTTP via módulo proxy entre o Concentrador de Serviços e os sistemas ou usuários remotos.

O Concetrador de Serviços está subdividido em dois grupos, categorizando os serviços de software providos e formando o ecossistema *mashup* através de: (1) ***CloudSatCD Groupware Tool***, onde estão os serviços de *groupware*; e (2) ***CloudSatCD Requirements Tool***, onde estão os serviços de requisitos e o SatBudgets, que adicionalmente se comunica com o SpaceESB. A Figura 5.7 ilustra abstratamente a arquitetura, tal como a disposição dos ambientes e ferramentas de CloudSatCD.

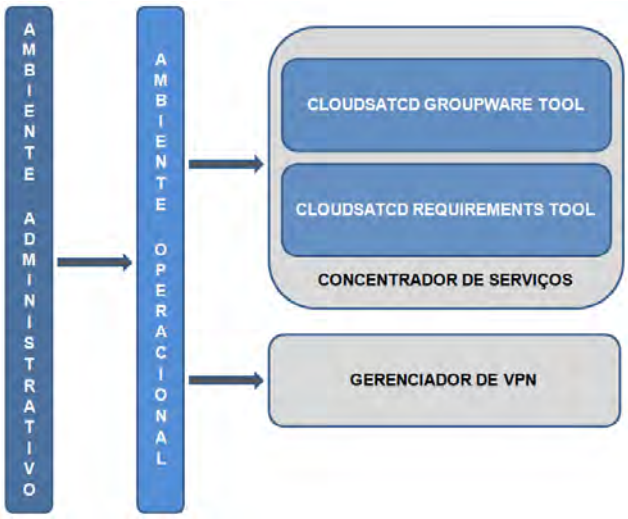


Figura 5.7 - Arquitetura de CloudSatCD

CloudSatCD Groupware Tool disponibiliza diversos serviços de software de *groupware* como gerenciador administrativo (*Admin*), webmail (*Email*), agenda corporativa (*Calendar*), catálogo de endereços (*Address Book*), gestão de arquivos (*File Manager*), gestão de tarefas (*Infolog*), gestão de projetos (*Project Manager*), gestão do tempo de trabalho (*Time Sheet*), gestão de incidentes (*Tracking System*), marcadores (*Bookmarks*), gestão de conhecimento

(*Knowledge Base*), gestão de recursos (*Resources*), gestão de sites (*Site Manager*), gestão de documentos colaborativos (*Wiki*), notícias corporativas (*News Admin*), enquetes (*Polls*), e outros voltados para a adminstração.

CloudSatCD Requirements Tool disponibiliza serviços de software para trabalhar com requisitos, como gestão de requisitos (*Requirements*), gestão de testes (*Test Library*), gestão de *baseline* (*Release*), resultados de testes (*Test Results*), gestão de falhas (*Defects*), relatórios (*Reporting*), gestão de projetos de requisitos (Manage), gestão de usuários (*User*) e SatBudgets (invocação de balanços).

A Figura 5.8 demonstra os serviços de software providos pela arquitetura ao expandir o ecossistema *mashup*, bem como suas disposições no grupos de serviços e inclusão dos serviços do SatBudgets em sua versão web, cuja integração é abordada na seção 5.1.

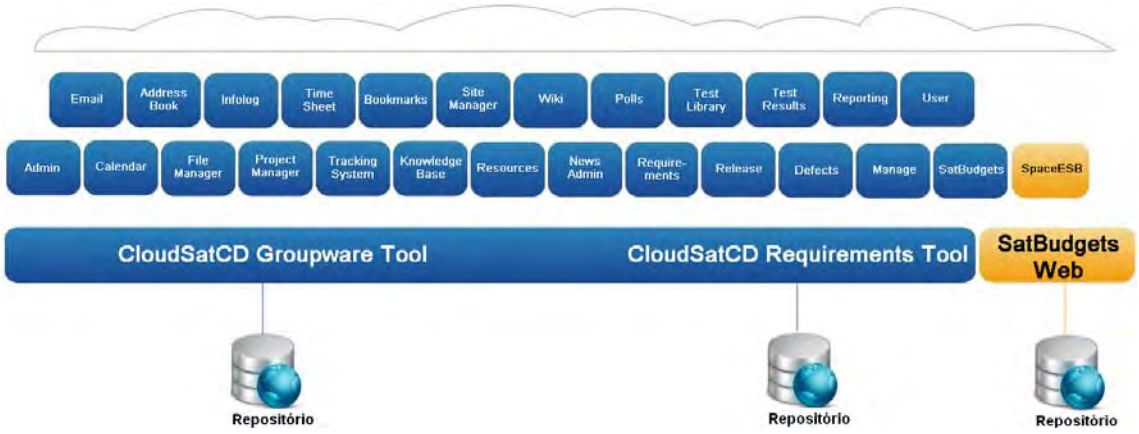


Figura 5.8 - Serviços de *e-Engineering* de CloudSatCD

5.1. Integração CloudSatCD-SatBudgets-SpaceESB

A integração dos sistemas legados, SatBudgets e SpaceESB, parte da necessidade de acesso remoto aos serviços providos por eles sem a necessidade de instalação do software localmente (*stand-alone*) e da possibilidade de utilização de um repositório de dados único para manter um

histórico de modelos de satélites anteriormente analisados, permitindo que tais modelos possam ser acessados posteriormente, além de garantir a integridade e consistência dos dados armazenados.

Adicionalmente foram acrescentados à versão original do SatBudgets cálculos relativos a algumas áreas-chave do conhecimento de gerenciamento de projetos baseados no PMBOK, como Recursos Humanos, Aquisição, Comunicação e Qualidade. Além destes novos balanços, a versão estendida do software permite que duas arquiteturas (dois arquivos XMI) possam ser analisadas e comparadas através dos relatórios e diagramas *Kiviats*[4], maximizando as possibilidades para a escolha entre duas arquiteturas candidatas (MEDEIROS et al., 2012).

No desenvolvimento da versão web foram utilizados: (1) o *framework* Grails, com a finalidade de aumento de produtividade na implementação; (2) a linguagem Groovy, em função da portabilidade Java; (3) o banco de dados *open source* PostgreSQL; e (4) o servidor de aplicações Apache Tomcat para o ambiente de produção.

A abordagem utilizada transforma os balanços existentes em serviços de rede RESTful, onde há a provisão da URL (*Uniform Resource Locator*) de acesso ao serviço, que por sua vez retorna um arquivo XML em resposta a uma requisição com o método HTTP GET.

A exposição dos serviços de balanços prestados por SatBudgets e SpaceESB é então feita utilizando SOA, através de *web services* REST entre CloudSatCD e SatBudgets, e através de *web services* baseados em SOAP e WSDL entre SatBudgets e SpaceESB, já existente (SOUZA, 2011a). Dessa forma os serviços são invocados por CloudSatCD e executados remotamente através da rede, por SatBudgets e/ou SpaceESB, dependendo dos balanços

[4] Gráfico do tipo radar que consiste de uma sequência de raios angulares com cada raio representando uma das variáveis. É desenhada uma linha ligando os valores de dados para cada um dos raios, dando a aparência de estrelas, onde cada estrela representa uma única observação multivariada com um número arbitrário de variáveis.

selecionados, para posteriormente devolver a resposta da requisição via arquivo XML. A Figura 5.9 ilustra abstratamente a integração feita.

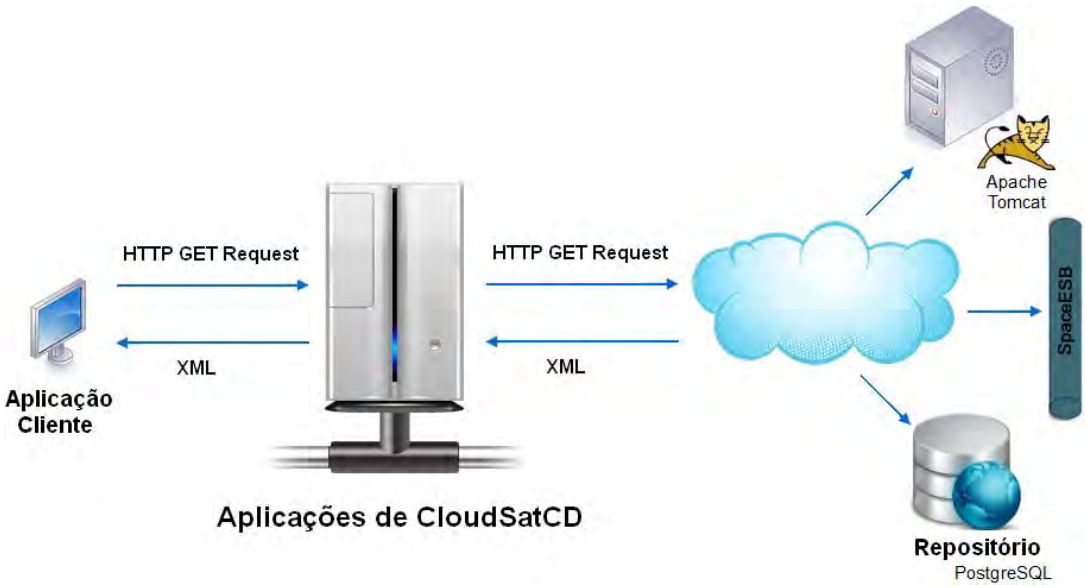


Figura 5.9 - Integração CloudSatCD-SatBudgets-SpaceESB

A invocação do serviço tem início do servidor web através da ferramenta de gerenciamento de requisitos, na máquina virtual 1. Um botão/link na ferramenta faz a ligação entre o servidor web e o servidor de aplicações exibindo a interface para acesso ao SatBudgets. Depois de configuradas e selecionadas as opções disponíveis, é feita uma chamada ao Apache da máquina virtual 2, onde há uma entrada para redirecionamentos HTTP baseados em *Proxy* de requisições internas, provenientes da máquina virtual 1, para a estrutura com o servidor que executa a versão web do SatBudgets, através de uma ligação via VPN.

Para acesso remoto aos balanços acontece basicamente o processo inverso ao que ocorre na ligação entre usuários externos e o sistema. Os redirecionamentos são totalmente transparentes ao usuário, que não tem conhecimento do acesso remoto, pois a VPN é estabelecida diretamente entre os servidores, fazendo com que a chamada seja vista como local. A Figura

5.10 demonstra as ligações existentes entre os servidores para o acesso aos serviços do SatBudgets.

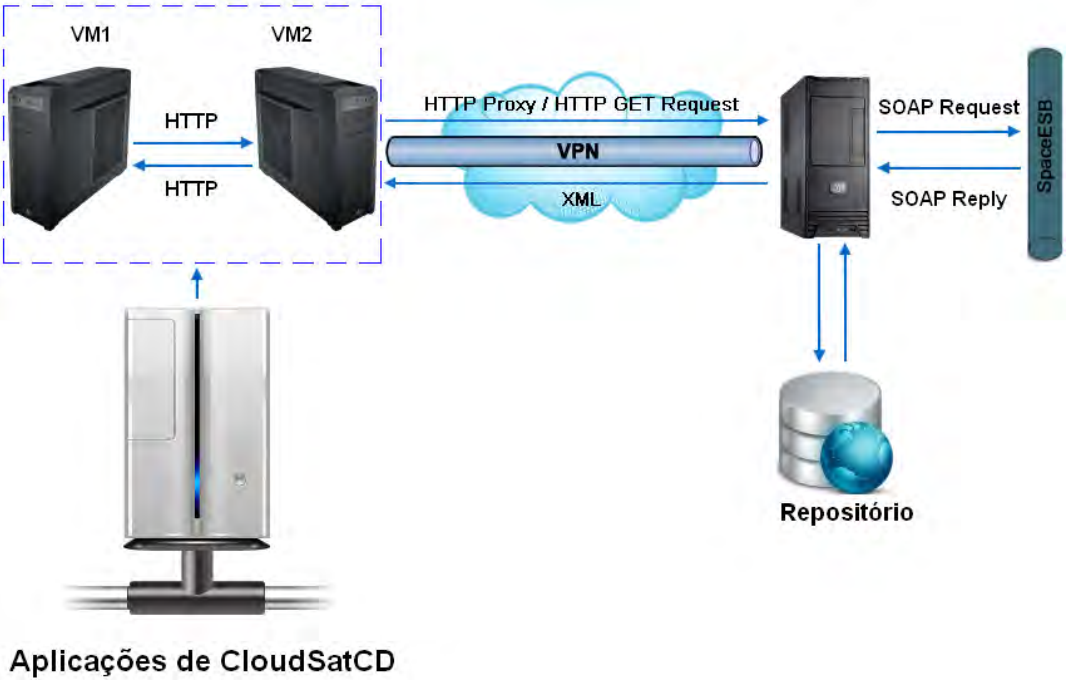


Figura 5.10 - Ligações CloudSatCD-SatBudgets-SpaceESB

CloudSatCD possui uma interface web, descrita na seção 5.2.2.1, que consome os *web services* REST disponibilizados invocando processamento remoto de serviços que realizam cálculos de balanços de arquiteturas, sejam cadastradas no banco de dados remoto ou inseridas em tempo de execução via interface.

Com o sistema em funcionamento, o ambiente operacional de CloudSatCD, responsável por prover software como serviços para os usuários, está concluído.

5.2. Serviços de Software disponibilizados

O sistema disponibiliza uma variedade de aplicações para colaboração e gerenciamento de projetos, onde qualquer usuário devidamente autorizado

com pouco ou nenhum conhecimento sobre os serviços é capaz de utilizá-los. As principais aplicações e *features* da solução são descritas a seguir, divididas entre os grupos *CloudSatCD Groupware Tool*, para serviços de *groupware*, e *CloudSatCD Requirements Tool*, para serviços de requisitos.

5.2.1. CloudSatCD Groupware Tool

Admin é o módulo de configuração administrativa da solução, somente exibido para usuários inseridos no grupo com privilégios administrativos. Esta aplicação possibilita automatizar diversos procedimentos relacionados à troca de informação e documentos, distribuição e coordenação de tarefas, de forma centralizada, de acordo com a necessidade de cada participante e coletivamente, auxiliando na gestão do fluxo de trabalho e proporcionando horizontalização do ambiente de engenharia de sistemas.

Um conjunto de regras pré-definidas para controle de acesso pode ser aplicado aos participantes quanto aos serviços disponíveis, Tal característica também permite implementar mecanismos de segurança baseados na necessidade do conhecimento, de forma que a informação seja acessada apenas dentro do contexto ao qual e para quem ela é necessária.

De maneira análoga, esse gerenciamento da equipe proporciona-se um diretório central que facilita a indexação de todos os participantes com suas respectivas informações pessoais e profissionais, sendo o sistema capaz de distinguir variados tipos e níveis de usuários baseado em controle de privilégios e acesso.

A Figura 5.11 exibe a página inicial do módulo ao centro, enquanto um menu na lateral esquerda permite o acesso a diversas outras páginas da interface administrativa, como gerenciamento de usuários, gerenciamento de grupos, configurações do sistema, aplicações, categorias, entre outros.

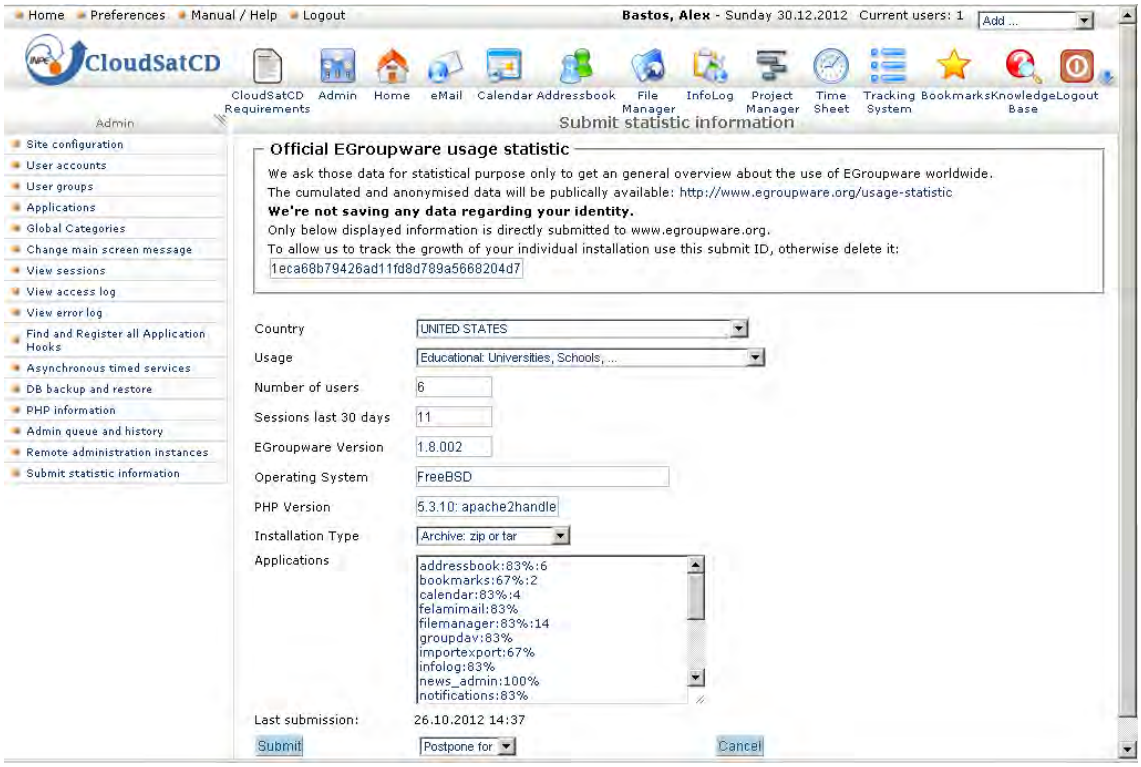


Figura 5.11 - *CloudSatCD Groupware Tool - Admin*

Entre as principais funcionalidades do módulo *Admin* estão (EGROUPWARE, 2011): gerenciamento de usuários; gerenciamento de grupos; gerenciamento de configurações do sistema; gerenciamento da segurança global do sistema; permissão de aplicações para usuários e grupos; gerenciamento de direitos de acesso para usuários e grupos no nível de aplicação; gerenciamento de notificações; individualização de páginas iniciais e de *login*; customização de logotipos e ícones; definição de conexão HTTPS; gerenciamento de ACL; gerenciamento de preferências forçadas, padrão e personalizáveis no nível de aplicação; configurações gerais das aplicações; visão de sessões e acessos a *logs*; restauração e *backup* do banco de dados; sincronização de dados com clientes *desktop*; sincronização de dados com dispositivos móveis através de SyncML e protocolos CalDAV, CardDAV e GroupDAV; acesso WebDAV ao gerenciador de arquivos; visualização de sessões; visualização do *logs* de acesso; visualização dos *logs* de erros; registro de aplicações; execução de

serviços assíncronos; histórico administrativo; configuração de instâncias para administração remota; entre outros.

Email é uma aplicação de correio eletrônico baseada em cliente IMAP completamente integrada e conectada aos outros serviços que a solução oferece, tornando possível o recebimento e envio de e-mails, seja através de servidor local pré-configurado ou com perfis adicionados para servidores remotos, em ambos os casos podendo utilizar autenticação se necessária.

É possível definir quais aplicações, usuários e grupos têm acesso ao serviço, ajudando a distribuir informações e, por conseguinte, a gerir as comunicações, estabelecendo um canal direto entre os participantes e evitando que haja lacunas entre a troca de mensagens de maneira efetiva e sempre necessária, dando acesso a informações de forma consolidada. A Figura 5.12 ilustra a interface de e-mail ativa ao centro, enquanto o menu na lateral esquerda permite o acesso a botões de ações, às pastas do usuário no servidor e às configurações gerais do serviço.

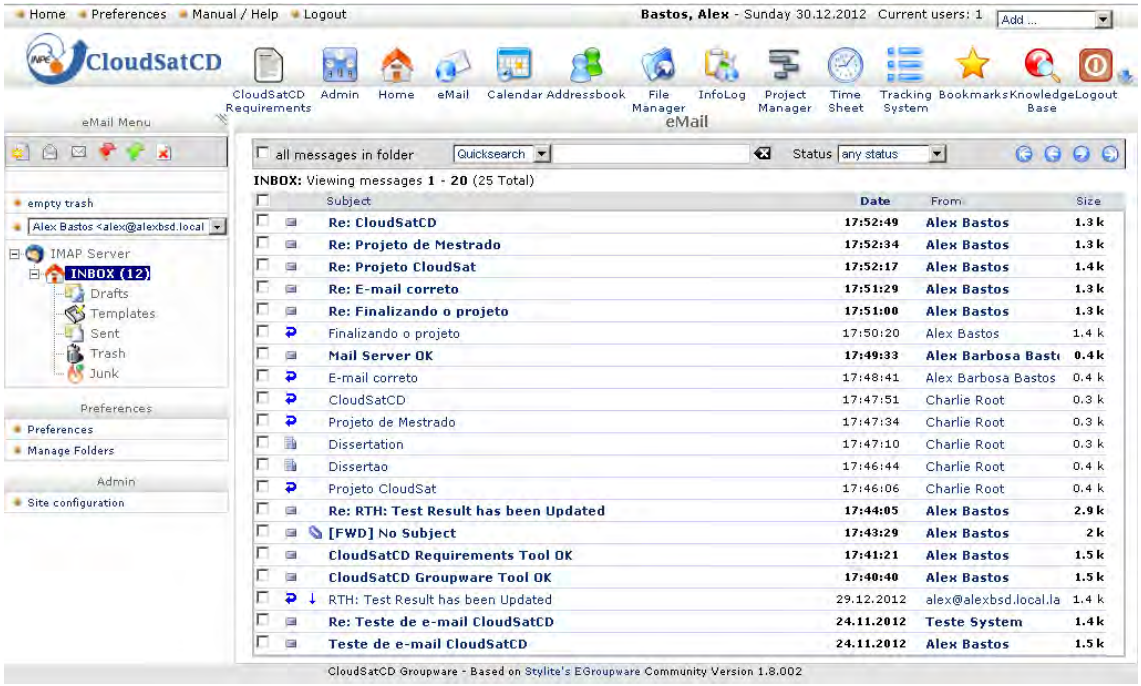


Figura 5.12 - *CloudSatCD Groupware Tool - Email*

Adicionalmente, o módulo *Email* possibilita que o usuário configure suas próprias conta e identidade de e-mail, caso essa funcionalidade tenha sido habilitada pelo administrador.

É possível destacar dentre as principais funcionalidades do serviço de *Email* (EGROUPWARE, 2011): visualização de múltiplas identidades e contas de e-mail IMAP; seleção de contatos a partir de catálogos de endereços e listas de distribuição; composição de mensagem em formato HTML ou texto simples (*plain text*); filtragem e ordenação de e-mails por diversos parâmetros (lidos, não lidos, marcados, respondidos, encaminhados, entre outros); múltiplos e-mails por vez; salvar anexos diretamente no gerenciador de arquivos; funções de impressão e busca; possibilidade de autenticação STMP e IMAP (SSL); uso de rascunhos e *templates*; adicionar e salvar facilmente contatos de e-mail no catálogo de endereços; entre outros.

Calendar é um eficiente e poderoso calendário/organizador corporativo que possibilita aos usuários organizar eventos, compartilhar agendas e delegar de tarefas a outros participantes, além de agendar reuniões em grupo automaticamente com base no tempo disponível no calendário de outros usuários.

Ao administrar o tempo consegue-se agendar tarefas, minimizar o tempo gasto em atividades, marcar reuniões, centralizar as informações de forma a organizar e maximizar o planejamento de tarefas, delimitar e estimar marcos e metas para o empreendimento.

A Figura 5.13 ilustra a interface do calendário em modo de visualização *default* ao centro, enquanto o menu na lateral esquerda permite o acesso às opções de visualização, planejamento em grupo, adição de eventos e outras preferências e configurações gerais do serviço.

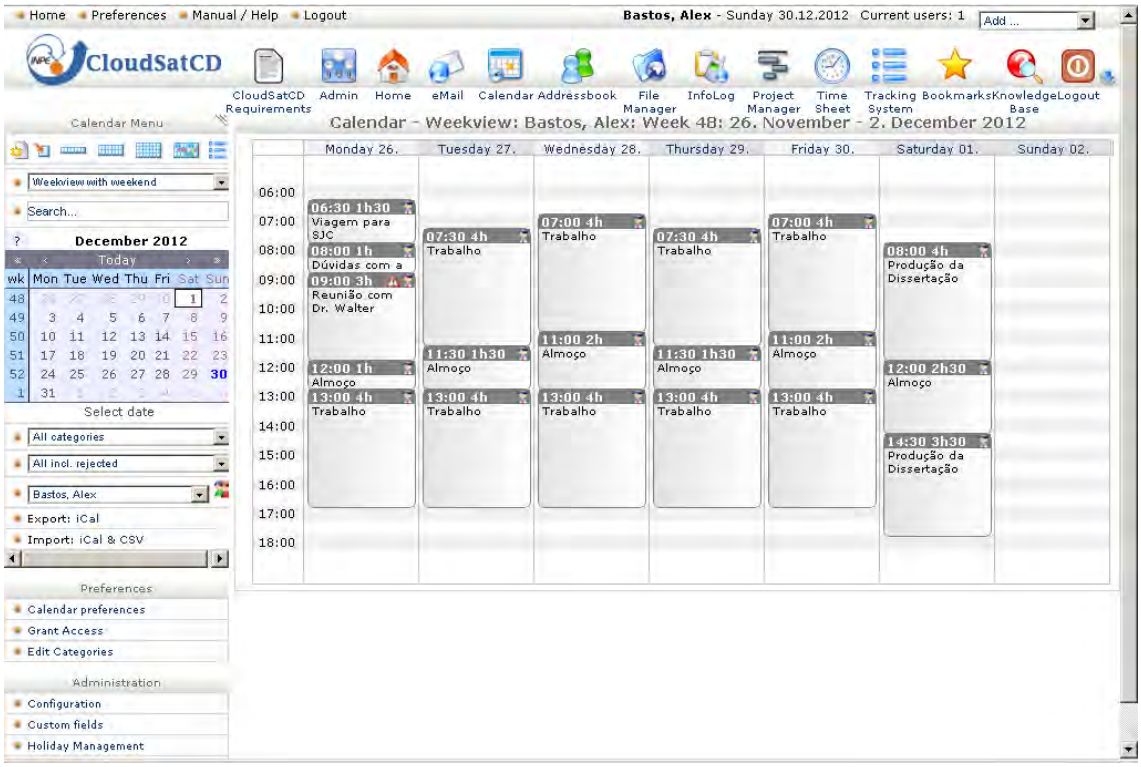


Figura 5.13 - *CloudSatCD Groupware Tool - Calendar*

O serviço *Calendar* possui uma série de funcionalidades, tais como (EGROUPWARE, 2011): programação e agendamento de tarefas, grupos, recursos e contatos; inserção de alarmes e lembretes; gerenciamento de conflitos e avisos; busca por tempo livre; customização de eventos; seleção de visões do calendário e planejamentos; envio, aceitação e rejeição de convites; uso de ACL; visualização dos eventos aceitos pelos participantes; criação de múltiplas categorias com cores em nível de grupo; lista de recursos de para eventos; notificação automática por e-mail; sincronização de dados com clientes; entre outros.

Address Book é um aplicativo para gerenciamento de informações de contatos que pode ser vinculado a outras aplicações do *groupware*, além de importar e exportar contatos em formatos variados. A Figura 5.14 exibe um exemplo de lista de contatos ao centro, enquanto o menu na lateral esquerda permite o acesso às opções gerais do serviço.

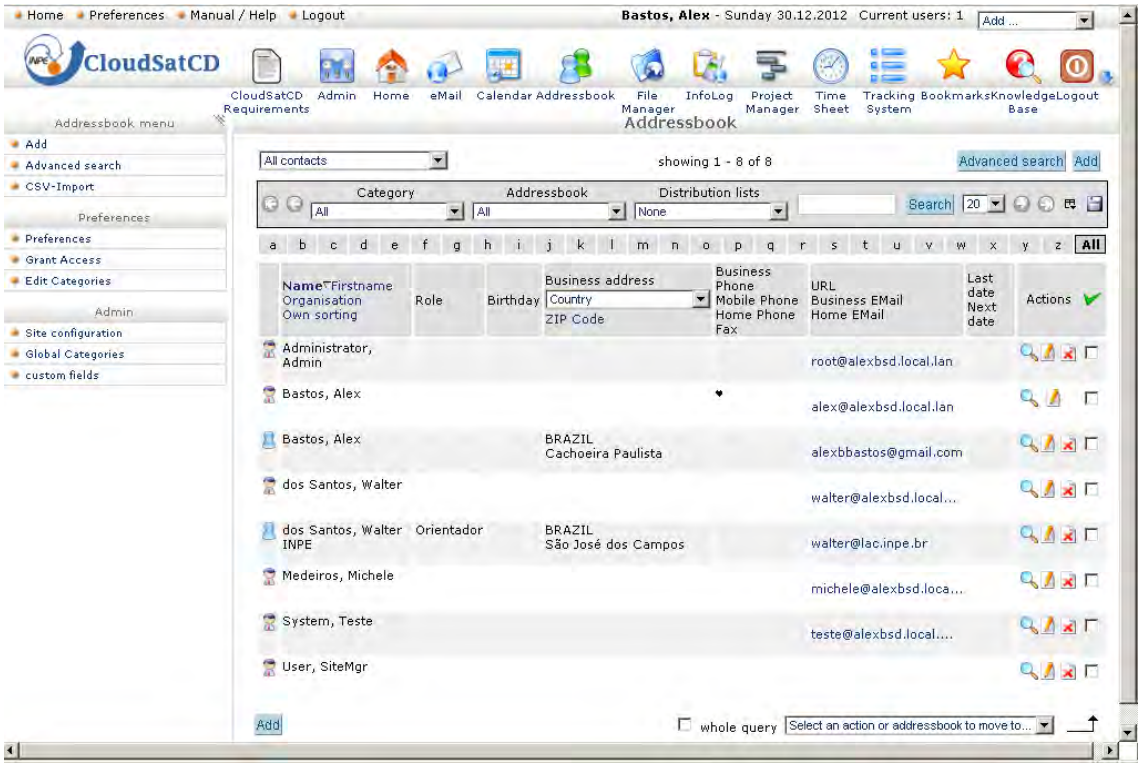


Figura 5.14 - *CloudSatCD Groupware Tool - Address Book*

Entre as principais características do serviço *Address Book* destacam-se (EGROUPWARE, 2011): múltiplos catálogos de endereços e tipos de contatos; organização dos contatos com lista de distribuição; seleção, ordenação e filtragem de contatos por parâmetros variados; busca avançada; link entre contatos e outros dados como tarefas ou projetos; anexação de arquivos para um contato; uso de ACL; gerenciamento de múltiplos contatos por vez; composição de e-mail a partir dos contatos; compartilhamento e proteção de catálogos de contatos; integração com telefone; sincronização de dados com clientes; entre outros.

File Manager é um módulo gerenciador de conteúdo que permite aos usuários armazenar, compartilhar, e administrar arquivos e documentos no sistema que pode ser utilizado como um repositório centralizado, constituindo assim um componente essencial num ambiente de trabalho colaborativo.

Com a gestão de documentos é possível criar, armazenar, atualizar e acessar documentos virtualmente, gerando um concentrador de informação documental relevante aos projetos de engenharia, permitindo compartilhamento e implementação de processos e/ou atividades associadas.

A aplicação utiliza VFS (*Virtual File System*) para armazenar e possibilita que os arquivos sejam compartilhados com outros usuários através de ACLs. A Figura 5.15 ilustra a página inicial do serviço ao centro, contendo os arquivos disponibilizados com suas respectivas permissões e opções de gerenciamento, enquanto o menu na lateral esquerda possibilita o acesso às opções e configurações gerais.

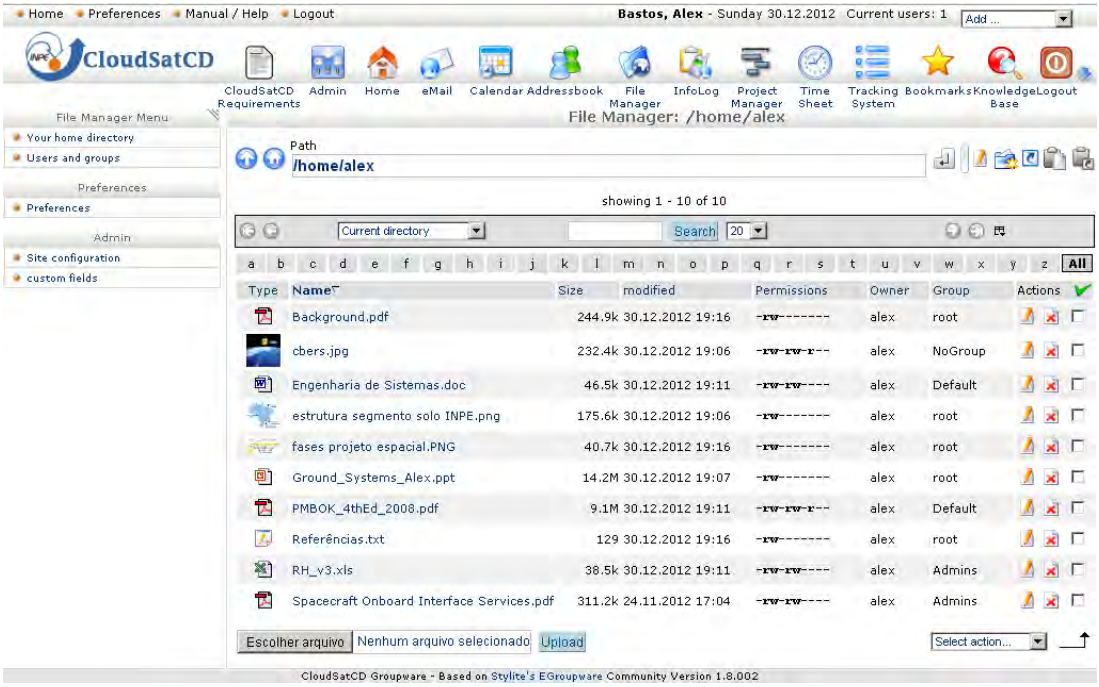


Figura 5.15 - *CloudSatCD Groupware Tool - File Manager*

Dentre as opções disponibilizadas pelo *File Manager* é possível salientar (EGROUPWARE, 2011): área de dados de usuário individual; funções comuns como *upload*, renomear, copiar, mover, criar diretórios, excluir, adicionar e editar arquivos de texto, navegação pelo diretório, visualização de atributos; área de trabalho personalizável; permissões para arquivos, diretórios, usuários

e grupos; diretórios compartilhados para cada grupo; busca e critérios de seleção; criação de diretórios e subdiretórios; coletar arquivos a partir de múltiplos diretórios para copiar e colar; adicionar comentários e campos extras; *upload* de múltiplos arquivos a partir do *browser* ou conexão WebDAV[5] (*Web-based Distributed Authoring and Versioning*); uso de ACL; acesso WebDAV pelo cliente; entre outros.

Infolog (Task Management) é uma ferramenta para gerenciamento de tarefas e atividades que atua como uma aplicação CRM (*Customer Relationship Management*) capaz de utilizar informações do catálogo de endereços combinadas com lista de tarefas, notas e chamadas telefônicas, permitindo maior fluxo de informação ao mesmo tempo em que aumenta a potencialidade do sistema. A Figura 5.16 ilustra um exemplo de página inicial com tarefas ao centro, enquanto o menu na lateral esquerda permite o acesso às opções e configurações gerais do serviço.

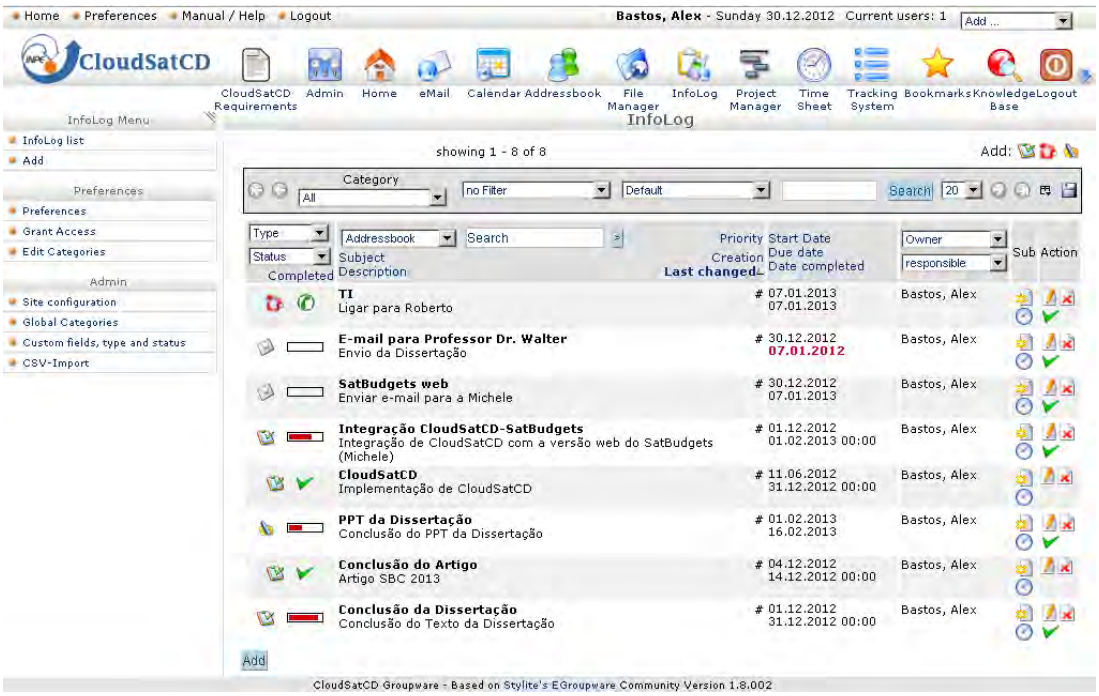


Figura 5.16 - *CloudSatCD Groupware Tool - Infolog*

[5] Extensão do protocolo HTTP para transferência de arquivos. Possui suporte a bloqueio de recursos, impedindo alterações ao mesmo tempo em um arquivo que esteja sendo editado.

Entre as principais funções do serviço *Infolog* destacam-se (EGROUPWARE, 2011): organização de processos de negócios por tipos de tarefas individuais; criação de tarefas e subtarefas; uso de filtragem com critérios pré-definidos para gerenciar tarefas; inserção e visualização de links e arquivos anexados; filtragem flexível por parâmetros variados (projeto, *status*, tipo, categoria, última modificação, entre outros); relacionamento entre tarefas e o gerenciador de projetos (*Project Manager*); visão CRM que exibe tarefas por contatos e/ou organização; delegação de tarefas a usuários e/ou grupos; definição de início, status, prioridades, porcentagem completada e devidas datas; ligação entre tarefas e outros dados do *groupware*, como contatos e projetos; inserção de arquivos a uma tarefa; sistema de busca completo; uso de ACL; funções de cópia e impressão; sincronização de dados com clientes; entre outros.

Project Manager é o módulo para gerenciamento de projetos que permite planejar, organizar e controlar a execução de processos, visando atingir de forma controlada e com esforço comum os objetivos dos projetos. Possui forte integração com as outras aplicações existentes como o gerenciamento de tarefas (*Infolog*), calendário (*Calendar*) e sistema de rastreio (*Tracking System*) para obtenção de fonte de dados, links e troca de informações. É possível criar dependências ou ligações entre projetos, de modo que, por exemplo, um projeto seja definido como parte de outro projeto (subprojeto).

Ao gerir um projeto disponibiliza-se um conjunto de ferramentas e regras definidas para a automatização parcial ou total de processos na gestão, tornando possível administrar, monitorizar e controlar os diversos processos de seu ciclo de vida.

A Figura 5.17 exibe um exemplo de página inicial do gerenciador contendo a lista de projetos nos quais o usuário é um participante, ao centro, enquanto na lateral esquerda é exibido um menu de ações, opções e preferências disponíveis no *Project Manager*, dentre as quais é importante salientar a lista

de elementos (participantes), o diagrama de Gantt[6] (*Gantt Chart*) e a lista de preços (custos).

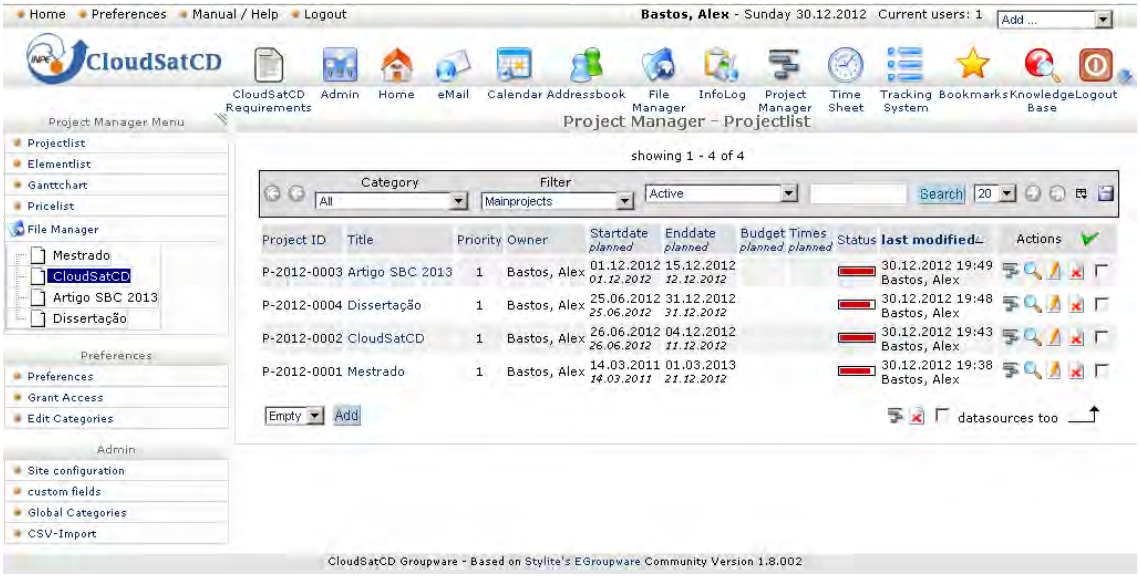


Figura 5.17 - *CloudSatCD Groupware Tool - Project Manager*

A Figura 5.18 ilustra um exemplo simples de visualização de *status* de um projeto selecionado, exibindo a porcentagem de trabalho concluída em relação aos prazos estabelecidos e aos marcos (*milestones*) definidos para a criação do *Gantt Chart*, tornando perceptível toda a estrutura e o planejamento feito do projeto. Através dessa interface também é possível adicionar, alterar ou excluir marcos, definir datas de início, início planejado, fim, fim planejado, nível de profundidade utilizado, restrições, recursos e filtros.

Uma ferramenta como *Gantt Chart* torna-se uma poderosa aliada no gerenciamento e execução de projetos, principalmente por permitir gerenciar também tempo (cronograma) e recursos relacionados aos componentes da EAP (Estrutura Analítica do Projeto), além da coleta de dados e sequenciamento e monitoramento de atividades.

[6] Tipo de gráfico desenvolvido por Henry Gantt bastante utilizado para exibir o avanço das etapas de um projeto em intervalos de tempo que representam início e fim de cada fase em forma de barras coloridas sobre o eixo horizontal do gráfico.

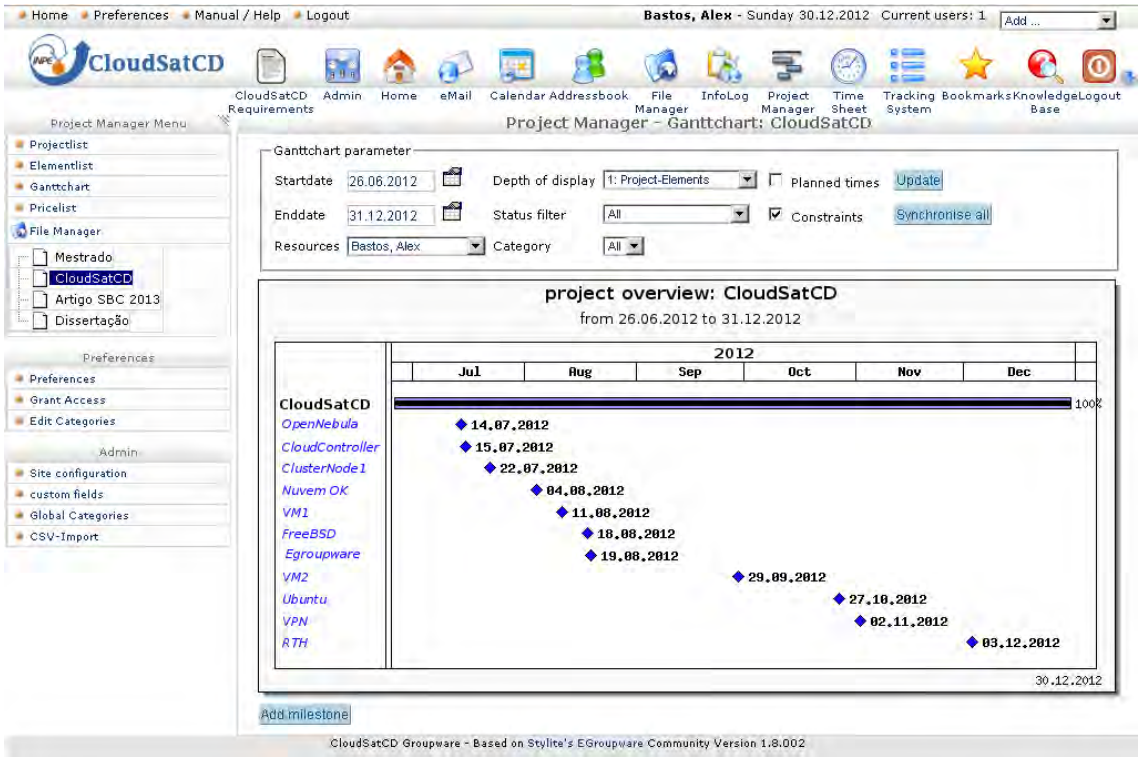


Figura 5.18 - *CloudSatCD Groupware Tool - Project Manager GanttChart*

Entre as principais funcionalidades do módulo *Project Manager* destacam-se (EGROUPWARE, 2011): visualização e ordenação de projetos por ID (*Identity Document*) do projeto ou título; uso de “árvore” de projeto para navegação; filtragem flexível por principais e subprojetos; exibição do progresso do projeto com *status*, datas e orçamentos (*budgets*); visualização de todas as entradas relacionadas ao projeto na lista de elementos; visualização dos processos do projeto em diagramas de Gantt (*Gantt Chart*) no nível preferido; criação de listas de custo com informações detalhadas sobre custo unitário, validação e disponibilidade em geral ou sobre um nível de projeto; exibição de arquivos armazenados relacionados ao projeto; criação e adição de projetos hierárquicos; criação e uso de vários *templates* de projetos; planejamento e gerenciamento de equipe e orçamento (*budget*); definição de marcos (*milestones*) e restrições; captura de horas trabalhadas e custo; adição de novas ou existentes entradas a partir de outras aplicações; anexação de

arquivos ao projeto; criação de campos extras para informações adicionais; função de histórico; uso de ACL; definição de tipos de contas permitidas; entre outros.

Time Sheet é uma aplicação em forma de planilha para rastreio de atividades através do tempo, totalmente integrada ao gerenciador de projetos (*Project Manager*). Tais planilhas de tempo possibilitam o gerenciamento e auditoria da documentação de tempo de trabalho, como também podem ser vinculadas a tarefas, projetos, contatos ou quaisquer outros tipos de informações no *groupware*.

A Figura 5.19 demonstra um simples exemplo de página inicial do *Time Sheet*, onde ao centro são encontradas algumas informações essenciais relacionadas a atividades e/ou projetos cadastrados, enquanto na lateral esquerda é exibido um menu para configuração de preferências gerais.

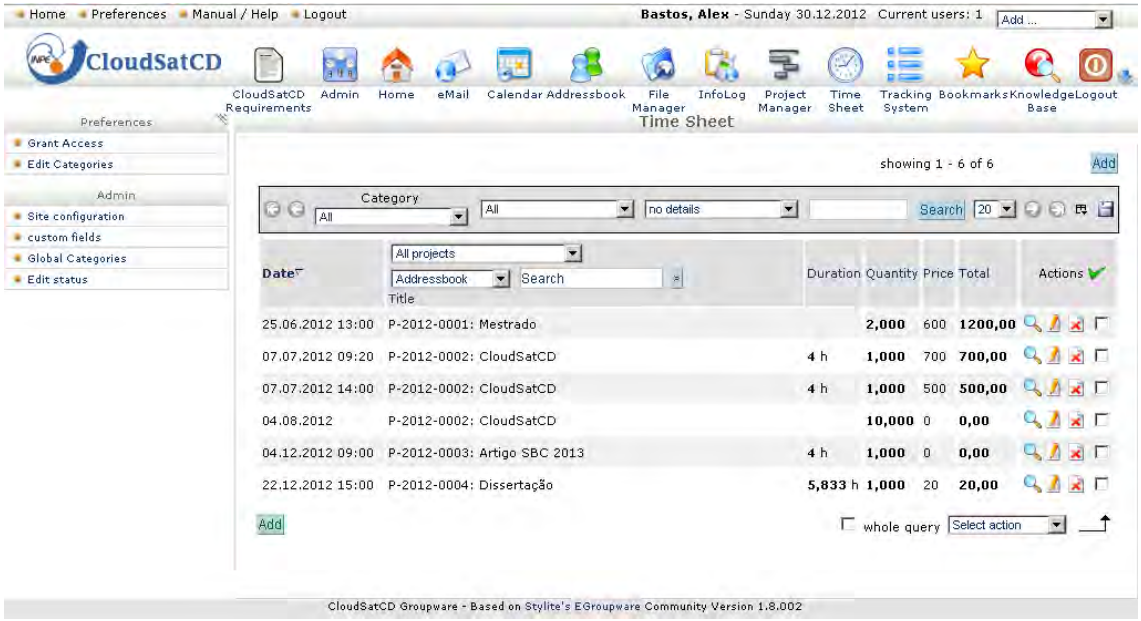


Figura 5.19 - *CloudSatCD Groupware Tool - Time Sheet*

As principais funcionalidades do *Time Sheet* são (EGROUPWARE, 2011): organização do tempo da documentação em categorias e *status* diferentes;

análise de dados por diferentes períodos de tempo para exibir tempos da documentação e orçamentos (*budgets*) acumulados; critérios de seleção por projetos, categorias, *status*, datas, usuário e contato; criação de entradas na planilha de tempo com descrições detalhadas; definição de data, duração e quantidade; links e arquivos anexados; busca completa; função de histórico; uso de ACL; criação de campos extras para informações adicionais; possibilidade de restringir as mudanças de status para o administrador apenas; entre outros.

Tracking System é uma aplicação para sistema de rastreamento (*Tracker*) e gerenciamento de serviços que permite gerenciar, auditar e organizar os processos e gestão de incidentes de forma padronizada e eficiente, exibindo na interface todas as filas do *tracker,* que retratam os processos e como tratá-los individualmente, onde cada fila contém *tickets* (entradas) diferentes, que podem ser alocados para várias categorias e responsáveis, além de indicar o nível de gravidade e/ou prioridade de cada entrada no sistema.

Esse módulo pode ser utilizado também para auxiliar na gestão de mudanças e quaisquer outros tipos de processos que necessitem de uma estrutura de delegação definida e escalabilidade de procedimentos. Além de possuir forte ligação com os módulos *Infolog*, *Address Book*, *Project Manager* e *Time Sheet*, possibilita a visualização do progresso do documento de trabalho por módulos de texto pré-definidos e a impressão de recibos de trabalho para apresentação.

A Figura 5.20 exibe algumas entradas no *tracker*, onde ao centro são encontradas algumas informações essenciais relacionadas aos *tickets*, enquanto na lateral esquerda é exibido um menu para as preferências do módulo.

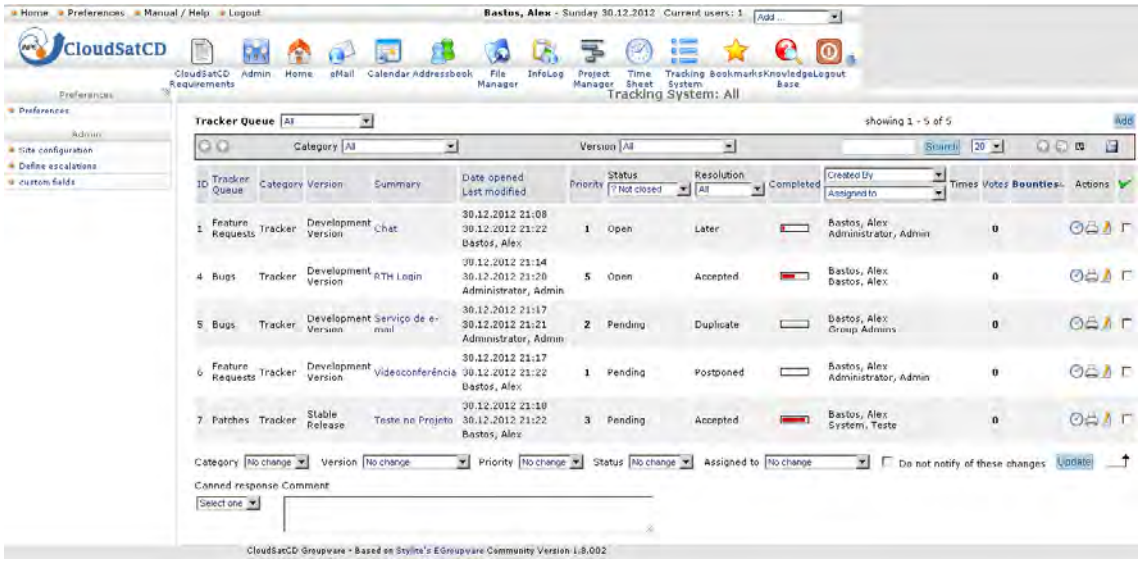


Figura 5.20 - *CloudSatCD Groupware Tool - Tracking System*

As principais funcionalidades do *Tracking System* são (EGROUPWARE, 2011): gerenciamento de serviços em filas de tickets diferentes; estrutura de filas de serviços com categorias e versões; acompanhamento da qualidade do serviço por *status* e resoluções; filtragem flexível por filas, categorias, versões, *status* e resoluções; exibição de entradas acumuladas na folha de tempo (*Time Sheet*); criação de *tickets* e adição de comentários; atribuição de *tickets* para usuários e grupos; autoatribuição de *tickets* de acordo com a categoria; definição de múltiplas estruturas para escalonamento da qualidade do serviço; definição de "atrasado e/ou "fechamento pendente" automaticamente; filas interligadas em projetos; entradas vinculadas e arquivos anexos; gerenciamento de múltiplos *tickets* de uma vez; criação de modelo de resposta, pré-visualização e edição de texto ao comentar; impressão de recibos de trabalho para apresentação; recebimento de notificações por e-mail; busca completa; uso de ACL; função de histórico; restrição de *tickets* para o criador ou grupo; entre outros.

Bookmarks é um aplicativo de marcadores que possibilita ao usuário salvar e compartilhar links (favoritos). Seu funcionamento é semelhante aos "Favoritos" dos navegadores, mas ao contrário destes, que salvam localmente, os links

ficam salvos no *groupware*, podendo ser acessados de qualquer lugar e navegador.

A Figura 5.21 exibe algumas marcações feitas e categorizadas ao centro, enquanto na lateral esquerda é exibido um menu de ações e configuração de preferências do módulo.

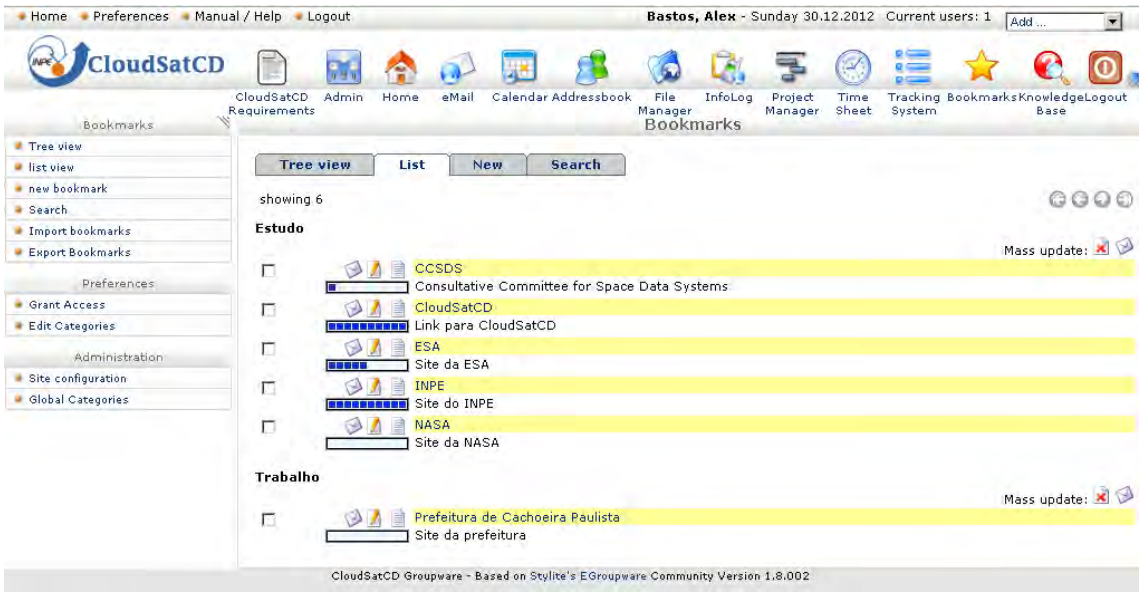


Figura 5.21 - *CloudSatCD Groupware Tool - Bookmarks*

Dentre as funcionalidades do módulo *Bookmarks* estão (EGROUPWARE, 2011): publicação e compartilhamento de links; estruturação de favoritos em categorias; classificação de favoritos; exibição em listas e árvores; uso de ACL; importação e exportação de links; pesquisa avançada por favoritos; envio de e-mail contendo links a partir do módulo; permissões e privilégios de acesso em nível de grupos e usuários individuais; entre outros.

Knowledge Base é uma aplicação para gestão do conhecimento que permite aos usuários construírem uma base de conhecimento para armazenar, pesquisar e recuperar competências de forma a auxiliar as pessoas encontrarem recursos, minimizando tempo e custo através da reutilização do conhecimento.

Tal recurso possibilita criar uma grande base sólida de conhecimento e/ou informação, casos e lições aprendidas para facilitar a resolução de problemas, funcionando também como suporte aos usuários, colaborando uns com os outros.

A Figura 5.22 ilustra um exemplo com artigos e questões criadas e categorizadas ao centro, enquanto na lateral esquerda é exibido um menu de ações onde é possível criar artigos, fazer e responder perguntas (questões), gerenciar a base, gerenciar categorias e configurar as preferências do módulo.

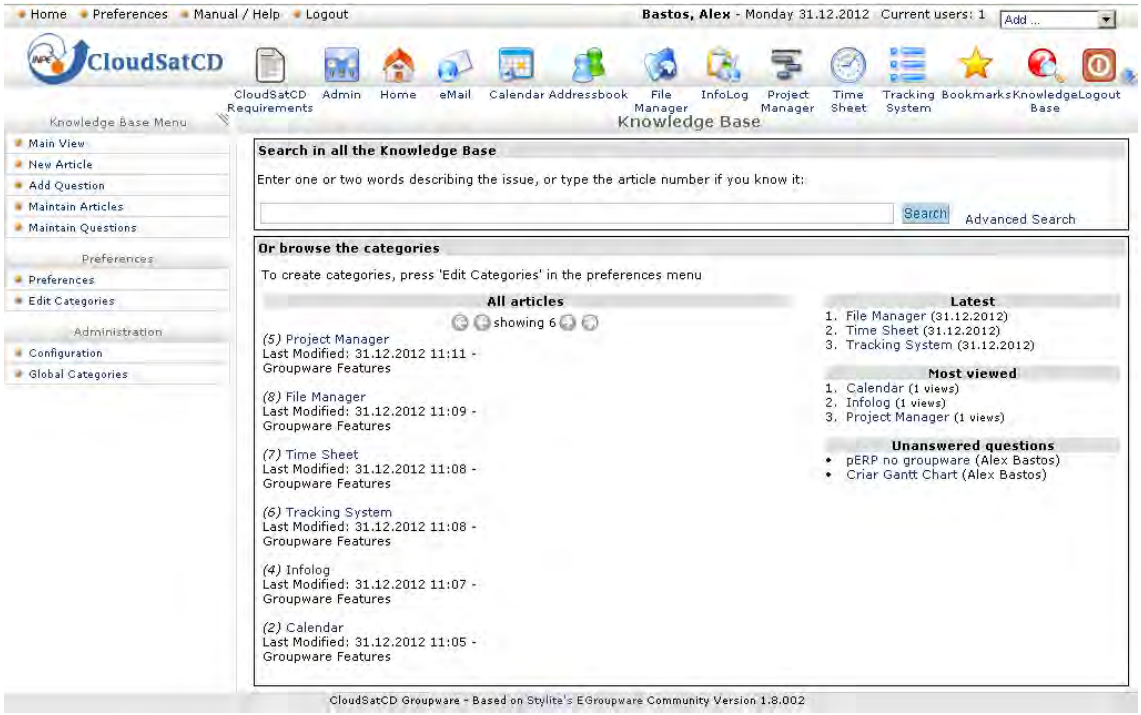


Figura 5.22 - *CloudSatCD Groupware Tool - Knowledge Base*

Entre as características do módulo *Knowledge Base* é possível destacar (EGROUPWARE, 2011): criação de categorias e subcategorias; preservação de artigos e questões; exibição das questões não respondidas, mais antigas e mais vistas; edição de artigos; ligação de artigos com outras aplicações como projetos, páginas *wiki*, entradas no *tracker*, entre outros; anexação de arquivos e imagens a partir do gerenciador de arquivos; busca avançada por diversos

critérios como palavras-chave e ID de artigos; exibição de comentários e classificações dos artigos; uso de ACL; definição de aprovação para publicação de artigos; histórico de alterações por data e usuários; impressão de artigos e questões; inserção de links e arquivos; entre outros.

Site Manager é a aplicação para criação e gerenciamento de conteúdo dinâmico e sites na web. Sua interface é simples e semelhante a um editor de texto, e não há necessidade de conhecimento prévio sobre linguagens de programação web como, por exemplo, HTML.

A Figura 5.23 ilustra um exemplo um site baseado no *template default* ao centro, onde é possível manipular todo seu conteúdo através de botões de ações dentro de cada seção do site, enquanto na lateral esquerda é exibido um menu de ações que permite configurar site, gerenciar suas propriedades, gerenciar notificações, gerenciar traduções, aceitar mudanças (*commit*), gerenciar conteúdo arquivado, visualizar e editar o site gerado e outros existentes, definir novos sites, e configurar as preferências do módulo.

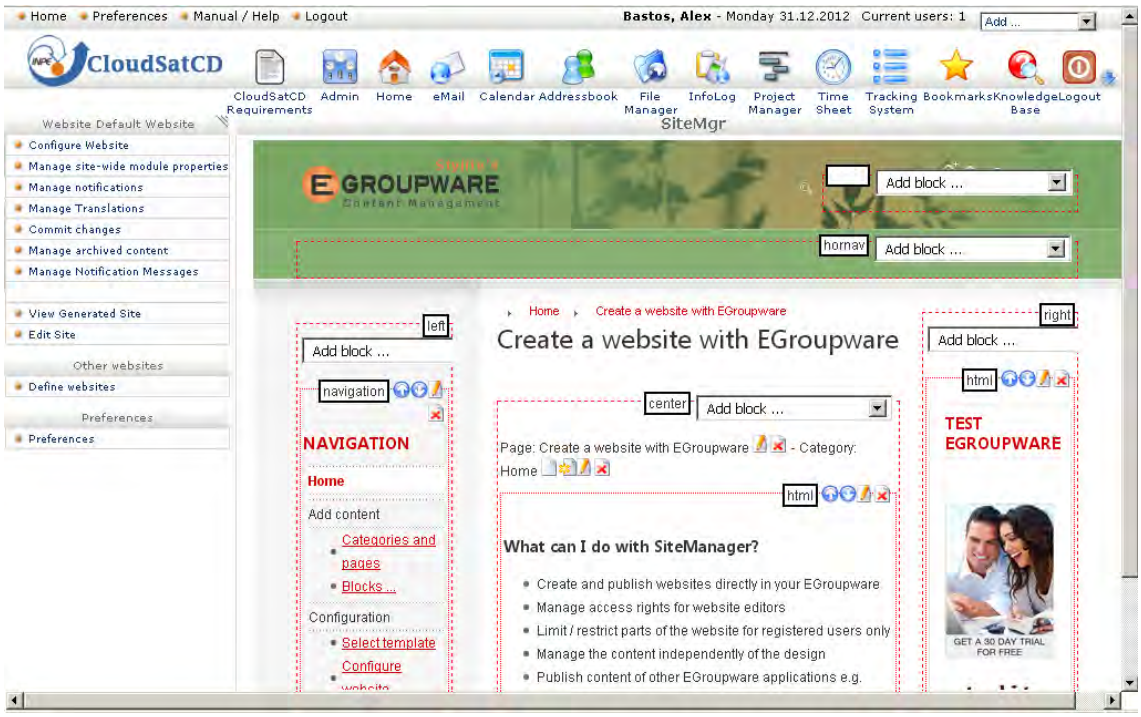


Figura 5.23 - *CloudSatCD Groupware Tool - Site Manager*

O conteúdo pode ser produzido por diferentes módulos que buscam dados de aplicações do *groupware* ou constroem a página a partir de argumentos especificados. Adicionalmente, pode lidar com inúmeros sites e com diferentes versões do mesmo conteúdo, possibilitando trabalhar em uma versão preliminar sem alterar a versão em produção.

Dentre os principais recursos disponibilizados pelo módulo *Site Manager* é possível destacar (EGROUPWARE, 2011): uso de *templates* no gerenciador de sites; gerenciamento de conteúdo por categorias, páginas e blocos; publicação de informação a partir do calendário, catálogo de endereços, sistema de rastreio, gerenciador de arquivos, notícias e *wiki*; geração de conteúdo a partir de fontes externas; uso de diferentes possibilidades de navegação; gerenciamento de múltiplos *websites*; configuração das propriedades do site; uso de versões diferentes para gerenciamento de publicação e arquivamento de conteúdo; definição do administrador do site para cada *website* individualmente; definição de permissões de leitura/escrita; possibilidade de restringir o conteúdo a usuários registrados; importar/exportar; entre outros.

Resources é um módulo de gerenciamento de recursos (inventário) que permite gerir os recursos disponíveis e inventário, e que também atua como uma ferramenta para fazer reservas de forma integrada a outras aplicações, principalmente ao calendário. Possibilita também documentar e gerenciar recursos, reservar recursos para compromissos e eventos da agenda corporativa, e ordená-los por categorias e campos personalizados.

A Figura 5.24 exibe um exemplo de um pequeno e simples cadastro de recursos disponíveis no centro da página, onde é possível visualizar e manipular todo seu conteúdo através de botões de ações, enquanto na lateral esquerda é exibido um menu de ações que permite listar e adicionar recursos, além de customizar categorias, permissões de acesso e campos personalizados.

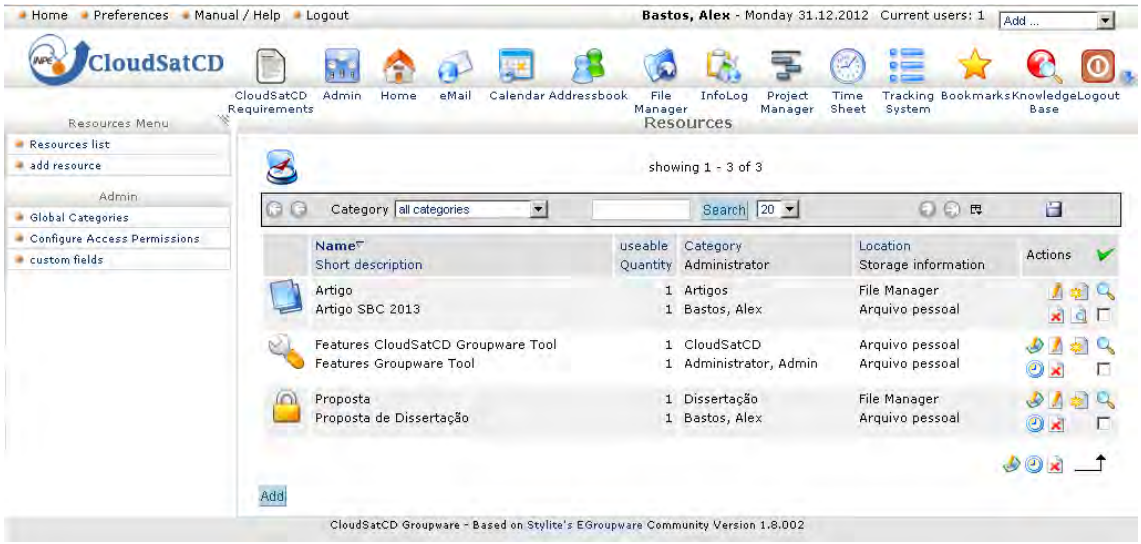


Figura 5.24 - *CloudSatCD Groupware Tool - Resources*

O módulo *Resources* tem como as principais características (EGROUPWARE, 2011): estrutura de recursos com categorias e subcategorias; exibição de recurso ou categoria no calendário relacionado; catálogo de recursos para compromissos; adição de descrição e figuras; definição da localização e armazenamento de informação; definição de quantidade, disponibilidade e informações de reserva; links de recursos para outros dados como contatos e projetos; arquivos anexos; definição de gerentes de recursos para aprovação de reservas; uso de ACL; entre outros.

Wiki é um aplicativo que oferece uma ferramenta para criação de documentos colaborativos e potencializa o conhecimento coletivo, onde cada página de um *wiki* é semelhante a uma página web, exceto pelo fato de que qualquer pessoa pode editá-la, desde que autorizada, e que não há necessidade de experiência em edição.

O *wiki* cria automaticamente links para outras páginas *wiki*, bastando incluir palavras em uma frase digitada, e se elas já forem páginas, o link aparece automaticamente. Isso possibilita estruturar a informação e organizar conhecimento e documentação em uma estrutura interligada de páginas *wiki*

diferentes, e facilita encontrar o conteúdo através do índice de pesquisa, sem deixar de proteger as informações por privilégios de acesso a usuários e grupos em nível de página.

A Figura 5.25 exibe um exemplo de uma página *wiki* no centro da página, onde é possível visualizar e manipular todo seu conteúdo através de links de ações, enquanto na lateral esquerda é exibido um menu de opções administrativas e um menu para visualizar mudanças recentes e configurar as preferências gerais.

Figura 5.25 - *CloudSatCD Groupware Tool - Wiki*

Entre as principais *features* do módulo *Wiki* é possível destacar as seguintes (EGROUPWARE, 2011): interligação de estruturas de páginas *wiki*; criação de novas páginas facilmente; renomeação de páginas e obtenção de todos os links automaticamente modificados; tradução de páginas em diferentes idiomas; uso do formato de escrita *wiki* ou uso de editor de páginas conveniente; inserção de links para arquivos e páginas web; publicação das

páginas *wikis* em *websites*; visualização de mudanças recentes; busca completa; uso de ACL; função de histórico; definição de prazo para manter versões de páginas; permitir/negar acesso anônimo; entre outros

News Admin é uma aplicação para publicação e leitura de notícias corporativas que permite também a criação de *feeds* RSS[7] (*Really Simple Syndication*) para divulgação. Através deste módulo é possível publicar conteúdo de destaque e outros temas relacionados aos projetos, deixando os participantes rapidamente informados sobre diversos assuntos.

A Figura 5.26 exibe um exemplo com notícias publicadas no centro da página, onde é possível ler e administrar o conteúdo, enquanto na lateral esquerda é exibido um menu de opções administrativas, configurações do serviço e leitura de notícias.

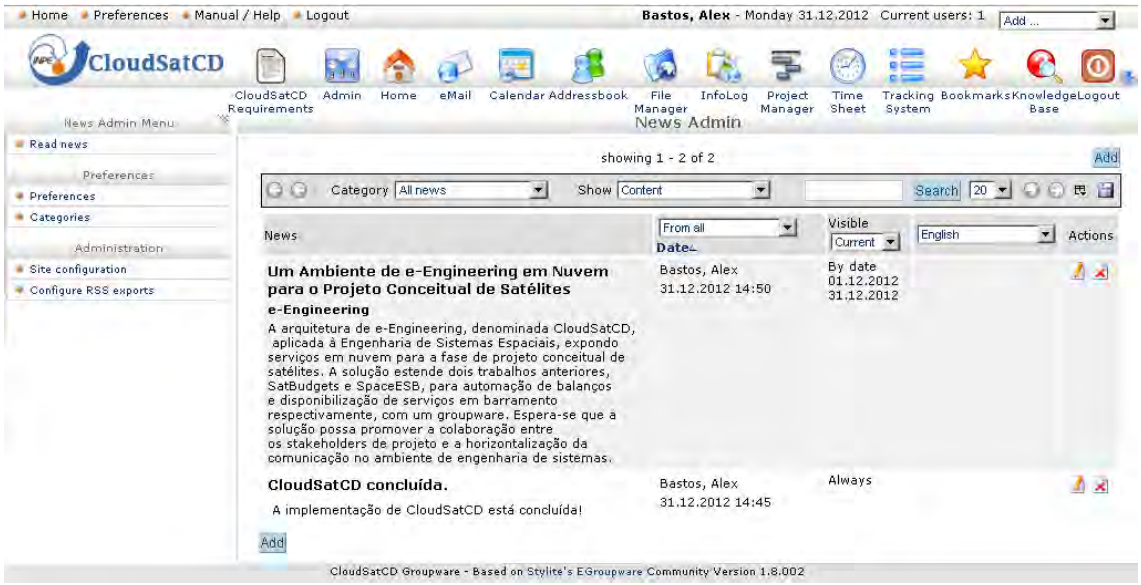


Figura 5.26 - *CloudSatCD Groupware Tool - News Admin*

Uma vez publicadas, aparecem tanto na página inicial da aplicação *News Admin* quanto na página de entrada do *groupware* ao fazer *logon*, configurada

[7] Padrão desenvolvido em XML para agregar conteúdo e permitir aos *sites* divulgarem atualizações a partir de links e resumos de notícias. Com um leitor de RSS (agregador), o usuário inscrito pode receber os *feeds* e permanecer informado sem precisar acessar o *site*.

dessa forma como *default* em *CloudSatCD Groupware Tool*, onde é exibida sua *homepage* com o calendário e as notícias, sendo estas últimas mostradas com o título dado à notícia em forma de link, que leva à notícia completa.

Destacam-se dentre as principais características do serviço *News Admin* (EGROUPWARE, 2011): publicação de notícias para diferentes grupos; publicação de notícias por data; estruturação de notícias em categorias e subcategorias; seleção de critérios por categorias, visibilidade, idiomas e criador das notícias; busca completa; uso de ACL; tradução de notícias; configuração e gerenciamento de RSS; programação para importar/exportar notícias; entre outros.

Polls é um serviço de enquetes onde é possível determinar as opções gerais para uma pesquisa de opinião permitindo que usuários escolham, dentre as respostas pré-configuradas, a opção que melhor responde à questão feita. Uma vez votada, a enquete exibe os resultados parciais com os resultados computados até o momento.

A Figura 5.27 exibe um exemplo de enquete publicada no centro da página, onde há uma situação de múltipla escolha para a resposta à questão, enquanto na lateral esquerda é exibido um menu de opções administrativas e um menu para visualização de enquetes e resultados.

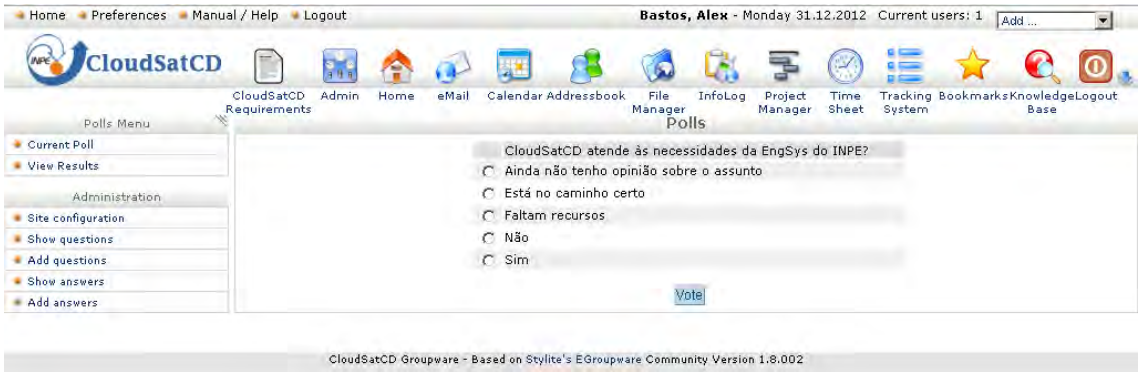


Figura 5.27 - *CloudSatCD Groupware Tool - Polls*

Entre suas principais funcionalidades é possível destacar (EGROUPWARE, 2011): Obtenção de *feedback* anônimo às perguntas em enquetes; adição de conjuntos de perguntas e respostas; visualização de resultados; seleção de enquetes; definição de permissões de direito de voto em nível de usuário ou grupo; limitação de resultados em nível de usuário e/ou grupo; entre outros.

A solução dispõe ainda de outras aplicações voltadas para desenvolvedores como ferramentas de tradução (*Translation Tools*) para alternar a linguagem de outras aplicações do sistema; ferramenta para edição e criação de *templates* (*eTemplate*) utilizados no sistema; gerenciamento de servidor Samba[8] (*Samba Admin*) para compartilhamento de arquivos em rede; e ferramenta para visualização e gerenciamento de site construído no *Site Manager* (*Website*).

Outros serviços como gerenciamento de sincronização de dados, informações sobre o PHP do sistema (*Phpsysinfo*); configurações de registro (*Registration*); importação/exportação (*Import/Export*) e gerenciamento de notificações (*Notifications*) também estão disponíveis, entretanto restritas ao grupo dos administradores.

5.2.2. CloudSatCD Requirements Tool

Requirements é um serviço para gerenciar os requisitos de projetos armazenando-os sob controle de versão e o sistema registra a data, hora e usuário que inseriu o requisito, além de permitir a criação de atribuições para cada versão, seja para liberação (*release*) ou para um indivíduo.

Tal recurso possibilita encontrar rapidamente todos os requisitos atribuídos a uma determinada versão ou usuário utilizando a funcionabilidade do filtro na página principal de requisitos. Existem várias áreas relacionadas com os requisitos, permitindo aos usuários a associar o tipo de documento, funcionalidade, área de cobertura, e prioridade para cada versão.

[8] Aplicação que permite o gerenciamento e compartilhamento de recursos em redes formadas por computadores com sistemas Windows e sistemas baseados em Unix.

Ao gerir os requisitos consegue-se estabelecer e sistematizar as necessidades do sistema, bem como suas restrições, permitindo que objetivos e metas relacionados sejam alcançados. Baseado nos requisitos do sistema, arquiteturas de satélites podem ser avaliadas buscando-se a uma solução ótima.

A Figura 5.28 exibe um exemplo com uma pequena lista de requisitos para o projeto selecionado no centro da página, onde estes podem ser ordenados através das colunas de informações disponíveis, detalhados ao serem selecionados individualmente, e filtrados com base em critérios pré-estabelecidos. Na parte superior, logo abaixo da Barra de Links Rápidos, encontra-se outra Barra de Links Rápidos com as opções existentes para esse serviço - *Requirments*, *Folder View*, *Add Requirement - Record*, *Add Requirement - File* e *Notifications* -, podendo haver inclusive sub-opções, dependendo da funcionalidade selecionada.

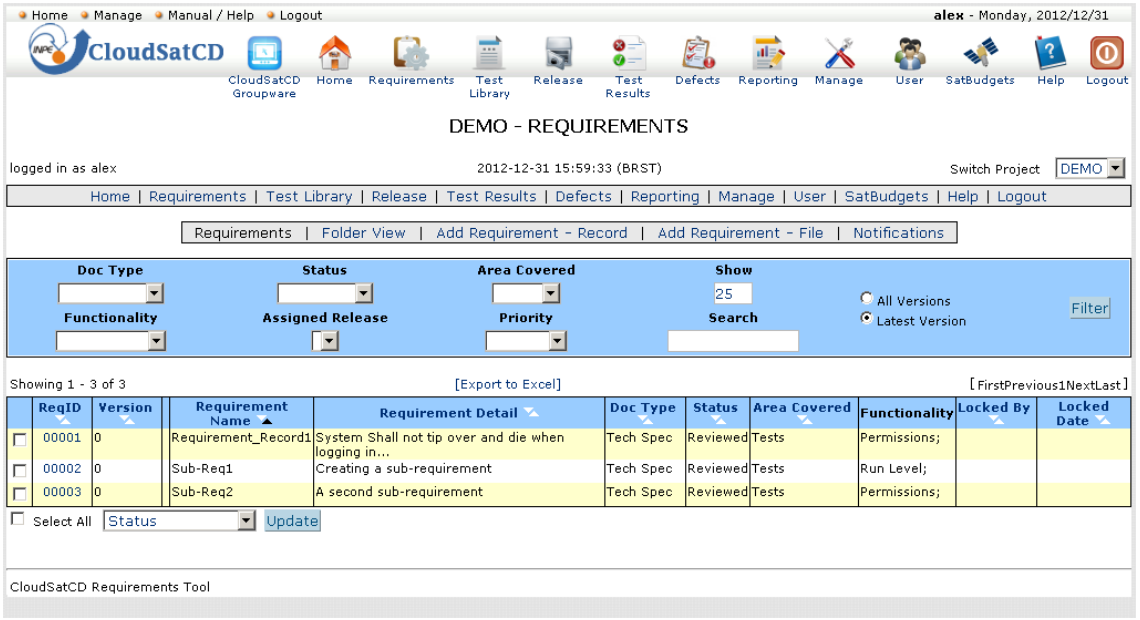


Figura 5.28 - *CloudSatCD Requirements Tool - Requirements*

Entre as principais características do módulo *Requirements* é possível destacar: adição de novos requisitos; adição de requisitos a partir de um

arquivo de requisitos; controle de versão; visualização de requisitos; detalhamento de requisitos; filtragem e pesquisa avançada de requisitos segundo critérios pré-estabelecidos; visualização do histórico de requisitos; associação requisitos a outros requisitos; associação de requisitos a testes; notificações de mudanças através de e-mail; visualização da árvore de diretórios de requisitos; exportação da lista de requisitos; entre outros.

Test Library é o módulo de teste do sistema e se relaciona de alguma forma com outros módulos (*Requirements*, *Release*, *Test Library*, e *Defects*), embora todos possam ser utilizados de forma independente, dependendo do grau de integração desejado.

Este módulo é destinado a fomentar uma equipe de teste distribuída ou co-localizada e, por essa razão, muitos dos campos relacionados aos testes podem não ser necessários caso haja uma equipe de teste local e pequena, mas há flexibilidade e suporte suficientes para equipe distribuídas. *BA Owner* e *QA Owner*, por exemplo, assumem que há uma divisão de trabalho entre o Analista de Negócios (BA) que está escrevendo um requisito ou teste e o engenheiro de controle de qualidade (QA), que automatiza ou executa o teste.

O módulo de teste é projetado para armazenar todos os testes. O sistema é projetado para fornecer um repositório único para todos os testes e resultados de testes, sejam eles testes de unidade, teste manual, ou testes automatizados. É possível também associar determinados metadados a fim de ajudar a organizar os testes.

A Figura 5.29 exibe um exemplo de testes para o projeto selecionado no centro da página, onde estes podem ser ordenados através das colunas de informações disponíveis, detalhados ao serem selecionados individualmente, e filtrados com base em critérios pré-estabelecidos. Na parte superior, logo abaixo da Barra de Links Rápidos, encontra-se outra Barra de Links Rápidos com as opções existentes para esse serviço - *Test Library*, *Add Test* e *Test*

Workflow -, podendo haver inclusive sub-opções, dependendo da funcionalidade selecionada.

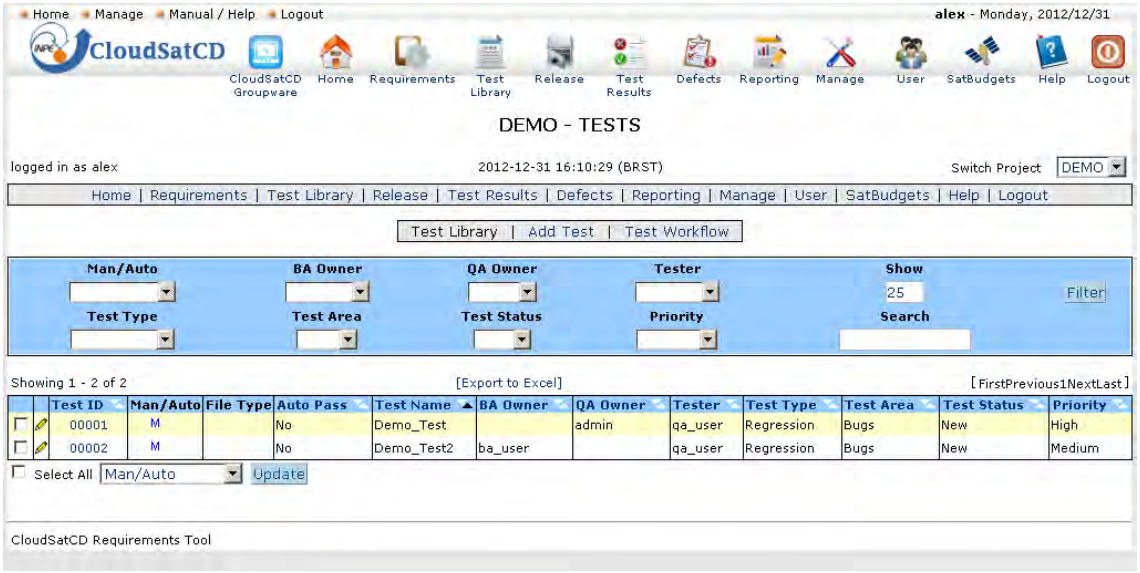


Figura 5.29 - *CloudSatCD Requirements Tool - Test Library*

Dentre as principais funcionalidades do módulo *Test Library* destacam-se: adição de novos testes; visualização de testes; detalhamento de testes; adições de estágios/etapas de testes; associação de testes a requisitos; definição de status de testes, área de testes, tipo de testes, analista de negócios, controle de qualidade, testador para cada teste; exportação da lista de testes; importação de etapas de testes; associação de documentos a testes; definição do fluxo de trabalho dos testes; entre outros.

Release é o modulo que permite aos usuários adicionar *releases*[9] e *builds*[10] a um projeto, além de possibilitar que requisitos e defeitos sejam atribuídos a qualquer *release*. Este recurso foi projetado para ajudar também no Gerenciamento de Escopo/Release, facilitando ver a carga de trabalho para uma determinada versão.

[9] Versão do sistema validada após diversos tipos de testes, com garantia de que todos os itens foram devidamente testados, avaliados, aceitos e disponíveis na nova *baseline*.
[10] Versão ainda incompleta do sistema em desenvolvimento, mas com certa estabilidade. Normalmente apresenta limitações conhecidas e espaço para integração de funcionalidades.

Os resultados de toda a execução de teste são relacionados de volta a um *release*, *build* e conjunto de teste (*test set*), sendo este último um conjunto de teste é um grupo de testes planejados para executar contra um *build* e *release* específicos. Sem a adição de um *release* e *build*, não é possível executar qualquer teste.

A Figura 5.30 seguir exibe a página inicial do módulo, onde há uma seção para inserção de um *release* ao projeto selecionado no centro da página, enquanto a tabela logo abaixo lista os *releases* existentes, podendo estes ser ordenados através das colunas de informações disponíveis e detalhados hierarquicamente ao serem selecionados individualmente, exibindo *builds* e conjunto de testes relacionados.

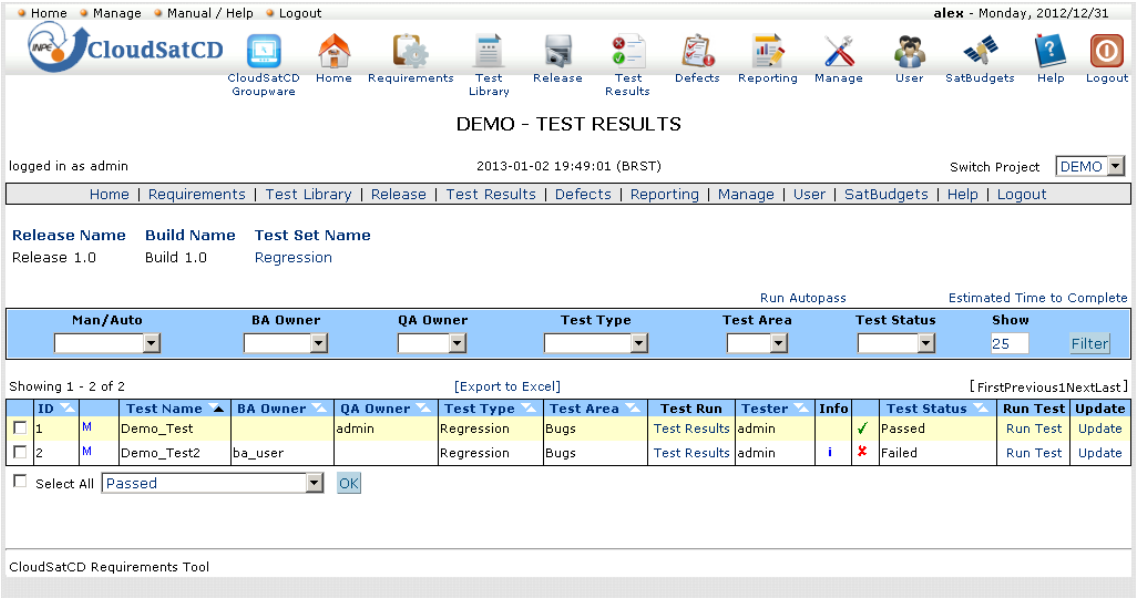


Figura 5.30 - *CloudSatCD Requirements Tool - Release*

Entre as principais *features* do módulo *Release* estão: adição e edição de um *release*; adição e edição de um *build*; adição e edição de um conjunto de teste (*test set*); visualização de um conjunto de teste; anexação de um arquivo de plano de testes; filtragem de conjunto de testes; cópia de conjunto de testes; inserção de *status* (*sign off*) em *releases*; entre outros.

Test Results é uma aplicação que contém um repositório central de todos os resultados de testes, tanto manuais quanto automatizados. Esse módulo solicita que o *release*, *build* e conjunto de testes adequados sejam selecionados para fornecer métricas precisas com todos os resultados dos testes armazenados para a seleção feita, podendo encontrar vários níveis de resultados.

A Figura 5.31 exibe os resultados de testes para um determinado *release*, *build* e conjunto de testes, hierarquicamente, onde estes podem ser ordenados através das colunas de informações disponíveis, detalhados ao serem selecionados individualmente, filtrados com base em critérios pré-estabelecidos, atualizados e pode ser executado um novo teste diretamente.

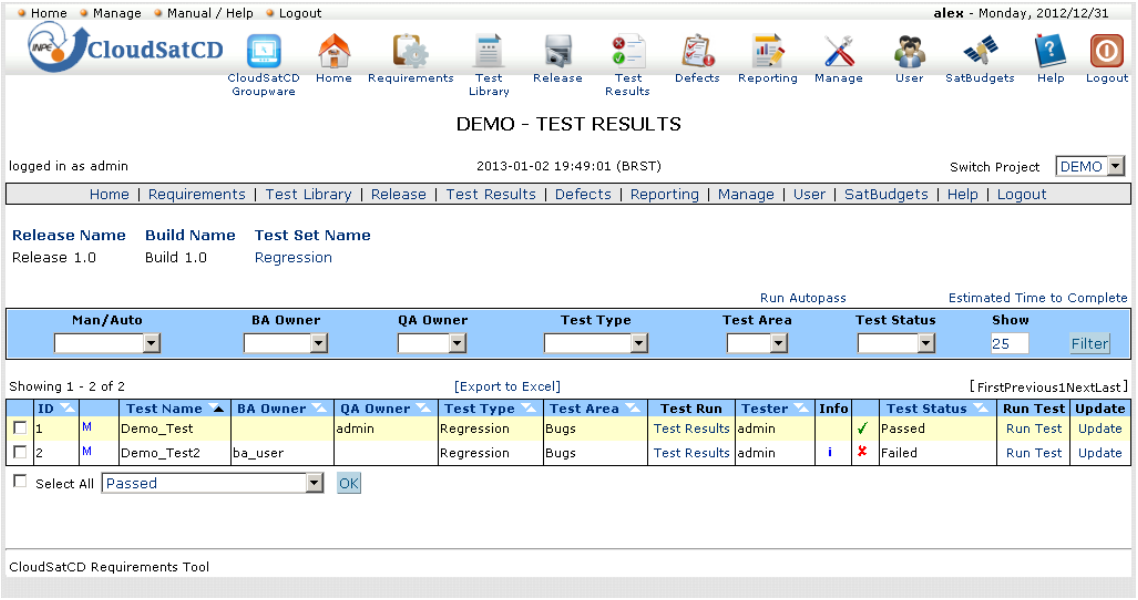


Figura 5.31 - *CloudSatCD Requirements Tool - Test Results*

Dentre os níveis de resultados encontrados, o primeiro lista todos os testes planejados para executar no conjunto de teste. Cada teste tem um *status*, de modo que os usuários possam ter uma noção geral de como o teste está progredindo e ter uma visão geral da execução do teste.

O segundo nível permite visualizar quantas vezes o teste foi executado. O teste pode ter sido executado muitas vezes contra um *build* particular para ter passado (*Passed*). O terceiro nível permite visualizar as verificações individuais que foram realizadas. É possível ainda visualizar cada etapa realizada no teste e se esta passou (*Passed*) ou falhou (*Failed*).

Os principais recursos do módulo *Test Results* são: visualização de resultados de testes; visualização de teste executado; visualização de verificações; re-execução de testes diretamente a partir dos resultados; atualização de testes; filtragem avançada; visualização de documentação relacionada aos testes e seus resultados; visualização do tempo estimado de testes; execução de *autopass*; exportação dos resultados em planilhas; entre outros.

Defects é o modulo para registro, armazenamento, gerenciamento e rastreabilidade de falhas, além de atuar como um *tracker* para *bugs*. Tal aplicação permite localizar problemas reportados visando sua resolução através de análises e correções, maximizando a qualidade e minimizando riscos associados.

Esse módulo pode auxiliar no gerenciamento da configuração também, uma vez que esse processo visa controlar, auditar e gerenciar mudanças e conteúdo relacionados ao projeto, fazendo parte de todo o processo de desenvolvimento de um produto. Nesse contexto a rastreabilidade permite analisar e aprovar as possíveis solicitações de mudanças ocasionadas pelas falhas e, por conseguinte, a liberação de *baselines* de requisitos.

A Figura 5.32 exibe um exemplo com falhas encontradas no projeto selecionado no centro da página, onde estas podem ser ordenadas através das colunas de informações disponíveis, detalhadas ao serem selecionados individualmente, e filtradas com base em critérios pré-estabelecidos. Na parte superior, logo abaixo da Barra de Links Rápidos, encontra-se outra Barra de Links Rápidos com as opções existentes para esse serviço - *View Defects* e

Add Defect -, podendo haver inclusive sub-opções, dependendo da funcionalidade selecionada.

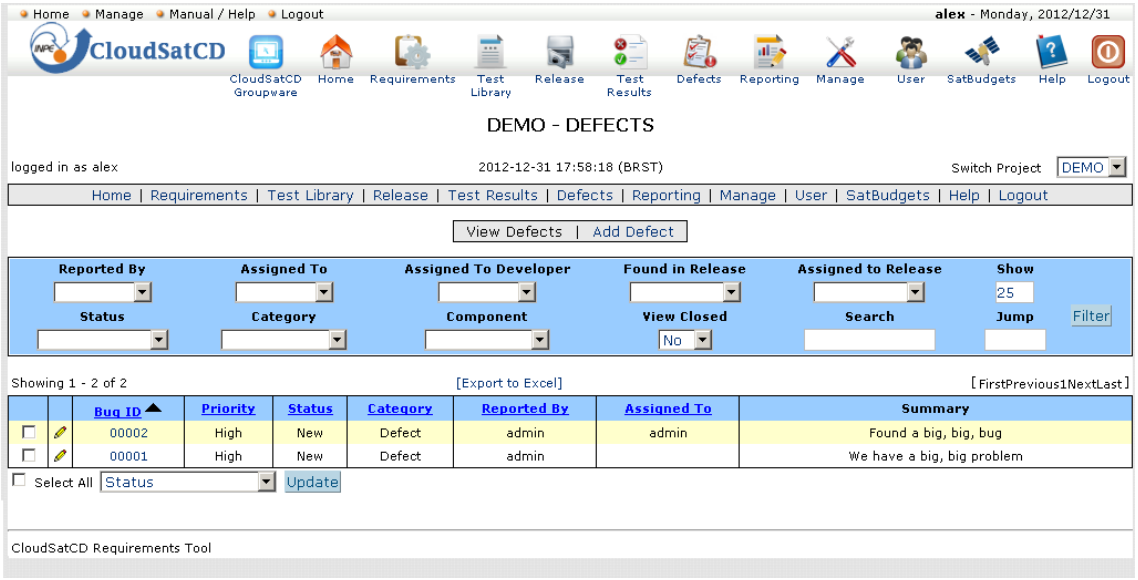


Figura 5.32 - *CloudSatCD Requirements Tool - Defects*

Entre os principais recursos do módulo *Defects* destacam-se: visualização de falhas; adição de falha/defeito; documentação de questões encontradas na seção de acompanhamento de problemas (*Reporting*); envio/upload de informações adicionais para falhas; histórico de falhas; identificação da falha, status, prioridade, severidade, data, por quem foi reportada, identificação da verificação, nome do projeto, componente; relacionamento entre falhas; árvore de falhas; filtro de busca para falhas baseada em diversos critérios; entre outros.

Reporting é o modulo para geração de relatórios e gráficos sobre o *status* de diversas áreas do sistema, tais como *Test Area*, *Build Status*, *Failed Verifications*, *Requirements Coverage*, *Test Signoff, Test Sets Status,* entre outros, possibilitando visualizar e analisar os resultados obtidos durante o projeto.

A Figura 5.33 exibe um exemplo com relatórios e gráficos de *signoff* para um determinado *release*, *build* e conjunto de testes (*test set*), hierarquicamente, onde é possível analisar diversos dados estatísticos relacionados.

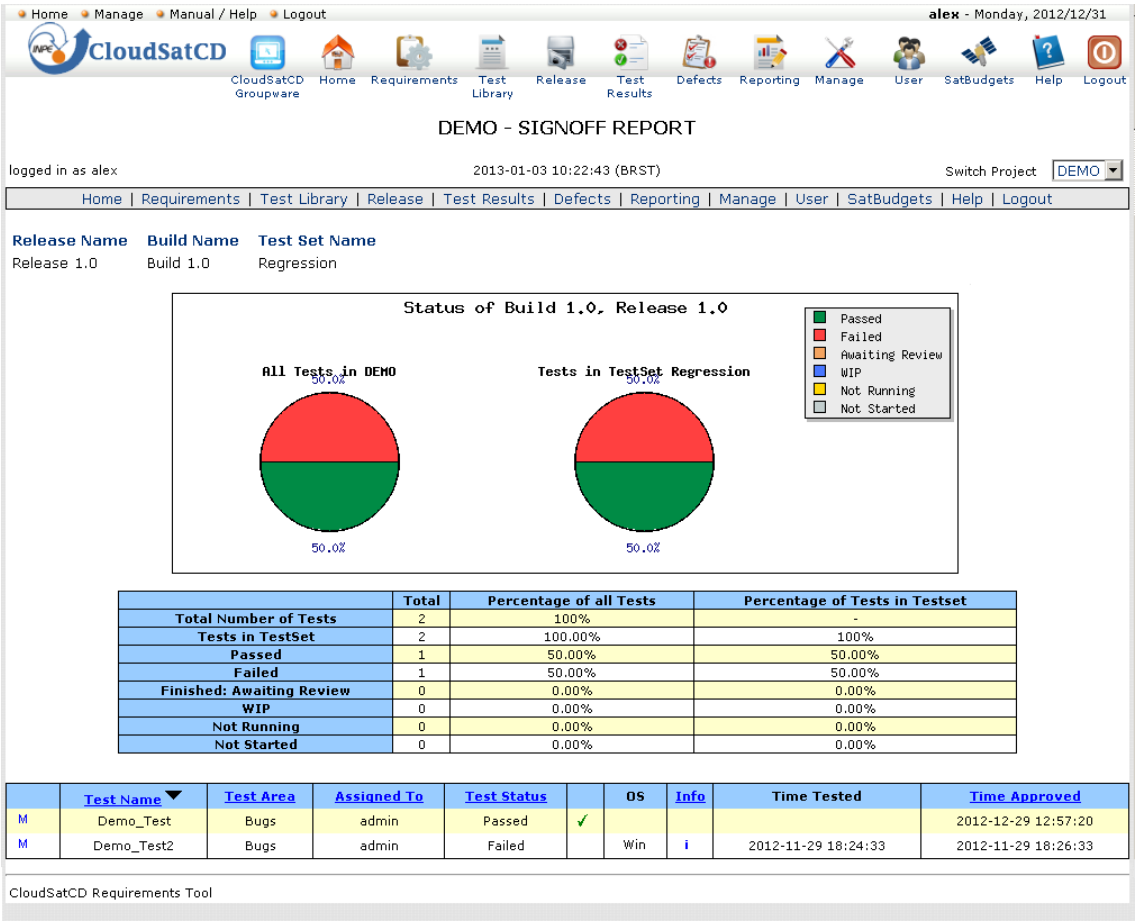


Figura 5.33 - *CloudSatCD Requirements Tool - Reporting*

Destacam-se entre as principais funcionalidades do módulo Reporting: criação de relatórios personalizados; visualização do número de testes executados pela área de teste para um determinado *release*, *build* e *test set*; exibição do *status* de todos os testes para um determinado *release* e *build*; visualização de informações detalhadas sobre todas as verificações falhas para um determinado *release*, *build* e *test set*; visualização da cobertura de requisitos para um determinado *release*, *build* e *test set*; exibição de informações sobre testes em um determinado *release*, *build* e *test set*, sendo útil para decidir

signoff em testes; visualização dos conjuntos de testes ordenados por data de criação; exibição informações sobre o que os testes estão sendo executados no momento; entre outros.

Manage é o serviço para gerenciamento de projetos baseado em requisitos, testes e falhas, tornando possível identificar antecipadamente quaisquer eventuais falhas ou não conformidades, otimizando tempo, custo e o uso de recursos, ao mesmo tempo em que garante o alinhamento entre as tarefas executadas e os objetivos estabelecidos.

A Figura 5.34 exibe um simples exemplo de área de cobertura de requisitos (*Requirement Area Covered*) para o projeto selecionado no centro da página, onde são encontradas informações disponíveis para as áreas de cobertura definidas. Na parte superior, logo abaixo da Barra de Links Rápidos, encontra-se outra Barra de Links Rápidos com as opções existentes para esse serviço - *Manage Project* e *Add Project* -, podendo haver inclusive sub-opções, dependendo da funcionalidade selecionada.

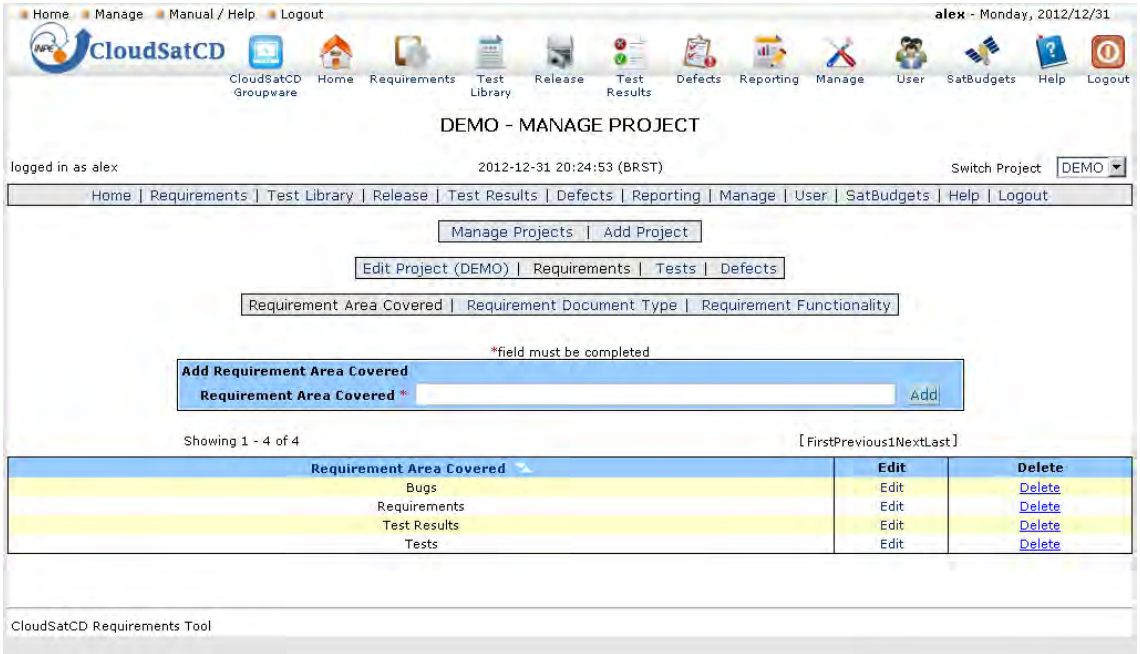


Figura 5.34 - *CloudSatCD Requirements Tool - Manage*

O administrador, ou gerente, do projeto pode também gerenciar as permissões-padrão e privilégios dos usuários em um projeto, e o envio de e-mail para notificações especificadas. A coordenação eficaz dos participantes e outros recursos envolvidos para a execução do trabalho é indispensável para alcançar o progresso e manter a interação entre os diversos participantes, visando à redução de riscos durante o ciclo de vida dos projetos.

Entre os principais recursos do módulo *Manage* é possível destacar: adição, edição e administração de projetos; adição de usuários cadastrados aos projetos; preferências de projetos para requisitos; preferências de projetos para testes; preferências de projetos para falhas; adição, edição e visualização de notícias (*news*) para projetos; administração de usuários em projetos; adição, edição e administração de áreas de coberturas de requisitos; adição, edição e administração de tipos de documentos de requisitos; adição, edição e administração de funcionalidades de requisitos; arquivamento de resultados de testes; arquivamento de testes; adição, edição e administração de áreas de testes; adição, edição e administração de tipos de documentos de testes; adição, edição e administração de ambientes de testes; adição, edição e administração de máquinas para testes, baseada em redes de computadores; adição, edição e administração de tipos de testes; adição, edição e administração de categorias de falhas; adição, edição e administração de componentes de falhas; entre outros.

User é o modulo para gerenciamento de usuários que permite alterar determinadas permissões para usuários específicos e, para usuários com direitos administrativos, possibilita adicionar novos usuários ao banco de dados do sistema.

Ao adicionar um usuário, um projeto padrão deve ser selecionado. Este é o primeiro projeto que irá aparecer quando o usuário fizer *logon* no sistema pela primeira vez. Depois que um usuário é adicionado ao sistema, pode ser associado a outros projetos por um administrador ou alguém com permissões

de gerenciamento (*manager*) para o projeto específico. O administrador ou gerente pode definir um papel (*manager*, *developer* ou *user*) e definir algumas das preferências relacionadas a notificações por e-mail para os usuários.

Se o administrador deseja conceder permissões diferentes para projetos diferentes, deve adicionar o usuário a um projeto com as permissões adequadas e adicioná-lo a outros projetos no módulo *Manage*, dando então ao usuário as permissões corretas para outros projetos.

A Figura 5.35 exibe um exemplo com os usuários cadastrados no projeto selecionado no centro da página, onde são encontradas informações disponíveis para estes. Na parte superior, logo abaixo da Barra de Links Rápidos, encontra-se outra Barra de Links Rápidos com as opções existentes para esse serviço - *My Account*, *All Users* e *Add New User* -, podendo haver inclusive sub-opções, dependendo da funcionalidade selecionada.

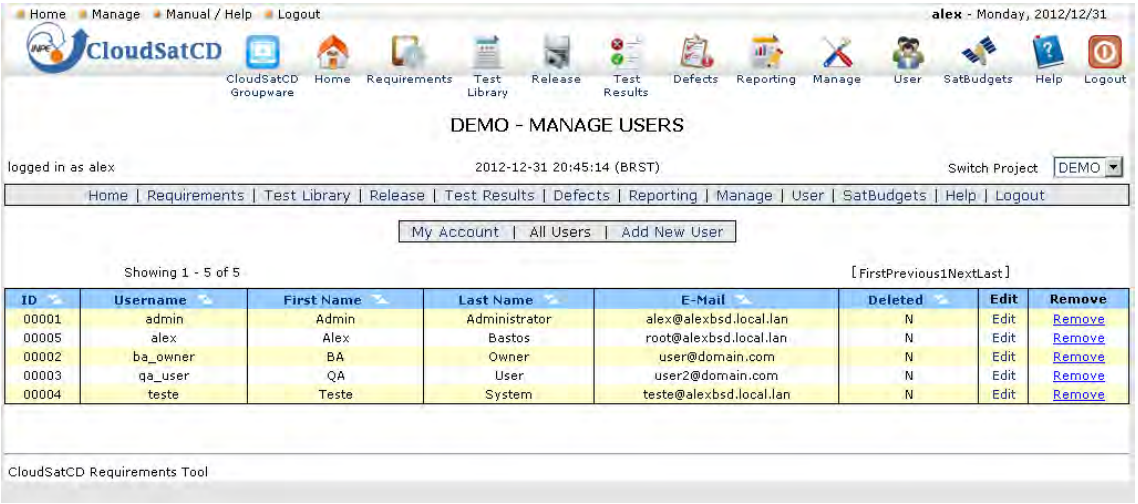


Figura 5.35 - *CloudSatCD Requirements Tool - User*

É possível destacar dentre as principais *features* do módulo *User*: gerenciar usuários cadastrados; adicionar usuários; editar as contas e as permissões de usuários; excluir usuários; alterar as preferências de e-mail de usuários; adicionar usuários a projetos; definir o grupo para usuários; definir um usuário

como administrador do sistema, porém somente outro administrador pode executar essa atribuição; definir os papéis de Analista de Negócios (*BA Owner*) e Controle de Qualidade (*QA Owner*) a usuários; entre outros.

5.2.2.1. Disponibilização de SatBudgets em nuvem

SatBudgets é o modulo para invocação de serviços web de balanços de disciplinas de engenharia de sistemas espaciais em diversas áreas do conhecimento, podendo estar alocados na base de dados ou remotamente em parceiros via SpaceESB.

O processo de modelagem conceitual de satélites necessita que balanços (*budgets*) sejam executados de forma iterativa, a fim de avaliar previamente os aspectos fundamentais de um satélite, tal quais suas restrições, de forma que a viabilidade de determinada arquitetura seja antevista e garantida.

Esse módulo permite que arquiteturas de satélites, cadastradas na base de dados ou inseridas manualmente, sejam avaliadas quanto à viabilidade. Ao inserir uma arquitetura na interface, é possível selecionar os balanços desejados e invocar processamento para que o software extraia os parâmetros do modelo, execute cálculos baseado em regras de negócios pré-definidas e retorne os resultados.

Essa interface de acesso é hospedada e executada sob o servidor de aplicações de CloudSatCD, o Glassfish, e codificada utilizando PHP e Java simultaneamente. A Figura 5.36 exibe a interface, que pode ser invocada através do link *SatBudgets* dentro de CloudSatCD Requirements Tool, onde usuário insere o nome da arquitetura que será avaliada e sua versão (opcional) e faz o *upload* do(s) modelo(s) da arquitetura(s) construído(s) em SysML (XMI) ou escolhe dentre os modelos já existentes, caso esteja(m) salvo(s) na base de dados remota, seleciona opcionalmente configurações adicionais como o modo de operação do satélite, os balanços (*budgets*) desejados, e o formato de saída do relatório desejado.

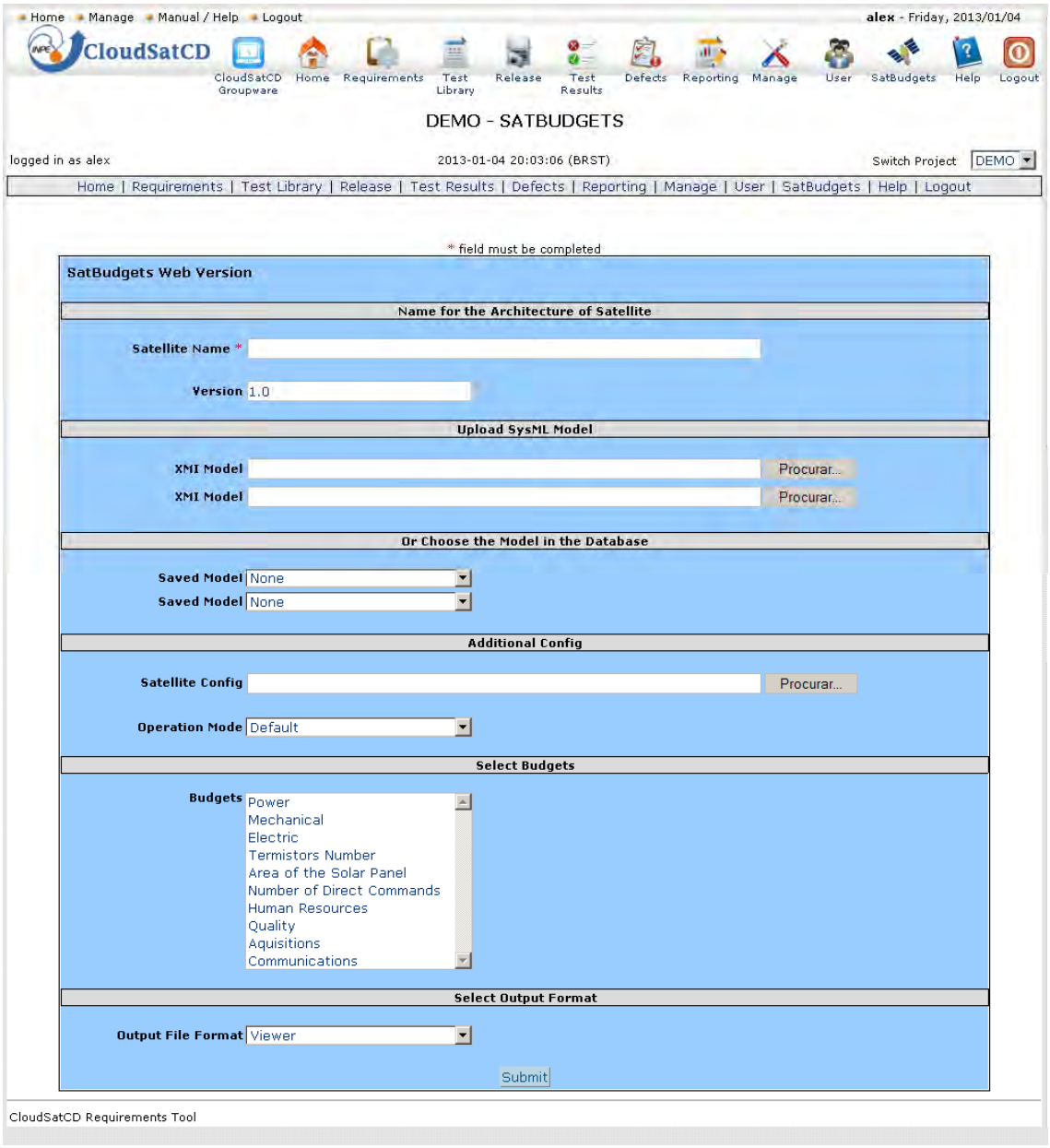
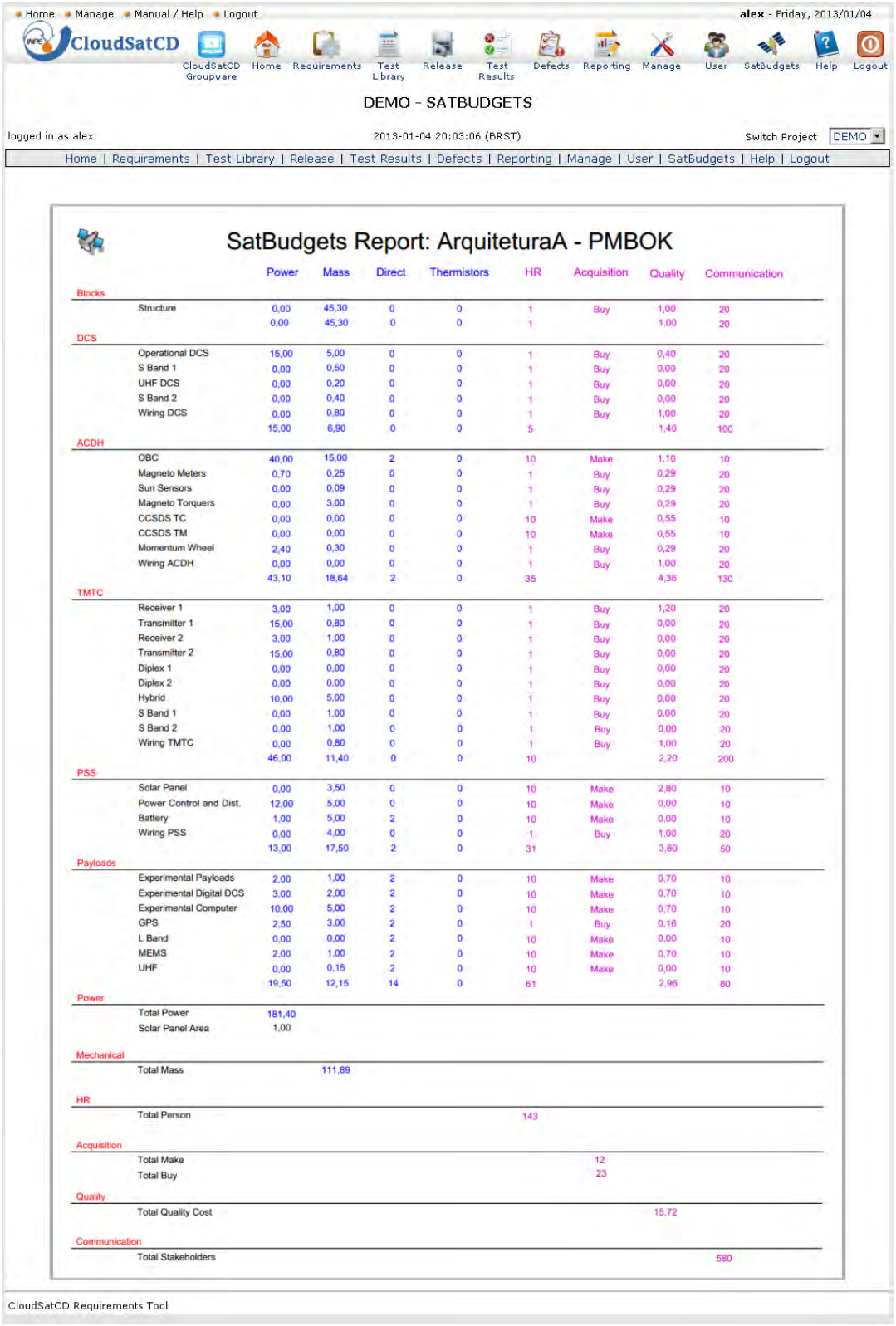


Figura 5.36 - *CloudSatCD Requirements Tool - SatBudgets*

A Figura 5.37 exibe um exemplo com balanços invocados para uma determinada arquitetura inserida, chamada de ArquiteturaA - PMBOK, onde é possível analisar diversos dados estatísticos relacionados através do resultado apresentado.



		Power	Mass	Direct	Thermistors	HR	Acquisition	Quality	Communication
Blocks									
	Structure	0,00	45,30	0	0	1	Buy	1,00	20
		0,00	45,30	0	0	1		1,00	20
DCS									
	Operational DCS	15,00	5,00	0	0	1	Buy	0,40	20
	S Band 1	0,00	0,50	0	0	1	Buy	0,00	20
	UHF DCS	0,00	0,20	0	0	1	Buy	0,00	20
	S Band 2	0,00	0,40	0	0	1	Buy	0,00	20
	Wiring DCS	0,00	0,80	0	0	1	Buy	1,00	20
		15,00	6,90	0	0	5		1,40	100
ACDH									
	OBC	40,00	15,00	2	0	10	Make	1,10	10
	Magneto Meters	0,70	0,25	0	0	1	Buy	0,29	20
	Sun Sensors	0,00	0,09	0	0	1	Buy	0,29	20
	Magneto Torquers	0,00	3,00	0	0	1	Buy	0,29	20
	CCSDS TC	0,00	0,00	0	0	10	Make	0,55	10
	CCSDS TM	0,00	0,00	0	0	10	Make	0,55	10
	Momentum Wheel	2,40	0,30	0	0	1	Buy	0,29	20
	Wiring ACDH	0,00	0,00	0	0	1	Buy	1,00	20
		43,10	18,64	2	0	35		4,36	130
TMTC									
	Receiver 1	3,00	1,00	0	0	1	Buy	1,20	20
	Transmitter 1	15,00	0,80	0	0	1	Buy	0,00	20
	Receiver 2	3,00	1,00	0	0	1	Buy	0,00	20
	Transmitter 2	15,00	0,80	0	0	1	Buy	0,00	20
	Diplex 1	0,00	0,00	0	0	1	Buy	0,00	20
	Diplex 2	0,00	0,00	0	0	1	Buy	0,00	20
	Hybrid	10,00	5,00	0	0	1	Buy	0,00	20
	S Band 1	0,00	1,00	0	0	1	Buy	0,00	20
	S Band 2	0,00	1,00	0	0	1	Buy	0,00	20
	Wiring TMTC	0,00	0,80	0	0	1	Buy	1,00	20
		46,00	11,40	0	0	10		2,20	200
PSS									
	Solar Panel	0,00	3,50	0	0	10	Make	2,80	10
	Power Control and Dist.	12,00	5,00	0	0	10	Make	0,00	10
	Battery	1,00	5,00	2	0	10	Make	0,00	10
	Wiring PSS	0,00	4,00	0	0	1	Buy	1,00	20
		13,00	17,50	2	0	31		3,80	50
Payloads									
	Experimental Payloads	2,00	1,00	2	0	10	Make	0,70	10
	Experimental Digital DCS	3,00	2,00	2	0	10	Make	0,70	10
	Experimental Computer	10,00	5,00	2	0	10	Make	0,70	10
	GPS	2,50	3,00	2	0	1	Buy	0,16	20
	L Band	0,00	0,00	2	0	10	Make	0,00	10
	MEMS	2,00	1,00	2	0	10	Make	0,70	10
	UHF	0,00	0,15	2	0	10	Make	0,00	10
		19,50	12,15	14	0	61		2,96	80
Power									
	Total Power	181,40							
	Solar Panel Area	1,00							
Mechanical									
	Total Mass		111,89						
HR									
	Total Person					143			
Acquisition									
	Total Make						12		
	Total Buy						23		
Quality									
	Total Quality Cost							15,72	
Communication									
	Total Stakeholders								580

Figura 5.37 - *CloudSatCD Requirements Tool - SatBudgets Results*

Pode-se destacar entre as principais funcionalidades do módulo *SatBudgets*: invocação de *web services* de balanços mecânico, elétrico, computação de bordo, número de termistores, área do painel solar, número de comandos diretos, recursos humanos, qualidade, comunicações e aquisições para projetos de satélites; inserção de arquiteturas de satélites na base de dados; apresentação de resultados em gráficos; apresentação de resultados em formato XML; exportação de dados; entre outros.

5.3. Avaliação geral

Concluída a prototipação do ambiente proposto, é possível evidenciar a separação que existe entre a infraestrutura subjacente e as aplicações providas por CloudSatCD.

Enquanto a infraestrutura é totalmente criada sob máquinas físicas e segue o modelo IaaS para criação de máquinas virtuais, as aplicações são criadas em máquinas virtuais e seguem o modelo de provisão de serviços de software.

Dessa forma os usuários têm acesso a uma interface disponível via *browser* com uma variedade de recursos e serviços disponibilizados em nuvem, sem precisar instalar nada localmente ou conhecer a infraestrutura subjacente, formando um *appliance*[11] de *e-Engineering* voltado para o projeto conceitual de satélites, conforme demonstrado de maneira global na Figura 5.38. A partir da arquitetura disponibilizada é possível gerar uma matriz de rastreabilidade para os requisitos inicialmente levantados, conforme demonstrado na Tabela 5.2.

A chave dessa infraestrutura é a facilidade de uso, pois o usuário não necessita fazer nenhuma configuração, ainda que o ambiente seja altamente customizável. A necessidade de treinamento tende a ser baixa, pois o sistema é intuitivo e autoexplicativo, além de contar com manuais dentro do próprio sistema.

[11] Dispositivo, ferramenta ou equipamento desenvolvido e configurado para executar uma função/tarefa específica em um sistema. Normalmente é baseado em um software de uso genérico, porém otimizado para integrar somente componentes necessários à aplicação-alvo.

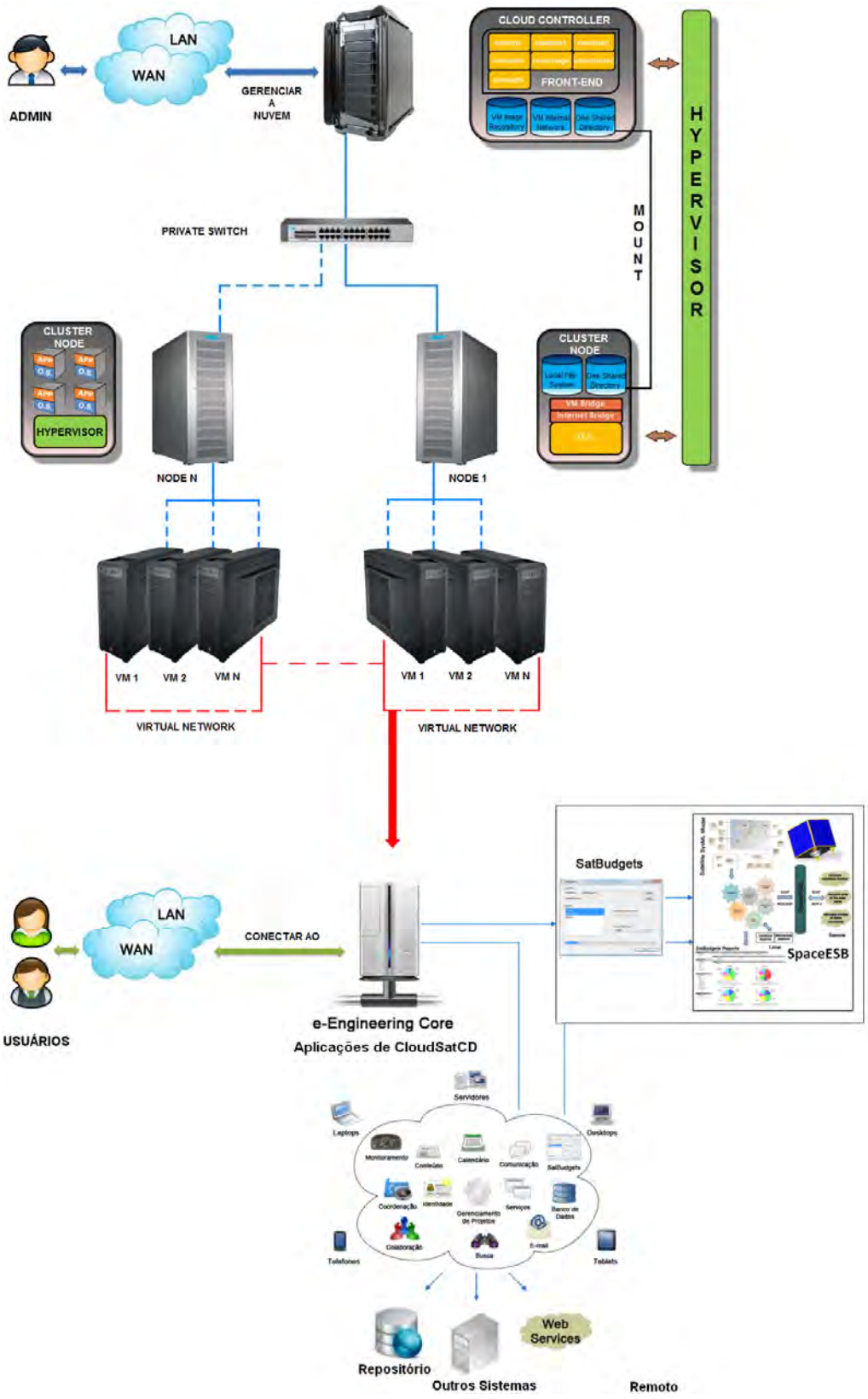


Figura 5.38 - Visão geral da infraestrutura e aplicação de CloudSatCD

Tabela 5.2 - Matriz de rastreabilidade de CloudSatCD

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9	RF10	RF11	RF12	RF13	RF14	RF15	RF16	RF17	RF18	RF19
e-Engineering	X	X	X	X	X														
Nuvem privada	X	X	X	X															
SaaS	X	X	X	X	X														
Groupware	X	X	X																
Admin	X		X																
Email	X		X						X										
Calendar	X		X					X											
Address Book	X		X																
File Manager	X		X										X						
Infolog	X		X											X					
Project Manager	X		X			X													
Time Sheet	X		X					X											
Tracking System	X		X									X					X		
Bookmarks	X		X																
Knowledge Base	X		X				X												
Site Manager	X		X																
Resources	X		X								X								
Wiki	X		X							X									
News Admin	X		X																
Polls	X		X																
Requirements	X		X												X	X			
Test Library	X		X														X		
Release	X		X													X			
Test Results	X		X														X		
Defects	X		X									X					X		
Reporting	X		X																
Manage	X		X			X													
User	X		X																
SatBudgets	X		X															X	
SpaceESB	X																		X

6 CONCLUSÕES

Neste capítulo são apresentadas algumas considerações finais e, adicionalmente, encerra o trabalho com algumas sugestões de trabalhos futuros seguindo a linha de pesquisa adotada.

6.1. Considerações Finais

Dentro do contexto de um projeto de pesquisa, a tecnologia tem um papel facilitador, principalmente por seu apoio à colaboração, compartilhamento de informação e gerenciamento de dados. Nesse sentido, a aceitação dos diversos processos de gerenciamento de projetos por parte dos envolvidos é condição fundamental para o sucesso.

Neste contexto, a Tecnologia da Informação (TI) atual passa por um processo de evolução do uso de pacotes de softwares autônomos servindo a um conjunto bem definido e limitado de usuários para o uso de software como um serviço em rede a um número grande de usuários.

Este trabalho propôs uma infraestrutura, denominada CloudSatCD, baseado em um ambiente SaaS para integração de *groupware* e gerenciador de requisitos a dois sistemas legados, SatBudgets e SpaceESB, formando um ecossistema de *e-Engineering*, focado inicialmente no projeto conceitual de satélites, uma importante fase inicial no ciclo de vida para projetos de engenharia espacial.

A abordagem apresentada possibilita que participantes de projetos de sistemas espaciais possam interagir ativamente na concepção de um projeto espacial, colaborando, comunicando, coordenando, aumentando a agilidade em decisões e flexibilidade.

A solução, baseada em um sistema de cinco camadas, esconde a complexidade da nuvem criando a abstração de *e-Engineering*, permitindo aos usuários interagir de forma transparente com um portal *mashup* web na

máquina do cliente. Posteriormente, pode se comunicar através da rede com as outras camadas que podem ser residentes em outros sistemas federados, formando um ambiente web colaborativo que pode consumir e prover informação e serviços. Almeja-se com isto lidar melhor com alguns desafios e aumento da produtividade técnica em Engenharia de Sistemas Espaciais.

A forma ubíqua de distribuir informações para uma grande quantidade de dispositivos e plataformas de software através dos serviços de rede constitui um paradigma na computação distribuída. Através computação em nuvem aliada a ferramentas *open source* e padrões abertos pode-se aumentar a flexibilidade e reduzir custos, através da construção de uma nuvem privada.

CloudSatCD propicia contribuir no processo de tomada de decisão, colaboração, comunicação e coordenação, além de auxiliar na escolha de opções de arquiteturas candidatas ao projeto, objeto principal das técnicas de DSE. Tal integração promove informações e conhecimento, que podem ser disponibilizados e/ou compartilhados. Isto faz com que procedimentos de especialistas em suas respectivas áreas de conhecimento deixem de ser muitas vezes trabalhados individualmente.

Como CloudSatCD é uma solução altamente customizável, escalável e com suporte multiplataforma, novos recursos e serviços podem ser agregados a qualquer tempo, evoluindo à medida que a demanda por tecnologia computacional e outros serviços aumenta.

Adicionalmente espera-se que o ambiente possa: incentivar o desenvolvimento da engenharia de sistemas através de uma infraestrutura adequada que atenda a fase inicial de projetos de satélite; facilitar a geração de interfaces web adaptadas para os projetos de satélites, assim como a interação entre os engenheiros de sistemas e parceiros do projeto através da web; promover a reutilização e compartilhamento de dados de projeto; facilitar o gerenciamento de projetos de pesquisa.

6.2. Trabalhos Futuros

Por se tratar de uma nova abordagem, com tecnologias e conceitos recentes, ainda não totalmente explorados e maturados, há diversas possibilidades para extensão e expansão de CloudSatCD, onde algumas são apresentadas nos tópicos a seguir:

- Implantação do sistema nas áreas de engenharia de sistemas do INPE;
- Adição de serviços de mensagem instantânea e videoconferência;
- Inserção de novos serviços de software visando apoiar outras fases do projeto de satélites;
- Inserção de ferramentas para modelagem SysML como serviço integrado;
- Inserção de aplicativos para edição de planilhas, edição de texto e edição de slides;
- Inserção de uma ferramenta para gerenciar fluxo de trabalho (*workflow*);
- Integração da atual solução com outros sistemas legados;
- Integração com outros serviços web;
- Expansão da rede física, potencializando o poder de processamento da mesma e aumentando o número de máquinas virtuais;
- Armazenamento dos dados em redes de dados dedicadas.

REFERÊNCIAS BIBLIOGRÁFICAS

ADVANCED MICRO DEVICES, INC. (AMD). **AMD cloud computing**: AMD creates a private cloud to increase its own productivity, improve accessibility, lower costs and maintain its position as industry innovator. 2011, Disponível em: <http://blogs.amd.com/stories/files/2012/06/CRP-May-2011-Cloud-Case-Study-FINAL1.pdf>. Acesso em: jul. de 2012.

ALMEIDA, F. J. de. Estudo e escolha de metodologia para o projeto conceitual. **Revista de Ciência & Tecnologia**, v. 8, nº 16, p. 31-42, 2000.

APACHE. **Apache ServiceMix**: is ServiceMix the right ESB for me?, 2012. Disponível em: <http://servicemix.apache.org/is-servicemix-the-right-esb-for-me.html>. Acesso em: mar de 2012.

APACHE HTTP SERVER. **The apache HTTP server project**. Disponível em: <http://httpd.apache.org/>. Acesso em: mar. de 2012.

APACHE MAVEN. **Apache Maven project**. Disponível em: <http://maven.apache.org/>. Acesso em: jul. de 2012.

APACHE SERVICEMIX **Apache ServiceMix**. Disponível em: <http://servicemix.apache.org/>. Acesso em: mar. de 2012.

BANDECCHI M.; MELTON B.; ONGARO F. Concurrent engineering applied to space mission assessment and design. **ESA bulletin**, n.99, 1999, Noordwijk, Holanda. Disponível em: <http://www.esa.int/esapub/bulletin/bullet99/bande99.pdf>. Acesso em: fev. de 2012.

BOOCH, G.; BROWN, A.W. Collaborative development environments. **Advances in Computers**, v. 59, p. 1- 27, 2003. http://dx.doi.org/10.1016/S0065-2458(03)59001-5.

BOTELHO, E. X.; VIDAL, J. M. B. CSCW-trabalho cooperativo suportado por computador. **Cefet-RN, holos** -, Ano 21, v. 1, p. 130-137, maio 2005. ISSN 1807-1600.

CAUCHO RESIN. **Caucho resin**: reliable, open source application server. Disponível em: <http://quercus.caucho.com/>. Acesso em: jul. de 2012.

CHAPPEL, D. SOAP vs. REST: complements or competitors?. In: ESRI INTERNATIONAL USER CONFERENCE, 2009, San Diego, CA. **Proceedings…**San Diego, CA: Esri, 2009.

CLAMAV. **Clam AntiVirus**. Disponível em: <http://www.clamav.net>. Acesso em: jul. de 2012.

COLLABORATIVE WORK ENVIRONMENT (CWE),. **The CCSDS Collaborative Work Environment**. Disponível em <http://cwe.ccsds.org/default.aspx>. Acesso em: nov. de 2012.

COURIER-IMAP. **The courier imap server**. Disponível em: <http://www.courier-mta.org/imap/>. Acesso em: jul. de 2012.

CUCO, A. P. C.; SOUSA, F. L.; NETO, A. J. S. Uma metodologia para a geração de layout no projeto conceitual de satélites artificiais usando técnicas evolutivas de otimização multiobjetivo. In: WORKSHOP EM ENGENHARIA E TECNOLOGIA ESPACIAIS, 1. (WETE), 2010, São José dos Campos. **Anais...** São José dos Campos: INPE, 2010. v. IWETE2010-1062. DVD. ISSN 2177-3114. Disponível em: <http://urlib.net/8JMKD3MGP7W/399ES4B>. Acesso em: 14 maio 2013..

CULLER, D. E.; GARCÍA, J. A. P. Using internet based concurrent engineering tools to educate multinational students about the design, process planning and manufacture of new products. In: ASEE/IEEE FRONTIERS IN EDUCATION CONFERENCE, 34., Savannah, GA. **Proceedings…** , Savannah, GA: IEEE, 2004.

DE-KONING, H-P.; FAVARO, J.; MAZZINI, S.; SCHREINER, R.; OLIVE, X. Next generation requirements engineering. In: ANNUAL INCOSE INTERNATIONAL SYMPOSIUM (IS 2012), 22., 2012, 9 to 12 July, Rome, Italy. **Proceedings…** Rome, 2012.

DOS-SANTOS, W.A. LEONOR, B.B.F.; STEPHANY, S. A knowledge-based approach to deal with conceptual satellite design. **Lecture Notes in Computer Science** - Conceptual Modeling - ER 2009, v. 1, p. 487–500, 2009.

E-SCIENCE DIGITAL REPOSITORIES (E-SCIDR). **Towards a european e-infrastructure for e-Science digital repositories**: a report for the European Commission**.** The Digital Archiving Consultancy Limited (ed). Technical Report 2006 S88-092641. Middlesex, UK, Final report, 2008. Disponível em <http://www.e-scidr.eu/wp-content/uploads/2007/03/e-SciDR_DAC_Final_Report.pdf>. Acesso em: jul. de 2011.

EGROUPWARE, **EGroupware home**. Disponível em: <http://www.egroupware.org/>. Acesso em: mar. de 2012.

EGROUPWARE. **Feature list**: egroupware enterprise line - egroupware community edition. Stylite Computing Center, 2011. Disponível em:

<http://www.egroupware.org/uploads/documents/epl-features/Features-and-advantages-EPL-vs-CE.pdf>. Acesso em: out. 2012.

ERL, T. **Service oriented architecture**: a field guide to integrating xml and web services. Upper Saddle River, New Jersey: The Prentice Hall, 2004.

ERL, T. **SOA design patterns**. 1. ed. Upper Saddle River, New Jersey: The Prentice Hall, 2008. 856 p.

ESERYEL, D.; GANESAN, R.; EDMONDS, G. S. Review of computer-supported collaborative work systems. **Educational Technology & Society**, v. 5, n.2, 2002. Syracuse University.

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION. **Space engineering** – engineering design model data exchange (CDF). Noordwijk, Netherlands: ECSS Secretariat, ESA-ESTEC Requirements & Standards Division, 20 oct. 2010. (ECSS E-TM-10-25A).

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION. **Space engineering** – structural general requirements. Noordwijk, Netherlands: ECSS Secretariat, ESA-ESTEC Requirements & Standards Division, rev. 1, 15 nov. 2008. (ECSS E-ST-32C).

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION. **Space engineering** – system engineering general requirements. Noordwijk, Netherlands: ECSS Secretariat, ESA-ESTEC Requirements & Standards Division, 6 mar. 2009. (ECSS E-ST-10C).

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION. **Space engineering** – technical requirements specification. Noordwijk, Netherlands: ECSS Secretariat, ESA-ESTEC Requirements & Standards Division, 6 oct. 2009. (ECSS E-ST-10-06C).

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION. **Space project management** – project planning and implementation. Noordwijk, Netherlands: ECSS Secretariat, ESA-ESTEC Requirements & Standards Division, rev. 1, 6 mar. 2009. (ECSS M-ST-10C).

EYNARD, B.; LIÉNARD, S.; CHARLES, S.; ODINOT, A. Web-based collaborative engineering support system: applications in mechanical design and structural analysis. **Concurrent Engineering**, v. 13, p.145-153, 2005. DOI: 10.1177/1063293X05053799.

FIELDING, R. T. **Architectural styles and the design of network-based software architectures**. Dissertation (Information and Computer Science) - University of California, Irvine, 2000.

FREEBSD. **The FreeBSD project**. Disponível em: <http://www.freebsd.org/>. Acesso em: mar. de 2012.

FRIEDENTHAL, S.; MOORE, A.; STEINER, R. **OMG systems modeling language (OMG SysML) tutorial**, Systems Engineering for the Planet, INCOSE, 2008.

FUJITSU. **Fujitsu announces "engineering cloud" for new era in manufacturing**. Disponível em: <http://www.fujitsu.com/global/news/pr/archives/month/2011/20110621-02.html>. Acesso em: jun. de 2012.

FUKS, H.; RAPOSO, A. B.; GEROSA, M.A. Do modelo de colaboração 3C à engenharia de groupware. In: WEBMIDIA 2003 - SIMPÓSIO BRASILEIRO DE SISTEMAS MULTIMÍDIA E WEB, TRILHA ESPECIAL DE TRABALHO COOPERATIVO ASSISTIDO POR COMPUTADOR, 2003, Salvador, BA. **Anais eletrônicos...** Salvador-BA, 2003. p. 445-452.

GIORGIO, F. di; PAPARO, V. BASSO, V. ROSSER, X. Applying collaborative system engineering in thales alenia space: lessons learned and best practices. In: INTERNATIONAL WORKSHOP ON SYSTEMS & CONCURRENT ENGINEERING FOR SPACE APPLICATIONS - SECESA 2012, 5., 2012, Lisbon. **Proceedings…**17-19 Oct., Lisbon, Portugal, 2012.

GLASSFISH. **Glassfish** - the open source application server - java.net. Disponível em: <http://glassfish.java.net/>. Acesso: em mar. de 2012.

GROSSI, B. E. **Estudo do modelo de computação orientada a serviços e sua aplicação a um sistema de mineração de dados**. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Minas Gerais - UFMG. 2005. Disponível em: <http://www.bibliotecadigital.ufmg.br/dspace/bitstream/1843/RVMR-6EAFWY/1/brunoestolanogrossi.pdf>. Acesso em: out. de 2011.

GRUDIN, J. Computer-supported cooperative work: history and focus. **Computer**, v. 27, n.5, p. 19–26, 1994.

GRUDIN, J. Why CSCW applications fail: problems in the design and evaluation of organization of organizational interfaces. In: ACM CONFERENCE ON COMPUTER-SUPPORTED COOPERATIVE WORK, 1988, New York. **Proceedings...** New York: ACM Press, 1988. p. 85–93.

GUERRA, C. **Definição e desenvolvimento de ferramentas colaborativas para a fase de projeto conceitual de produto**. 2007. 184p. Dissertação (Mestrado em Engenharia Mecânica) - Universidade Federal de Santa Catarina, Florianópolis, 2007.

GUIA LIVRE. **Referência de migração para software livre do governo federal**. Brasília, 2005. 297 p. Organizado por Grupo de Trabalho Migração para Software Livre.

HAMBLOCH, P.; DE-PASCALE, F.; KRUIJFF M. ALBATROS - a space system engineering tool. In: INTERNATIONAL ASTRONAUTICAL CONGRESS, 2007, Hyderabad. **Proceedings…** Hyderabad, 2007. (IAC-07-D5.1.05).

HANSEN, M. D. **SOA using java web services**. 1. ed. Upper Saddle River: Prentice Hall, 2007. 608 p.

HEGEDÜS, Á.; HORVÁTH, Á.; RÁTH, I.; VARRÓ, D. **A Model-driven framework for guided design space exploration**. Budapest, Hungary: Budapest University of Technology and Economics, 2010.

HEY, T.; TREFETHEN, A. e-Science and its implications. **Philosophical Transactions of the Royal Society A**. v. 361, p.1809–1825, 2003.

HURWITZ, J.; BLOOR, R.; KAUFMAN, M.; HALPER, F. **Arquitetura Orientada a Serviços** – SOA para leigos. 2. ed. Rio de Janeiro: Alta Books. 2009.

IMETI, **Summer International Symposium on Collaborative Engineering and Science: isCES 2012**. The 5th International Multi-Conference on Engineering and Technological Innovation: IMETI 2012. July 17th - 20th, 2012 – Orlando, Florida, USA. Disponível em: <http://www.iiis2012.org/imeti/website/AboutConfer-isCES.asp?vc=20>. Acesso em: out. de 2012.

JANSEN, W.; GRANCE, T. **Guidelines on security and privacy in public cloud comnputing**. Gaithersburg, MD: NIST - National Institute of Standards and Technology, 2011. Special Publication 800-144.

JAVA. **Java + You**. Disponível em: <http://www.java.com>. Acesso em: mar. de 2012.

JOSUTTIS, N. M. **SOA na prática**: a arte da modelagem de sistemas distribuídos. Rio de Janeiro: Alta Books. 2008.

KEPES, B. **Understanding the cloud computing stack**: PaaS, SaaS, IaaS. CloudU White Paper. 2011.

KOOPMANS, M. **Refining the cloud stack**, A whitepaper on first lessons learned in the NEON project, 2010.

KRZEMINSKI, K.; JOZWIAK, I. Synergic intranet: an example of synergic it as the goal of e-engineering. In: HIPPE, Z.S. et al. (eds.): **Human -** computer systems interaction. Berlin: Spring Verlag, 2012. Part I, p. 359–370 (AISC 98).

KUK, S.H.; KIM, H.S.; LEE, J.K.; HAN, S.H.; PARK, S.W. An e-engineering framework based on service-oriented architecture and agent technologies. **Computers in Industry**, v.59, n.9, Elsevier, p.923-935, 2008.

KULESZA, R; ALVES, L. G. P.; ALMEIDA, F. L.; JARDINI, M. F.; VASCONCELOS, M. A. V. M.; FILHO, G. L. S. Giga-Colab: uma arquitetura de ambiente virtual colaborativo com suporte a serviços multimídia. In: Simpósio Brasileiro de Sistema Colaborativos, 4., 2007, Rio de Janeiro. **Anais...** Porto Alegre : Sociedade Brasileira de Computação, 2007. v. 1. p. 111-124.

KVM. **Kernel-based virtual machine**. Disponível em: <http://www.linux-kvm.org>. Acesso em: ago. de 2012.

LARSON, W. J.; WERTZ, J. R. **Space mission analysis and design**. 3 ed. California, 1999.

LEE, J.; PARK, S.; CHA, M.; KUK, S. H.; KIM, H. S. **Service composition system in consideration of the characteristics of engineering services. International Journal of Software Engineering and Its Applications**, v. 4, n. 1, Jan. 2010.

LEONOR, B. B. F. **Um enfoque baseado em conhecimento e dirigido a modelos de engenharia de requisitos para projeto conceitual de satélites**. 2010. 111 p., Dissertação (Mestrado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2010.

LIMA, M. F.; SICSÚ, A. B.; CABRAL, A. P. **Sistemas de workflow e groupware na gestão do conhecimento como diferencial competitivo**., Recife-PE: V IntEmpres, 2004.

LUCAS, M. **Dominando BSD**: o guia definitivo para o FreeBSD. Rio de Janeiro: Ed. Ciência Moderna, 2003. 585p.

MACEDO, M. O. **Especificação e desenvolvimento de uma ferramenta colaborativa de suporte ao desenvolvimento de projectos de reorganização e de implementação de sistemas de informação**. Dissertação (Mestrado Integrado em Engenharia Electrotécnica e de Computadores Major Automação) - Faculdade de Engenharia da Universidade do Porto, Portugal, 126 p., 2008.

MANTOVANI, C. M. C. A; MOURA, M. A. **Pesquisa científica em rede**: novas mediações, práticas discursivas e atores sociais. In: ACTAS DEL FORO

IBEROAMERICANO DE COMUNICACIÓN Y DIVULGACÍON CIENTÍFICA. Campinas, 2009. Disponível em: <http://www.oei.es/forocampinas/PDF_ACTAS/COMUNICACIONES/grupo2/185.pdf>

MASS.GOV. **Massachusetts science and technology/engineering curriculum framework**. Massachusetts Department of Elementary & Secondary Education, 2001. Disponível em: <http://www.doe.mass.edu/frameworks/scitech/2001/standards/strand4.html>. Acesso em: out. de 2012.

MEDEIROS, M.; BASTOS, A. B.; BARBOSA, C. A. M.; SIMÕES, M. C.; DOS SANTOS, W. A. Exploração do Espaço de Projeto de Satélites para um Cenário Reduzido de Análise. In: 1º SIMPÓSIO AEROESPACIAL BRASILEIRO (SAB 2012), 2012, São José dos Campos-SP. **Anais...** 2012. DVD.

MEDEIROS, V. P. de. Construção de sistemas web utilizando ferramentas livres. **Anuário da Produção de Iniciação Científica Discente**, v. 12, n. 14, Ano 2009. Faculdade Anhanguera de Taubaté, Publicação 2010. Disponível em: <http://sare.unianhanguera.edu.br/index.php/anuic/article/viewFile/1671/811>. Acesso em: out. de 2011.

MEJÍA, R.; MOLINA, A.; AUGENBROE, G. **Collaborative planning of a manufacturing design project through a novel e-engineering hub**. Campus Monterrey, México: European Commission and the Chair in Mechatronics from the ITESM, 2005. e-HUBs consortium (IST-2001-34031).

MILLS, K. L. Computer-supported cooperative work. In: **Encyclopedia of library and information science**. 2.ed, New York: Marcel Dekker, Inc., 2003. DOI:10.1081/E-ELIS 120008706, 666-677.

MOSER, T.; MORDINYI, R.; WINKLER, D.; BIFFL, S. Engineering project management using the engineering cockpit - a collaboration platform for project managers and engineers.In: IEEE International Conference, 9., 26-29 July, 2011. Caparica, Lisbon. **Proceedings…** Caparica, Lisbon: IEEE, 2011. P. 579-584 . Industrial Informatics (INDIN).

MYSQL. **MySQL**: the world's most popular open source database. Disponível em: <http://www.mysql.com/>. Acesso em: mar. de 2012.

NIEDERÉE, C; RISSE, T; PAUKERT, M.; STEIN, A. **An architecture blueprint for knowlege-based e-science**. German e-Science, 2007. Disponível em: <http://www.ipsi.fraunhofer.de/~stein/publications/NiedereeRissePaukertStein_GES07.pdf>. Acesso em: jul de 2011.

Open Concurrent Design Server (OCDS). **Open concurrent design server community portal**. Disponível em <http://atlas.estec.esa.int/uci_wiki/tiki-index.php>. Acesso em: nov. de 2012.

OPENNEBULA. **The open source solution for data center virtualization**. Disponível em: <http://opennebula.org/>. Acesso em: ago. de 2012.

OPENVPN. **Open source VPN**. Disponível em: <http://openvpn.net/>. Acesso em: nov. de 2012.

PAPAZOGLOU, M. P.; TRAVERSO, P.; DUSTDAR, S.; LEYMANN, F. **Service-oriented computing**: state of the art and research challenges. IEEE Computer Society. November, 2007. Disponível em: <http://citeseer.ist.psu.edu/viewdoc/download?doi=10.1.1.139.1724&rep=rep1&type=pdf>. Acesso em: out. de 2011.

PFEIFER. **The domain and goals of CSCW**. University of Calgary, 1995. Disponível em: <http://ksi.cpsc.ucalgary.ca/courses/547-95/pfeifer/cscw_domain.html>. Acesso em: abr. de 2012.

PHP. **PHP**: hypertext processor. Disponível em: <http://www.php.net/>. Acesso em: mar. de 2012.

PHPMYADMIN. **phpMyAdmin home**. Disponível em: <http://www.phpmyadmin.net/>. Acesso em: out. de 2012.

PIZETTE, L.; RAINES, G. **Products to build a private cloud**. Systems Engineering at MITRE, Cloud Computing Series, 2010.

PMBOK. **Um guia do conhecimentos em gerenciamento de projetos (Guia PMBOK)**. 4. ed. Project Management Institute, Inc. - PMI, 2008.

POPTOP. **The PPTP server for Linux**. Disponível em: <http://poptop.sourceforge.net/>. Acesso em: nov. de 2012.

POSTFIX. **The Postfix home page**. Disponível em: <http://www.postfix.org/>. Acesso em: jul. de 2012.

RAMA, J.; BISHOP, J. **Survey and comparison of CSCW groupware applications**. University of Pretoria, South Africa. 2006.

RATIONAL SURVIVABILIT. **The vagaries of cloudcabulary**: why public, private, internal & external definitions don't work... 2009. Disponível em: <http://www.rationalsurvivability.com/blog/2009/04/the-vagaries-of-cloudcabulary-why-public-private-internal-external-definitions-dont-work/>. Acesso em: abr. de 2012.

REDONDO, H. S.; CALLEJA, P. D.; GONZÁLES, D. R. **Implementación de un controlador de VirtualBox para OpenNebula**. Madrid: Facultad de Informática, Universidad Complutense de Madrid, Espanha, 2011.

ROSS, J. W.; WEILL, P.; ROBERTSON, D. **Enterprise architecture as strategy: creating a foundation for business execution**. Cambridge: Harvard Business School Press, 2006.

RTH BLOG. **RTH - requirements and testing hub**. Disponível em: <http://requirementsandtestinghub.wordpress.com/>. Acesso em: out. de 2012.

SCARFONE, K.; SOUPPAYA, M.; HOFFMAN, P. **Guide to security for full virtualization technologies**. Gaithersburg, MD: NIST - National Institute of Standards and Technology, 2011. Special Publication 800-125.

SHEN, W. Editorial of the special issue on knowledge sharing in collaborative design environments. **Computer in Industry**, v.52, n.1, p. 1-3, 2003.

SILVA, L. M. da; BRAGA, R.; CAMPOS, F. Um Framework para composição semântica de workflows científicos. **VETOR - Revista de Ciências Exatas e Engenharias**, v. 19, n. 1, 2009.

SMITH, P. L.; DAWDY, A. B.; TRAFTON, T. W.; NOVAK, R. G. Concurrent design at aerospace. **Crosslink:** The aerospace corporation magazine of advances in aerospace technology. v.2, n. 1, 2001. Disponível em: <http://www.aero.org/publications/crosslink/winter2001/01.html>. Acesso em: abr. de 2012.

SOUSA, F.R.C.; MOREIRA, L.O.; MACHADO, J.C. **Computação em nuvem**: conceitos, tecnologias, aplicações e desafios. In: ERCEMAPI, 2009, Parnaíba. **Anais...** Universidade Federal do Ceará – UFC, 2009.

SOUZA, A.C.C.; DOS-SANTOS, W.A. **Automating services for spacecraft concept design via an enterprise service bus**. In: ISPE CONCURRENT ENGINEERING - CE2011, 2011, Massachusetts. **Proceedings...** Massachusetts Institute of Technology – MIT, USA, Jul., 2011.

SOUZA, A.C.C. **SpaceESB**: um ambiente colaborativo para apoio ao projeto conceitual de satélites usando barramento corporativo de serviços. 2011. 105 p. (sid.inpe.br/mtc-m19/2011/12.06.19.01-TDI). Dissertação (Mestrado em Engenharia e Gerenciamento de Sistemas Espaciais) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2011. Disponível em: <http://urlib.net/8JMKD3MGP7W/3ATRJUH>. Acesso em: fev de 2012.

SOUZA, C. R. B. **Groupware & CSCW**: conceitos básicos. Departamento de Informática da Universidade Federal do Pará – UFPA, 2011. Disponível em:

<http://www.ufpa.br/cdesouza/teaching/cscw-2006-2/1-introduction.pdf>. Acesso em: fev. de 2012.

SOUZA, P. N. de. **Curso Introdutório em gerenciamento de projetos espaciais**. São José dos Campos: INPE, 2011. (INPE-9605-PUD/126).

SPAMASSASSIN. **The Apache SpamAssassin project**. Disponível em: <http://spamassassin.apache.org/>. Acesso em: jul. de 2012.

SUN MICROSYSTEMS, INC (SUN). **Introduction to cloud computing architecture**. White Paper. 1.ed. 2009.

SZEPIELAK, D. REST-based service oriented architecture for dynamically integrated information systems. In: IBM PhD STUDENT SYMPOSIUM AT ICSOC 2006, 2006, Chicago, IL, USA. **Proceedings…** Chicago, IBM, 2006.

TAURION, C. **Cloud computing** - computação em nuvem: transformando o mundo da tecnologia da informação. Ed. Brasport, 2009. 228p.

TAYLOR, J. **What is e-Science?** The London e-Science Centre, 1999. Disponível em: <http://www.lesc.ic.ac.uk/admin/escience.html>. Acesso em: fev. de 2012.

TEIXEIRA, M. A. M. **Introdução a web services**. Departamento de Informática da Universidade Federal do Maranhão - DEINF/UFMA. 2011. Disponível em: <http://www.deinf.ufma.br/~mario/pos/soa/WS.pdf>. Acesso em: out. de 2011.

TORALDO, G. **OpenNebula 3 cloud computing**. London, UK: Packt Publishing, 2012. 315 p.

UBUNTU. **Ubuntu home**. Disponível em: <http://www.ubuntu.com/>. Acesso em: ago. de 2012.

VELTE, A.T.; VELTE, T.J.; ELSENPETER, R. **Cloud computing**: computação em nuvem – uma abordagem prática. Rio de Janeiro: Ed. Alta Books, 2011. 352 p.

VERAS, M. **Cloud computing:** nova arquitetura de TI. Rio de Janeiro: Brasport, 2012. 240p.

VIRTUALBOX. **Oracle VM Virtualbox**. Disponível em: <https://www.virtualbox.org/>. Acesso em: mar. de 2012.

WEBBER, J.; PARASTATIDIS, S.; ROBINSON, I. **REST in practice**: hypermedia and systems architecture. CA, USA: Ed. O' Reilly, 2010. 448 p.

YASUDA, M.. Fujitsu’s engineering cloud. **FUJITSU Sci. Tech. J.**, v. 48, n.. 4, p. 404-412, Oct. 2012.

YOSHIDA Y.; FUJITA, Y. SaaS and PaaS of engineering cloud**. FUJITSU Sci. Tech. J.**, v. 48, n.. 4, p. 428-433, Oct., 2012.

APÊNDICE A - FERRAMENTAS PARA PROTOTIPAÇÃO

Para a prototipação do ambiente descrito foram utilizadas as seguintes ferramentas, priorizando o tipo *open source* por ser possível alterá-las e adaptá-las conforme necessidade. A seguir cada ferramenta é descrita sumariamente.

- Sistemas Operacionais: FreeBSD e Ubuntu.
- Infraestrutura de Nuvem: OpenNebula.
- Banco de Dados: MySQL.
- Servidor Web: Apache HTTP Server.
- Administração web do Banco de Dados: phpMyAdmin.
- Servidor de Aplicações: Glassfish.
- Plataformas de desenvolvimento: PHP e Java.
- Classe PHP dentro do JAVA: Quercus.
- *Groupware*: eGroupware.
- Gerenciador de Requisitos: RTH.
- Plataforma *runtime*/ESB: Apache ServiceMix.
- *Hypervisores* (virtualização): VirtualBox e KVM.
- VPN: OpenVPN e PopTop.

FreeBSD é um sistema operacional *open source* baseado no UNIX, descendente da família BSD (*Berkeley Software Distribution*), desenvolvido pela Universidade de Berkeley. Por ser robusto, estável e flexível, é comumente usado em servidores (FREEBSD, 2012). Entre outras de suas

principais virtudes estão: portabilidade, eficiência e administração de software simplificada (LUCAS, 2003).

Ubuntu é um sistema operacional Linux construído a partir do Debian e desenvolvido pela comunidade visando segurança. Esse sistema proporciona um ambiente completo e funcional, onde programas adicionais são disponibilizados através da Internet. Está disponível nas versões Desktop e Server, sendo que esta última tem todo o necessário para ter um servidor operacional. Entre suas principais características destacam-se a rapidez e a facilidade de uso, além de ser livre (UBUNTU, 2012).

OpenNebula é um software de código aberto para virtualização de *datacenter* que oferece uma série de recursos em uma solução flexível para o gerenciamento abrangente e completo de centros de dados virtualizados, possibilitando a criação de nuvens IaaS em infraestruturas existentes (OPENNEBULA, 2012). A principal diferença entre OpenNebula e outras soluções do gênero é garantir aos usuários interoperabilidade completa com todos os componentes da infraestrutura existente já disponível. Ao contrário de outras alternativas de código aberto, OpenNebula não está restrito a um determinado *hypervisor*. Também não tem quaisquer requisitos específicos de infraestrutura, encaixando bem em qualquer ambiente pré-existente, armazenamento, rede, gerenciamento de usuário ou políticas. O modelo *plugin*[12] em que OpenNebula é implementado dá a integradores de sistemas a capacidade de personalizar todos os aspectos, incluindo virtualização, armazenamento de informações, autenticação, autorização e serviços remotos em nuvem. Toda ação é gerida por um *script* que pode ser facilmente modificado ou conectado com algum outro *script* personalizado ou software escrito em qualquer idioma e suportado pelo sistema operacional. Sua grande modularidade facilita a integração com outras soluções e o torna um dos mais

[12] *Plugin* é programa ou *script* utilizado para adicionar funcionalidades a outros programas estendendo a capacidade destes. Normalmente são instalados sob demanda.

avançados e altamente escaláveis kits de ferramentas de computação em nuvem (TORALDO, 2012).

MySQL é banco de dados *open source* mais popular do mundo devido ao seu alto desempenho, alta confiabilidade e facilidade de uso (MYSQL, 2012). Por ser um servidor SQL (*Structured Query Language*) leve, é interessante especialmente para aplicações web e similares (GUIA LIVRE, 2005). Para a estrutura proposta neste trabalho, MySQL será o repositório responsável por cuidar da camada de dados: *Data Layer*.

O Apache HTTP Server é um servidor web *open source* para os sistemas operacionais modernos, cujo objetivo é fornecer um servidor seguro, eficiente e extensível, sendo o mais popular e usado na Internet (APACHE HTTP SERVER, 2012). Uma combinação de produtos muito popular figura sob o nome LAMP: Linux, Apache, MySQL e PHP, fornecendo estrutura para os sites acessarem bancos de dados SQL através da linguagem PHP, onde todos os componentes são softwares livres. O Apache é servidor modular, com um motor de protocolo nuclear e uma grande seleção de módulos para propósitos específicos (GUIA LIVRE, 2005).

O phpMyAdmin é uma ferramenta de software livre escrita em PHP destinada a lidar com a administração do MySQL via rede. Sua interface web possui suporte a uma ampla gama de operações com o MySQL como gerenciamento de bancos de dados, tabelas, campos, relações, índices, usuários, permissões, entre outros, além de permitir executar diretamente qualquer declaração SQL (PHPMYADMIN, 2012).

GlassFish é um servidor de aplicações de produção da plataforma padrão *Java Enterprise Edition* 6 (Java EE 6). O projeto GlassFish *Open Source* fornece um processo estruturado para o desenvolvimento da plataforma de servidor que faz com que os novos recursos da plataforma Java EE sejam disponibilizados mais rapidamente, mantendo a característica mais importante do Java EE: compatibilidade (GLASSFISH, 2012).

PHP é uma linguagem de *script* amplamente utilizada especialmente adequada para o desenvolvimento web e pode ser incorporada ao HTML (*HyperText Markup Language*) na criação de páginas dinâmicas (PHP, 2012). Java é uma linguagem de programação e uma plataforma de computação. É a tecnologia que capacita muitos programas da mais alta qualidade, como utilitários, jogos e aplicativos corporativos, entre muitos outros, por exemplo. Há muitos aplicativos e sites que funcionam somente com o Java instalado, e muitos outros aplicativos e sites são desenvolvidos e disponibilizados com o suporte dessa tecnologia todos os dias, por ser rápido, seguro e confiável (JAVA, 2012). Estas duas serão as plataformas de desenvolvimento dos serviços prestados pela infraestrutura de *e-Engineering*.

Quercus é uma implementação Java de PHP liberado sob a licença *open source*. Possui muitos módulos PHP e extensões, permitindo a integração de serviços Java com *scripts* PHP. Isso possibilita que aplicações PHP tirem proveito automaticamente dos recursos de servidor de aplicativos executando Java, assim como pool de conexões e sessões de cluster. Quercus apresenta uma nova abordagem Java/PHP mista para aplicações web e serviços, onde Java e PHP se integram, e aplicações PHP podem optar por usar bibliotecas Java. Esta capacidade revolucionária é possível porque o código PHP é interpretado/compilado em Java e as bibliotecas Quercus são escritas inteiramente em Java. Para facilitar essa nova arquitetura, Quercus oferece API e interface para expor as bibliotecas Java para PHP (CAUCHO RESIN, 2012).

EGroupWare é a ferramenta de colaboração *online* que fornece as mais diversas soluções em uma única plataforma, tais como: compartilhamento de informações, catálogo de endereços, calendário e agenda, gerenciamento de tarefas de projetos e eventos, servidor de arquivos *online* e gestão de documentos, troca de dados e sincronização, e-mail, gerenciamento de conteúdos, entre outros (EGROUPWARE, 2012). Assim como outros softwares do gênero, a versão *community* não conta com alguns serviços disponíveis apenas na versão *professional* (paga) como, por exemplo, videoconferência.

O RTH (*Requirements and Testing Hub*) é uma ferramenta *open source* de gerenciamento de requisitos e testes, que também tem capacidades rastreamento de *bugs* e gerenciamento de falhas (RTH BLOG, 2012). RTH é uma solução escrita e elaborada em PHP que pode oferecer funcionalidades abrangentes raramente encontradas em outros softwares, além de permitir a criação de projetos com a funcionalidade que permite mapear os requisitos para os casos de uso, e casos de uso para os casos de teste. Possui também a capacidade de comunicação básica e permite que informações sejam exportadas para planilhas. Sua operação é simples e intuitiva até mesmo para usuários inexperientes.

Apache ServiceMix é um flexível recipiente de integração *open source* que unifica os recursos e funcionalidades dos aplicativos Apache em uma poderosa plataforma de *runtime*, que pode ser usada para construir soluções de integrações próprias. É um ESB e também um conjunto de ferramentas para implementação de SOA (APACHE SERVICEMIX, 2012).

VirtualBox é um aplicativo de virtualização (*hypervisor*) de plataforma cruzada (*cross-plataform*), ou seja, uma vez instalado no sistema hospedeiro (*host*), estende as capacidades do computador existente para que ele possa executar vários sistemas operacionais (*guests*) dentro de várias máquinas virtuais, ao mesmo tempo. Assim, consegue-se uma consolidação da infraestrutura, pois a virtualização pode reduzir significativamente os custos de hardware e de energia elétrica, já que, na maioria das vezes, os computadores só utilizam uma fração de seu potencial e rodam com baixas cargas de trabalho (VIRTUALBOX, 2012). Por essa razão o *hypervisor* é considerado por muitos como sendo uma camada fundamental do ambiente de computação em nuvem e um componente crítico para nuvem privada (PIZETTE, RAINES, 2010).

KVM (*Kernel-based Virtual Machine*) é uma solução de virtualização completa de código aberto que permite executar várias máquinas virtuais. Cada máquina virtual tem hardware virtualizado privado: placa de rede, disco, placa de vídeo,

entre outros. É composto de um módulo de *kernel*[13] carregável que fornece a infraestrutura de virtualização do núcleo e um módulo de processador específico. O KVM está incluído no *kernel* Linux principal desde a versão 2.6.20, e é estável e rápido. Também está disponível como um *patch*[14] para versões recentes do *kernel* do Linux e como um módulo externo que pode ser usado com outras diversas distribuições Linux (KVM, 2012). Este software disponibiliza uma interface para programas no espaço de usuário (*user space*) que podem ser invocados através do arquivo especial do sistema (*/dev/kvm*) (TORALDO, 2012).

OpenVPN é um software de código aberto para VPN (*Virtual Private Network*) segura que utiliza os protocolos SSL (*Secure Sockets Layer*) e TLS (*Transport Layer Security*), e suporta os métodos de autenticação flexíveis com base em certificados, cartões inteligentes, e/ou o par de credenciais nome de usuário/senha. Essa solução acomoda uma grande variedade de configurações, incluindo acesso remoto, VPN *site-to-site*[15], segurança de rede sem fio; e em escala empresarial soluções de acesso remoto com balanceamento de carga, redundância (*failover*) e controles de acesso minuciosos (OPENVPN, 2012).

PopTop é uma solução baseada em software livre de servidor PPTP para Linux. O protocolo PPTP é um método para implementação de redes privadas virtuais que usa um canal de controle sobre TCP e um túnel GRE (*Generic Routing Encapsulation*) operacional para encapsular pacotes PPP (*Point-to-Point Protocol*). A especificação PPTP não descreve criptografia ou recursos de autenticação e conta com o PPP que está sendo encapsulado para implementar a funcionalidade de segurança (POPTOP, 2012).

[13] *Kernel* é o núcleo do sistema operacional.
[14] *Patch* é um programa ou *script* criado para atualizar ou corrigir um software.
[15] Tipo de VPN que conecta redes inteiras com outras, gerando uma extensão das *Intranets*. Nesse tipo, os *hosts* não têm software cliente VPN, pois o tráfego é feito através de um *gateway* VPN.

APÊNDICE B - AMBIENTE ADMINISTRATIVO

O ambiente administrativo é composto pela base da arquitetura, sob a qual estão os recursos e serviços utilizados para controlar a infraestrutura subjacente da nuvem CloudSatCD.

Para a implantação da nuvem está sendo utilizado o OpenNebula, um software para gestão e construção de centros de dados e infraestruturas virtualizadas em nuvem. Este ambiente segue o modelo de serviços IaaS e é o responsável pela criação da infraestrutura de nuvem privada. Sua utilização e operações estão restritas à administração da nuvem, uma vez que o objetivo principal de CloudSatCD é fornecer serviços de software aos usuários finais, e não recursos computacionais.

Seguindo a arquitetura do OpenNebula, as máquinas físicas estão divididas em dois grupos: *Front-end*, o nó principal responsável por executar tarefas de controle e administração da infraestrututura, criação, implantação e gerenciamento de máquinas virtuais, que serão executadas nos *Cluster Nodes*, responsáveis pela sustentação das máquinas virtuais do ambiente. Seus *hostnames* estão definidos como *cloudsatcontroller* para o *Front-end* e *cloudsatnodeX*, para cada nó físico controlado pelo *Front-end*, sendo X o número deste quando adicionado (Ex: *cloudsatnode1*, *cloudsatnode2*), onde são necessárias no mínimo duas máquinas físicas, uma de cada grupo. Neste trabalho foram utilizadas duas máquinas físicas, um *Front-end* e um *Cluster Node*, chamadas de *cloudsatcontroller* e *cloudsatnode1*, respectivamente. Entretanto, é possível adicionar novas máquinas físicas a qualquer momento.

O gerenciamento dos serviços do OpenNebula é feito no nó principal, o *Front-end*, pelo *daemon*[16] *(Disk And Execution MONitor*) *oned*. Os *drivers* que atuam diretamente nos componentes do sistema operacional também estão no *Front-end*, sendo eles (TORALDO, 2012): *Transfer Drivers*, utilizados para

[16] *Daemon* é um programa que executa de forma independente em *background* ao invés de ser controlado diretamente por um usuário. Tipicamente seus nomes terminam com a letra "d".

administrar as imagens de discos no sistema de armazenamento, seja em modo compartilhado ou modo não compartilhado; *Virtual Machine Drivers*, utilizados para gerenciar as instâncias de máquinas virtuais nos *hosts* baseados nos *hypervisores* específicos; e *Information Drivers*, utilizados para recuperar o status atual das instâncias de máquinas virtuais e *hosts*, baseados também nos *hypervisores* específicos, eles são copiados e remotamente executados em todos os *hosts* físicos através de SSH (*Secure Shell*).

Nos *Cluster Nodes* são executados os *hypervisores* responsáveis pela virtualização e controle do ciclo de vida dos sistemas virtualizados. Há uma variedade de *hypervisores* que podem ser utilizados, como KVM, Xen, VMware e VirtualBox - este último através do OneVBox (REDONDO et al., 2011) - provendo flexibilidade ao ambiente. Na nuvem CloudSatCD são utilizados o KVM e o VirtualBox.

A comunicação entre os nós é feita através de SSH numa rede privada, através do usuário *oneadmin*. Paralelamente o OpenNebula utiliza um sistema de armazenamento para o repositório de imagens de discos utilizadas pelas máquinas virtuais assim como outros arquivos, e sua transferência entre os *hosts* pode ser via modo não compartilhado, através de SSH, ou via sistema de arquivos compartilhado, normalmente utilizando NFS (*Network File System*) (OPENNEBULA, 2012), conforme retrata a Figura B.1.

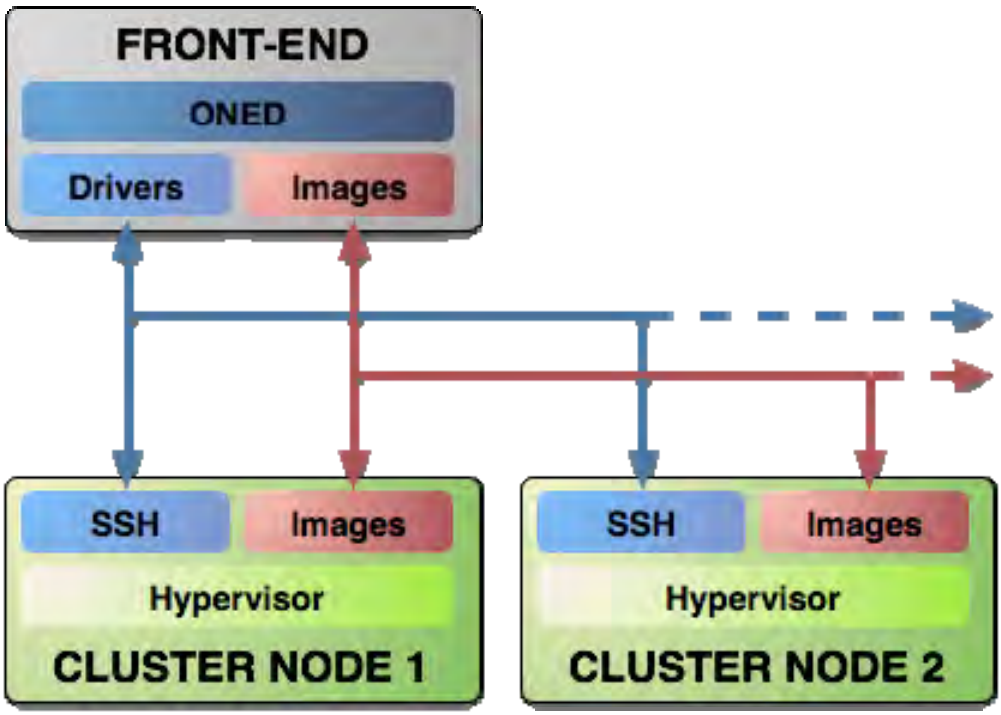


Figura B.1 - Sistema do OpenNebula
Fonte: Opennebula (2012)

Com o modo compartilhado é possível utilizar obter todas as vantagens das capacidades do *hypervisor*, como mover uma máquina virtual em execução entre diferentes máquinas físicas sem desconectar o cliente ou aplicação o mais rápido possível (*live migration*), além de alta velocidade de implantação de uma VM (*Virtual Machine*). De modo não compartilhado os arquivos são transferidos mais lentamente, via rede através de SSH, e o processo de migração de uma máquina virtual entre hosts só pode ser feito se a VM estiver completamente desligada (*cold migration*) (TORALDO, 2012). A infraestrutura da nuvem de *e-Engineering* CloudSatCD utiliza o modo compartilhado de armazenamento através de NFS, onde um diretório (*/srv/cloud/*) é compartilhado entre os *hosts Front-end* e *Cluster Node*, ilustrado na Figura B.2.

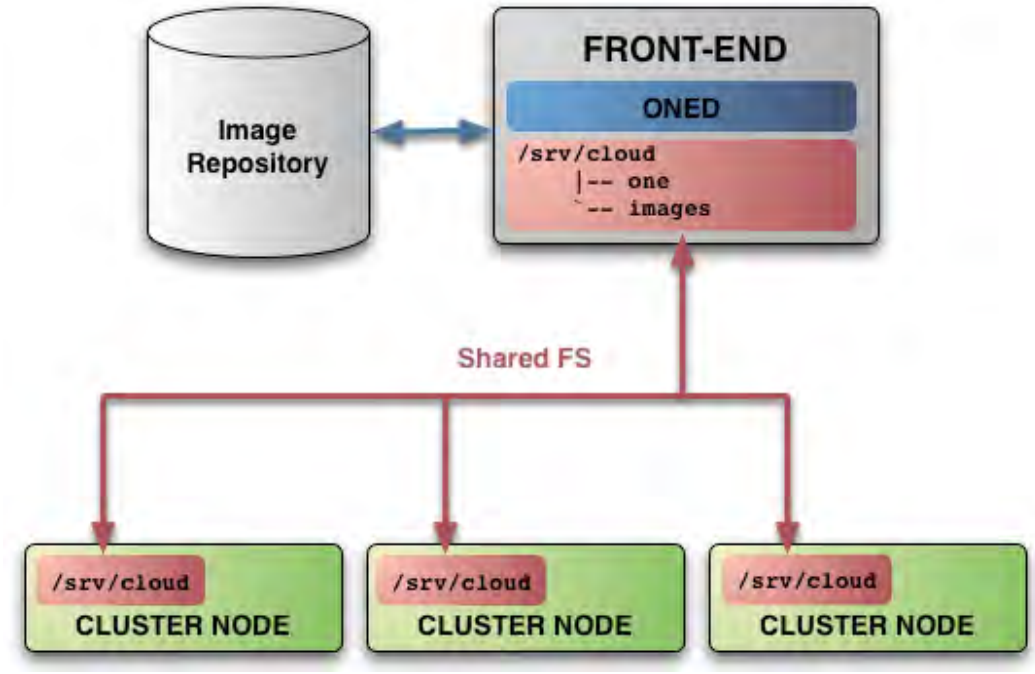


Figura B.2 - Repositório de imagens do OpenNebula em modo compartilhado
Fonte: Opennebula (2012)

Em cada uma das máquinas físicas utilizadas na infraestrutura CloudSatCD está instalado o sistema operacional Ubuntu, versão Server 12.04 x64 (64 bits). Por ser um sistema de endereçamento de 64 bits, não possui a restrição de memória existente nos sistemas x86 (32 bits) de reconhecer e trabalhar com até, no máximo, 4 GB (*GigaBytes*). Este sistema operacional foi escolhido devido a sua alta flexibilidade e integração com o OpenNebula, além de facilidade de instalação e administração num servidor completo e *open source*.

A preparação do sistema operacional inclui a instalação dos pacotes básicos e sua respectiva atualização, assim como configuração das interfaces de rede e seus endereços IP (*Internet Protocol*), nomes das máquinas (*hostnames*), usuário (*oneadmin*), grupo (*cloud*), diretórios (*/srv/cloud*), arquivos, inserção do *hostnames* conhecidos de cada máquina da rede no arquivo local de cada nó (*/etc/hosts*), permissões e serviços básicos utilizados pela infraestrutura, como SSH para acesso seguro entre as máquinas e administração remota via terminal.

A partir deste ponto as configurações começam a diferir entre *Front-end* e *Cluster Node*. O primeiro passo é a configuração das interfaces de rede adicionais, responsáveis pelas conexões entre os *hosts*. A interface secundária de *cloudsatcontroller* atua como *gateway* para a interface primária de *cloudsatnode1*, configurada em modo *bridge*, interligadas via *switch* privado. Este equipamento de rede é necessário para futura expansão do ambiente, pois, utilizando duas máquinas físicas apenas, o mesmo resultado poderia ser obtido utilizando conexão direta com cabo *cross-over*.

Configurações adicionais no serviço de SSH também são requeridas, visando permitir que as máquinas possuam acesso irrestrito entre si sem senha. Isso é alcançado através da geração de um par de chaves RSA[17], pública e privada, geradas no *Front-end* pelo usuário *oneadmin* e copiada a chave pública para o *host* remoto, *Cluster Node,* que usará a chave secreta para se conectar, utilizando também o usuário *oneadmin*, existente em ambos os *hosts*.

É necessária também a instalação e configuração do serviço de NFS (*Network File System*) para compartilhamento de arquivos e diretórios através da rede, com *cloudsatcontroller* sendo o servidor e *cloudsatnode1* sendo cliente. Uma vez configurado os *hosts* compartilham um diretório (*/srv/cloud)*, que é montado dinamicamente mesmo após reinício de qualquer uma das máquinas.

O próximo passo é a instalação do banco de dados no *Front-end*, o MySQL, e criação da base de dados de toda a infraestrutura com seu respectivo usuário, já iniciando a preparação para a instalação do software de criação da nuvem, OpenNebula.

A instalação dos *hypervisores* responsáveis por gerir as máquinas virtuais, KVM e VirtualBox, vem a seguir, sendo feita no *Cluster Node*. Ao contrário do KVM, a instalação do VirtualBox precisa de ajustes adicionais no sistema operacional pois não está contido diretamente na lista de softwares disponíveis

[17] RSA é um tipo de algoritmo de criptografia de dados considerado um dos mais seguros.

para o Ubuntu. Adicionalmente, também é necessário baixar e instalar manualmente o controlador (*driver*), denominado OneVBox, responsável pelo funcionamento/ligação do VirtualBox com o OpenNebula, já que este não possui suporte nativo àquele (REDONDO et al., 2011).

Com as máquinas devidamente preparadas, o OpenNebula pode ser instalado no *Front-end* pelo usuário *oneadmin*. Para ter maior flexibilidade durante a instalação, o software deve ser baixado diretamente do site do projeto e instalado pelo *source*. A customização da instalação, entre outras, serve para estabelecer o diretório padrão (*/srv/cloud/one*) e o grupo (*cloud*) para o OpenNebula.

A configuração do OpenNebula é feita em seu principal arquivo *(~/etc/oned.conf*) visando ajustar os parâmetros do banco de dados utilizado pelo software, o MySQL, neste caso, com a base de dados anteriormente criada. É preciso especificar a forma de transferência utilizada entre os nós no parâmetro TM_MAD, *Transfer Manager*. Este parâmetro indica ao OpenNebula qual é o modo, compartilhado ou não compartilhado, que deverá ser utilizado para transferir, criar, excluir e clonar imagens de máquinas virtuais. Apesar de ser possível utilizar mais de um modo de transferência numa infraestrutura, conforme descrito anteriormente, CloudSatCD utiliza o modo compartilhado através de NFS.

Também é necessário inserir manualmente as configurações para que o VirtualBox possa trabalhar em conjunto com o OpenNebula. As informações necessárias para especificar os *drivers* utilizados pelo VirtualBox são criadas através dos parâmetros IM_MAD e VM_MAD, *Information Manager* e *Virtualization Manager*, respectivamente. O primeiro é utilizado para obter informação dos anfitriões (hospedeiros) e são dependentes do *hypervisor* que está em uso, enquanto o segundo é utilizado para criar, administrar e monitorar VMs nos anfitriões. Tal qual o *Transfer Manager*, é possível habilitar os

múltiplos *drivers* de virtualização caso haja um ambiente com *hypervisores* diferentes, como é CloudSatCD, que usa KVM e VirtualBox.

Com estes ajustes feitos a configuração básica da infraestrutura está pronta e já é possível inicializar (com o comando *one start*) e operar o OpenNebula via linha de comando. Entretanto, para facilitar a administração, controle e monitoramento da nuvem, o OpenNebula dispõe de uma interface gráfica denominada OpenNebula Sunstone.

O OpenNebula Sunstone é um centro de operações apresentado numa interface web para gerenciamento da nuvem, residente no *Front-end*. Foi planejado para os usuários regulares e administradores, pois simplifica a execução de operações típicas e administração dos recursos em infraestruturas de nuvem privadas e híbridas, podendo ser adaptado a diferentes tipos de usuários. Possibilita ainda que o ambiente seja personalizado e estendido através de *plugins*.

O Sunstone permite também acessar uma máquina virtual via sessão VNC (*Virtual Network Computing*). Isto é alcançado usando um cliente VNC HTML5 baseado em *websocket*[18], o noVNC, no lado do cliente e um VNC *proxy* para tradução e redirecionamento das conexões de entrada no lado do servidor (TORALDO, 2012). Para habilitar o VNC no Sunstone, é preciso acessar o arquivo de configuração (*~/etc/sunstone-server.conf*) e ajustar os parâmetros referentes ao noVNC, bem como o endereço de rede, porta e método de autenticação de usuário pelos quais o Sunstone responde.

O último passo da configuração da interface web para o OpenNebula é permitir que o Sunstone utilize o VirtualBox. Isso deve ser feito através da inserção dos dados do *plugin* deste *hypervisor* no arquivo de *plugins* do Sunstone (*~/etc/sunstone-plugins.yaml*).

[18] *Websocket* é uma tecnologia que permite comunicação bilateral entre aplicativos e servidores sobre um único soquete TCP (*Transmission Control Protocol*), desde que suportem HTML5.

Feitos todos os ajustes e configurações necessários o OpenNebula Sunstone pode ser iniciado, com o comando *sunstone-server start*, e acessado via navegador através do par endereço e porta inseridos anteriormente no arquivo de configuração do Sunstone, e acessados pelo usuário *oneadmin*, através da interface mostrada na Figura B.3.

Figura B.3 - Tela de *login* do OpenNebula

Essa interface web utiliza interface XML-RPC[19], que é também usada pelos utilitários de linha de comando (TORALDO, 2012). Ao efetuar o *login*, o usuário é redirecionado para página inicial que exibe o painel de controle do OpenNebula (*Dashboard*). No topo da página, lado direito, é possível encontrar links úteis para recursos online do OpenNebula, tais como documentação, suporte e o site da comunidade responsável pelo projeto, além do link de *logout* para o usuário. Dentre estes, o link para a documentação é particularmente importante porque além de conter exemplos de configuração, principalmente via linha de comando, disponibiliza alguns exemplos de *templates* para diversos tipos de implementações.

[19] Protocolo de chamada de procedimento remoto (RPC) que utiliza XML para codificar suas chamadas e HTTP como um mecanismo de transporte.

O menu na esquerda é onde estão os recursos para monitoramento e administração da nuvem:

- *Dashboard*, a página inicial de monitoramento, são exibidas informações gerais em forma de gráficos e estatísticas de monitoramento, tais como: número de hosts, total de memória e processador em uso, taxas de transferência e largura de banda utilizada, total de máquinas virtuais e o estado delas, número de clusters, número de usuários e grupos, entre outros. Ainda possui um submenu de configuração onde é possui alterar o idioma e habilitar conexões seguras para *websockets* através de SSL (*Secure Sockets Layer*).

- *System* em sua página inicial exibe um resumo dos recursos do sistema e seus submenus *Users*, *Groups* e *ACLs* permitem, respectivamente, o gerenciamento avançado de usuários, grupos e controle de acesso.

- *Virtual Resources* em sua página inicial exibe um resumo dos recursos virtuais da infraestrutura e seus submenus *Virtual Machines*, *Templates* e *Images* permitem gerenciar de forma avançada, respectivamente, instâncias de máquinas virtuais, *templates* variados e o repositório de imagens.

- *Infrastructure* em sua página inicial exibe um resumo dos recursos de infraestrutura da nuvem e seus submenus *Clusters*, *Hosts*, *Datastores* e *Virtual Networks* permitem o gerenciamento avançado de, respectivamente, *clusters*, *hosts* (nós físicos), armazenamento e redes virtuais.

- *MarketPlace* exibe uma variedade de imagens que qualquer usuário da nuvem OpenNebula pode facilmente encontrar, importar para a infraestrutura local e implantar.

A Figura B.4 exibe a tela inicial do OpenNebula após o usuário executar o primeiro *logon*, ainda não tendo sido adicionado nenhum recurso à nuvem.

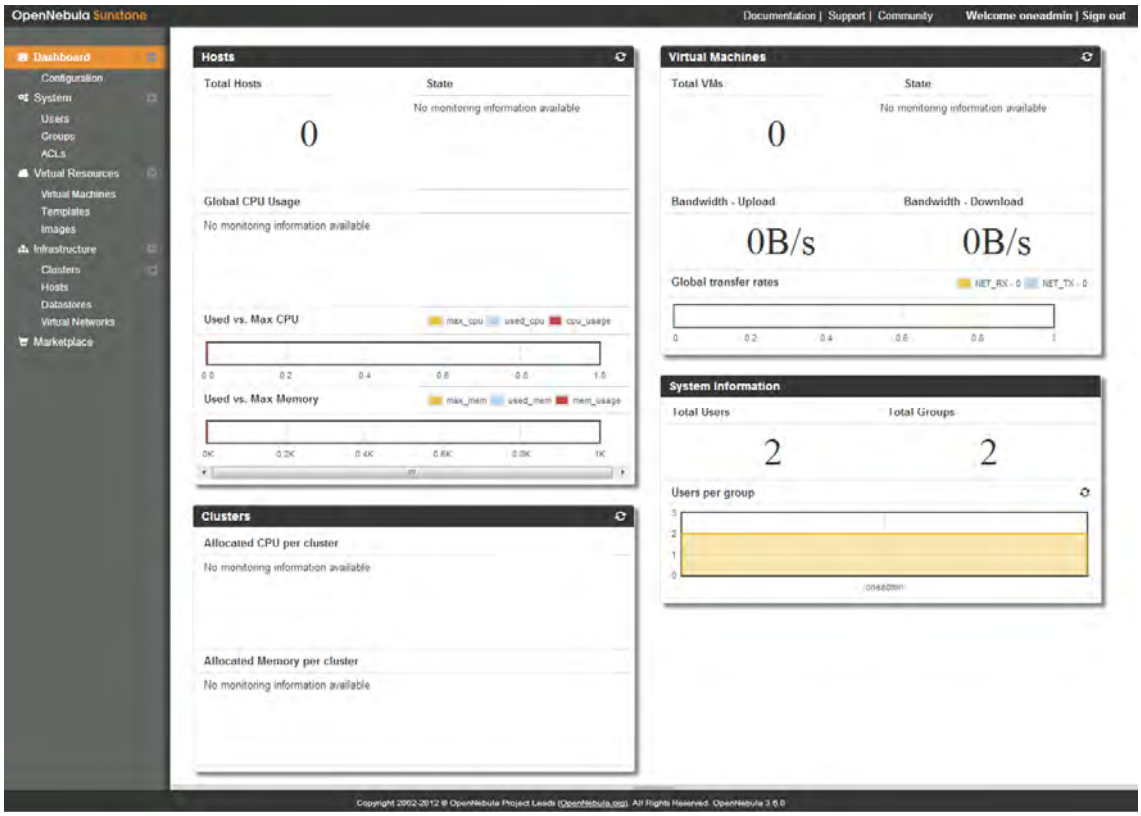


Figura B.4 - Tela inicial do OpenNebula

Primeiramente é preciso adicionar o *Cluster Node* ao OpenNebula, para que este seja capaz de monitorá-lo e executar ações sobre ele. Essa ação é feita ao acessar o menu *Infrastructure* e o submenu *Hosts*, conforme demonstrada na Figura B.5.

Em cada página recurso há um grupo de botões de ação que são utilizados a fim de interagir com o recurso atualmente selecionado. Quando um botão está com a cor cinza desbotada, é necessário primeiro selecionar pelo menos uma linha na tabela exibida abaixo dos botões usando as caixas (*checkbox*) à esquerda (TORALDO, 2012).

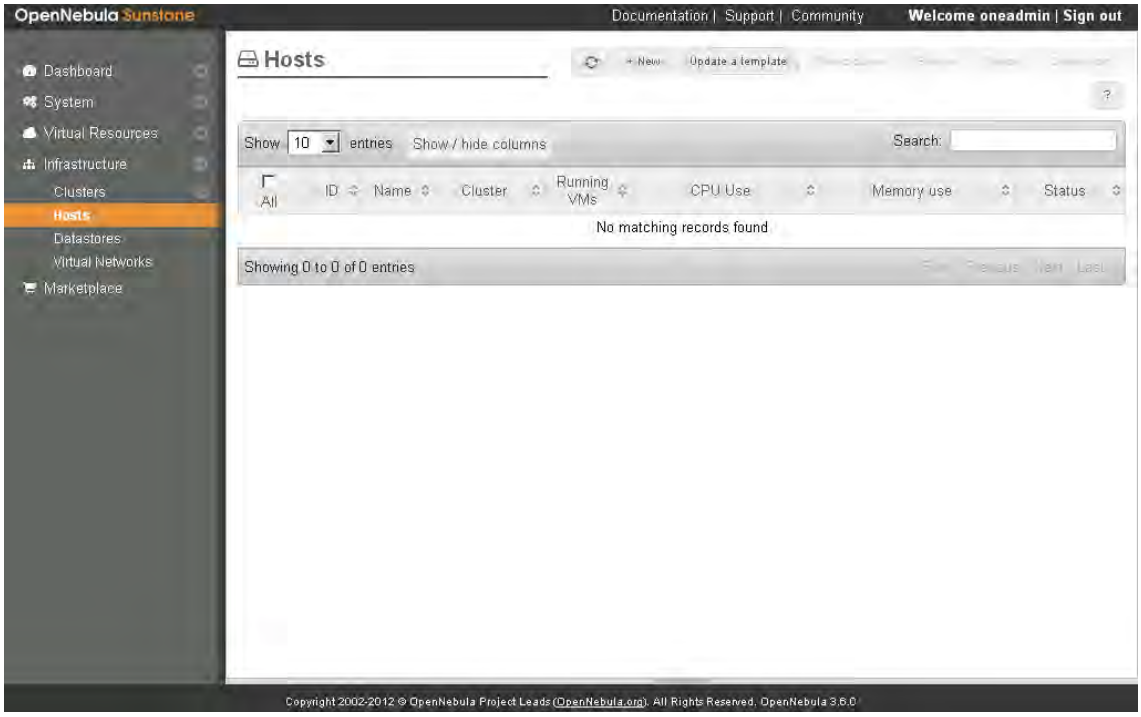


Figura B.5 - Gerenciamento de *hosts*

Ao pressionar o botão de nova ação (*New*), uma janela *pop-up* aparece solicitando as informações necessárias para adicionar um novo *host* para o banco de dados do OpenNebula. Este procedimento substitui a criação manual de um *template* para a adição de *host*, assim como o comando correspondente a essa ação via linha de comando (*onehost create*). Qualquer que seja a forma utilizada é preciso realizar previamente a configuração básica do *host*, incluindo o *login* SSH sem senha, como descrito anteriormente (TORALDO, 2012).

A Figura B.6 ilustra a janela *pop-up* exibida ao selecionar o botão *New*, com os campos preenchidos para adição do novo *host cloudsatnode1*, utilizando o KVM como *hypervisor* neste exemplo. Apenas o nome do host deve ser digitado, enquanto os outros campos são restritos às opções pré-configuradas no OpenNebula, escolhidas na caixa de seleção referente ao parâmetro considerado.

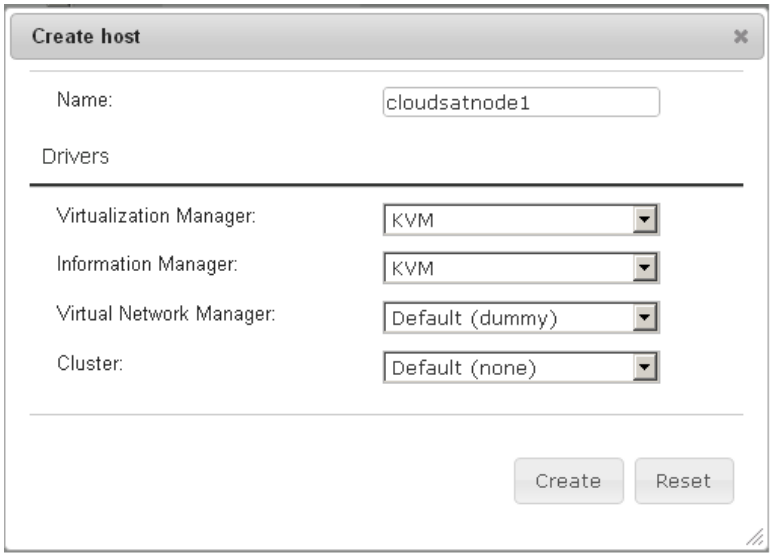


Figura B.6 - Inserção de *host*

Os parâmetros solicitados para a criação do *host* são:

- *Name*: nome do *host*.
- *Virtualization Manager*: *driver* gerenciador de virtualização (*hypervisor*), podendo ser KVM, VirtualBox, XEN, VMware, Dummy (*default*) e *Custom* (customizado). De acordo com o que está configurado no arquivo principal do OpenNebula (*oned.conf*) no parâmetro *VM_MAD*.
- *Information Manager*: *driver* gerenciador de informação (baseado no *hypervisor*) usado para monitorar o *host*, podendo ser KVM, VirtualBox, XEN, VMware, Dummy (*default*) e *Custom* (customizado). De acordo com o que está configurado no arquivo principal do OpenNebula (*oned.conf*) no parâmetro *IM_MAD*.
- *Virtual Network Manager*: *driver* gerenciador de redes virtuais utilizado, podendo ser *Default* (*driver* padrão que não realizar qualquer operação de rede; regras de *firewall* também são ignoradas), *Firewall* (restrições e políticas de acesso são aplicadas), *802.1Q* (restringe o acesso à rede através de *VLAN tagging*, que requer também apoio das chaves de

hardware), *ebtables* (restringe o acesso à rede por meio de regras *ebtables*; nenhuma configuração especial de hardware é requerida), *Open vSwitch* (restringe o acesso à rede com o Open Virtual Switch), *VMware* (usa a infraestrutura de rede VMware para fornecer uma rede isolada e 802.1Q compatível para VMs executadas sob o *hypervisor* VMware), e *Custom* (pode-se configurar conforme a necessidade deixando a rede customizada).

- *Cluster*: conjunto ou grupo de *hosts* ao qual o *host* criado fará parte.

Ao pressionar o botão para criação (*Create*), o novo *host* é apresentado na listagem central, e uma pequena caixa com uma mensagem empilhada com a ação executada será exibida no canto inferior direito da interface web. A Figura B.7 exibe a criação do novo *host* chamado *cloudsatnode1* feita através da interface web do Sunstone, enquanto a mesma criação do *host cloudsatnode1* via linha de comando tem o formato demonstrado na Figura B.8.

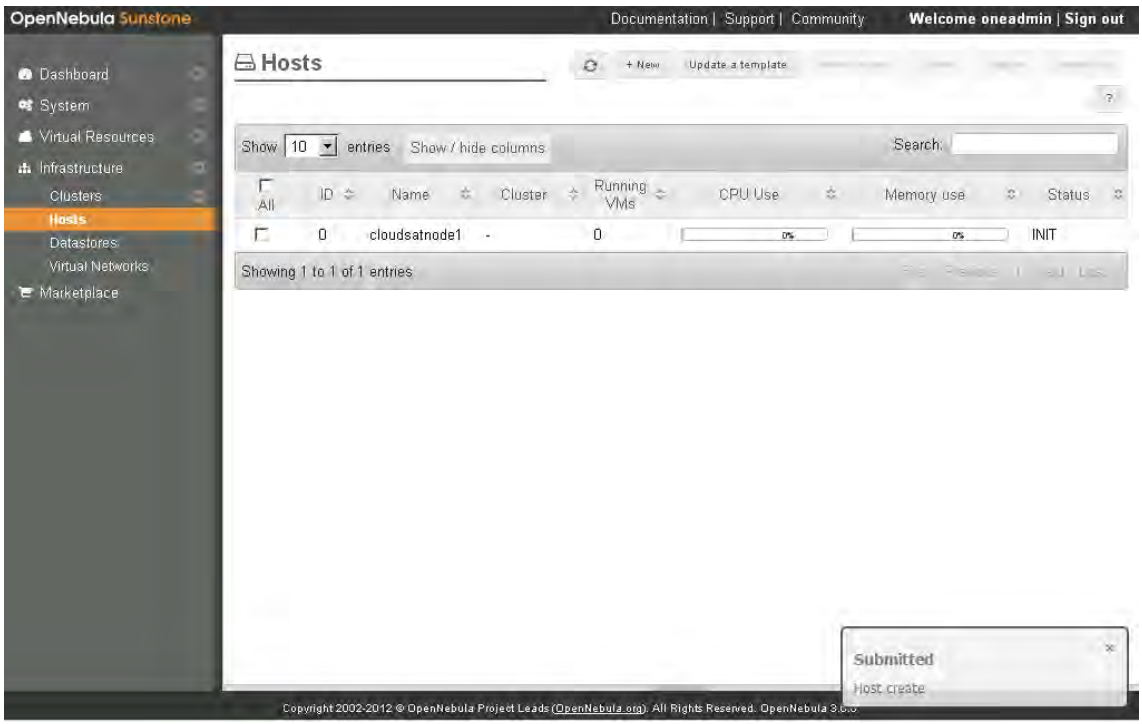


Figura B.7 - *Host* adicionado

```
oneadmin@cloudsatcontroller:~$ onehost create cloudsatnode1 -i im_kvm -v vmm_kvm -n dummy
```

Figura B.8 - Inserção de *host* via linha de comando

O quadro interno mostra os sistemas atualmente configurados juntamente com o seu número de identificação (*ID*), nome (*Name*), contagem das VMs em execução (*Running VMs*), uso de processador (*CPU Use*), uso de memória (*Memory Use*) e estado (*Status*) do *host*. As informações exibidas não são realmente em tempo real, mas são, de fato, os dados consultados recuperados pelo *driver* do gerenciador de informação (*Information Manager*), portanto seu tempo de atualização depende estritamente de como estão configurados atributos como *HOST_MONITORING_INTERVAL* e *HOST_PER_INTERVAL*, no arquivo principal do OpenNebula (*oned.conf*) (TORALDO, 2012).

É possível utilizar a caixa de seleção (*checkbox*) esquerda para selecionar o número de entradas em cada listagem exibidas na página e a caixa de texto para busca (*Search*) para filtrar os elementos apresentados, por exemplo, em busca de um nome de máquina específico (TORALDO, 2012). Também é possível escolher e alterar quais colunas são exibidas (*Show / hide columns*). A tabela ainda pode ser ordenada por uma determinada coluna clicando no cabeçalho da coluna correspondente ao ordenamento desejado.

O segundo botão de ação (*Update Template*) pode ser usado para alterar os atributos de *hosts* já criados. Neste caso específico, a maioria dos atributos do host é atualizada dinamicamente pelos *drivers* do gerenciador de informação (*Information Manager*) de seus respectivos *hypervisores* (TORALDO, 2012). Os outros botões servem para, respectivamente, selecionar um cluster (*Select Cluster*) ativar (*Enable*), desativar (*Disable*) e excluir (*Delete host*) um *host*. Do lado oposto há um botão para atualização da página central e dados exibidos (*Refresh*).

Quando uma linha da tabela é clicada, um painel de fundo aparece contendo informações detalhadas sobre os recursos selecionados, como ilustrado na

imagem seguinte. Na primeira guia (*Host Information*) são encontrados detalhes sobre o host da linha clicada como, por exemplo, o uso de recursos físicos e as VMs executando sobre ele. Na segunda aba (*Host template*) são listados os valores de atributos do *template* utilizado. A terceira e última aba (*Monitoring information*) exibe gráficos sobre o uso processador e memória no host selecionado, dando uma visão recente do uso de recursos do host atualmente selecionado para monitoramento (TORALDO, 2012).

No Sunstone praticamente todos os recursos da interface possuem funcionamento análogo ao comportamento apresentado pelo gerenciador de *hosts* visto, tais como botões, interfaces, *pop-ups*, caixas de seleção, entre outros, todos sendo intuitivos. Por tal razão o mesmo será demonstrado com menos detalhes nas seções adiante, dando prioridade apenas ao que realmente está sendo utilizado para criação do ambiente administrativo.

A partir deste ponto começam as configurações que irão possibilitar a criação do ambiente operacional. Uma vez conhecidos os sistemas operacionais convidados (*guests*) que farão parte da infraestrutura como instâncias de máquinas virtuais, é preciso obter e criar as respectivas imagens para instalação.

As imagens, que ficarão no repositório de imagens anteriormente citado (*/srv/cloud/one/images*), podem ser importadas a partir do *MarketPlace*, baixadas diretamente ou criadas a partir de um disco de instalação, seja diretamente ou transferidas via SCP (*Secure Copy*) a partir de um cliente qualquer com acesso remoto ao servidor da nuvem CloudSatCD.

Com a imagem no diretório do repositório da infraestrutura é preciso adicioná-la (botão *New*) ao OpenNebula através do menu responsável pelos recursos virtuais (*Virtual Resources*) no link do gerenciador de imagens (*Images*). A Figura B.9 exibe a tela para inserção de imagens de sistemas operacionais no OpenNebula, via Sunstone.

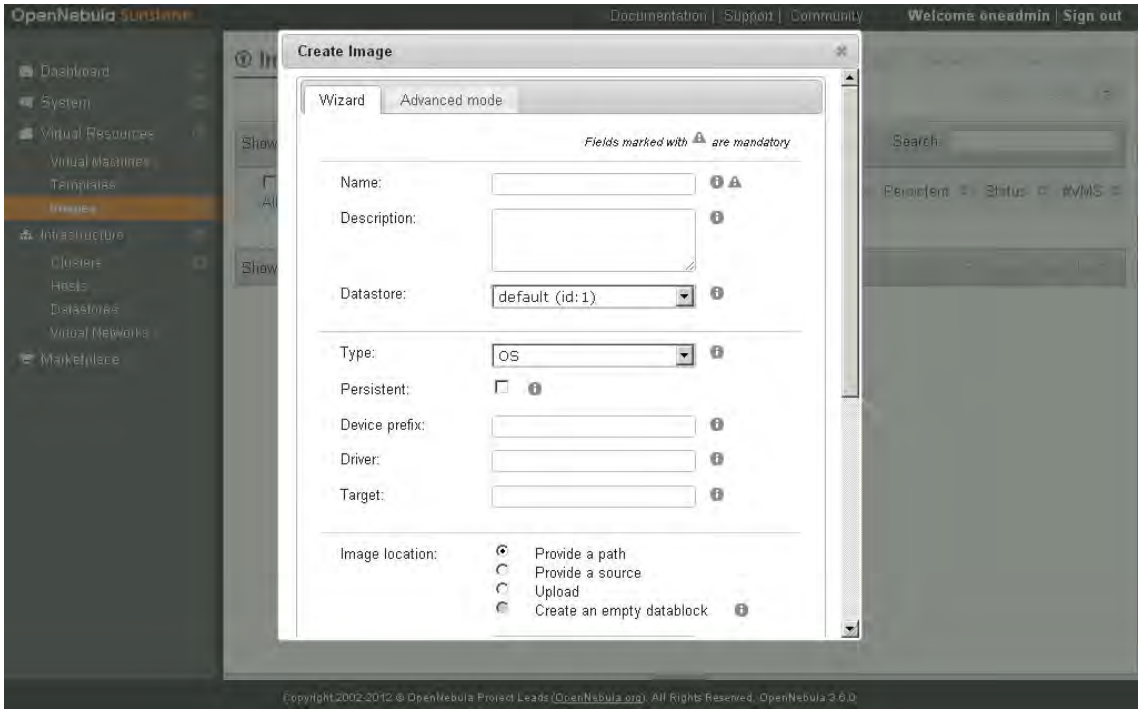


Figura B.9 - Inserção de imagem de sistema operacional

Há duas formas para criação de novas imagens: padrão (*Wizard*) e avançado (*Advanced mode*). O primeiro é baseado em campos que devem ser completados, guiando a criação, enquanto o segundo é baseado em *template*, exigindo a criação de modo textual com os parâmetros desejados.

Seguindo o modo padrão, os principais parâmetros a serem preenchidos são (TORALDO, 2012):

- *Name*: nome para a imagem, devendo ser único. É um atributo obrigatório.
- *Datastore*: o meio de armazenamento, ou seja, o repositório em si.
- *Type*: o tipo de imagem, podendo ser "*OS*" (contém um sistema operacional, cada modelo VM deve definir um disco referindo-se a uma imagem deste tipo), "*CDROM*" (imagem com dados somente para leitura; apenas uma imagem deste tipo pode ser usada em cada modelo

de máquina virtual), e "*DATABLOCK*" (imagem é um bloco de armazenamento de dados que podem ser acessados e modificados a partir de diferentes máquinas virtuais; pode ser criada a partir de dados anteriores existentes ou como uma unidade vazia). Pode ser omitido e o valor *default "OS"*, pré-configurado no arquivo principal do OpenNebula (*oned.conf*), é selecionado.

- *Device prefix*: para emulação do dispositivo com o qual a imagem será montada, podendo ser hd (*IDE*), sd (*SCSI*). Pode ser omitido e o valor *default "hd"*, pré-configurado no arquivo principal do OpenNebula (*oned.conf*), é selecionado.

- *Driver*: especifica um *driver* para mapear a imagem. KVM: *raw*, *qcow2*; Xen: *tap:aio:*, *file:*.

- *Image location*: forma de encontrar a imagem, podendo ser *PATH* (caminho para encontrar no sistema), Source (fonte a ser utilizada no atributo DISK, sendo útil para imagens não baseadas em arquivos), *Upload* (para enviar a imagem a partir de computador remoto) ou *Create an empty datablock* (para criar um bloco de dados vazio). É um atributo obrigatório.

Depois de confirmar a criação de uma nova imagem pode ser preciso aguardar que ela seja copiada da origem para o repositório. Na tabela de imagens são encontrados atributos comuns, incluindo status, por exemplo, iniciada (*INIT*), pronta (*READY*), usada (*USED*), e desabilitada (*DISABLED*), além da contagem da VMs que as estão usando no momento (TORALDO, 2012).

A Figura B.10 mostra as imagens criadas na infraestrutura de nuvem CloudSatCD, sendo duas, FreeBSD e Ubuntu, utilizadas para a criação das VMs, vistas adiante.

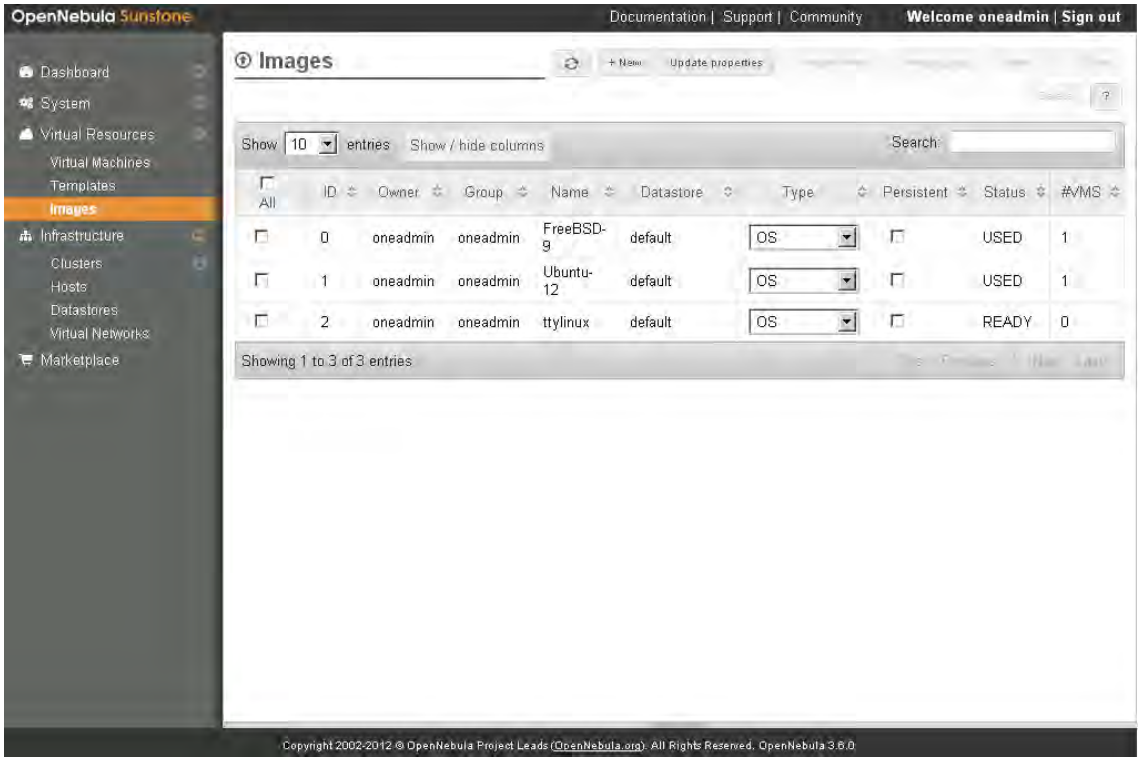


Figura B.10 - Gerenciamento de imagens de sistemas operacionais

O OpenNebula fornece um subsistema de rede facilmente adaptável e personalizável, a fim de ser integrado da melhor forma aos requisitos específicos da rede existente. Para oferecer conectividade de rede para as máquinas virtuais através dos diferentes *hosts*, a configuração padrão conecta a interface de rede da máquina virtual a uma ponte (*bridge*) no *host* físico.

Para que as máquinas virtuais se comuniquem, uma rede virtual deve ser criada. Um *host* pode estar ligado a uma ou mais redes, que estão disponíveis para as máquinas virtuais através das pontes (*bridges*) correspondentes. O OpenNebula permite a criação de redes virtuais mapeando-as sob redes físicas (OPENNEBULA, 2012). A Figura B.11 ilustra o modo de funcionamento das redes entre nó físico, máquina virtual e a Internet.

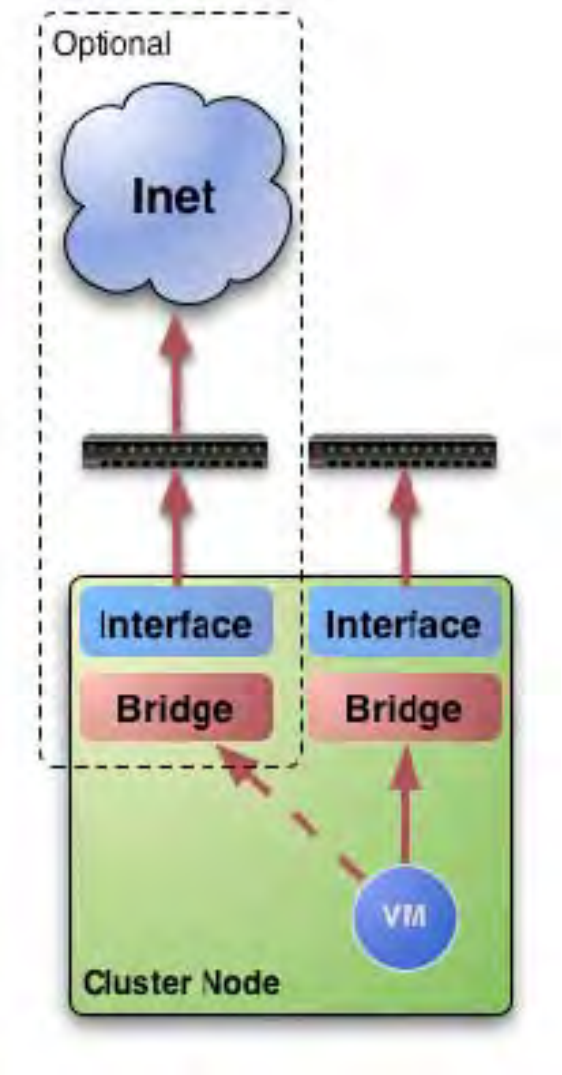


Figura B.11 - Rede entre máquina virtual, nó físico e Internet
Fonte: OPENNEBULA (2012)

O modo de criação é semelhante aos vistos anteriormente para *hosts* e imagens, sendo acessado pelo botão de adição de recursos (*New*) através do menu *Infrastructure* o link *Virtual Networks*. A Figura B.12 exibe a janela *pop-up* que surge na tela ao iniciar a inserção de uma nova rede no OpenNebula.

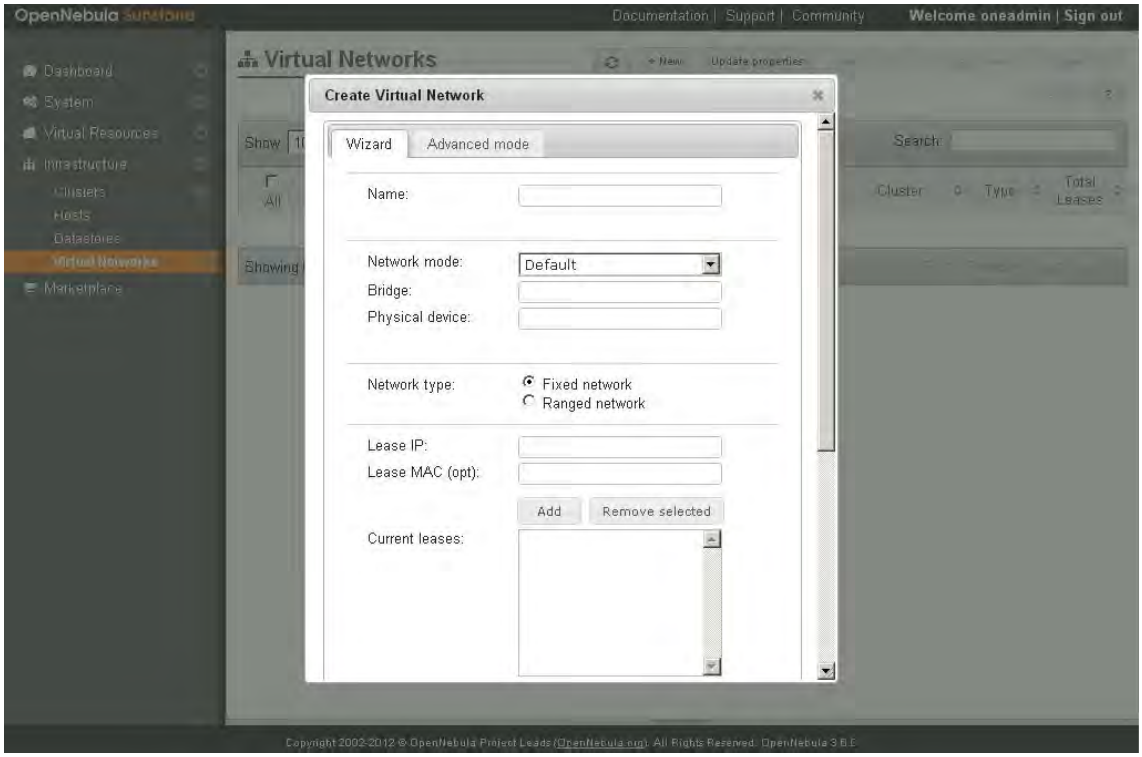


Figura B.12 - Inserção de rede virtual

Os principais campos a serem preenchidos no modo padrão (*Wizard*) são (TORALDO, 2012):

- *Name*: nome da rede virtual a ser criada.
- *Network mode*: modo da rede, podendo ser *Default* (*driver* padrão que não realizar qualquer operação de rede; regras de *firewall* também são ignoradas), *802.1Q* (restringe o acesso à rede através de *VLAN tagging*, que requer também apoio das chaves de hardware), *ebtables* (restringe o acesso à rede por meio de regras *ebtables*; nenhuma configuração especial de hardware é requerida), *Open vSwitch* (restringe o acesso à rede com o Open Virtual Switch), e *VMware* (usa a infraestrutura de rede VMware para fornecer uma rede isolada e 802.1Q compatível para VMs executadas sob o *hypervisor* VMware).
- *Bridge*: a interface responsável por fazer *bridge*, normalmente br0.

- *Physical device*: nome do dispositivo de rede físico que será anexado à ponte (bridge). É obrigatório apenas com o uso do *driver 802.1Q*.

- *Network type*: tipo de rede, podendo ser *Fixed network* (precisa de uma série de atributos para concessões – *leases* – de endereços IP utilizados pelas VMs) ou *Ranged network* (ao contrário da variação das concessões, contém uma gama de endereços IP que podem ser definidos de forma flexível utilizando atributos como endereço de rede, máscara de rede, entre outros).

Na tabela principal é exibida cada uma das redes com seus respectivos parâmetros gerais. A Figura B.13 exibe as redes criadas na infraestrutura de nuvem CloudSatCD, sendo *cloudsatcd-net* a rede utilizada pela infraestrutura, enquanto a rede *Small network* foi utilizada apenas para testes.

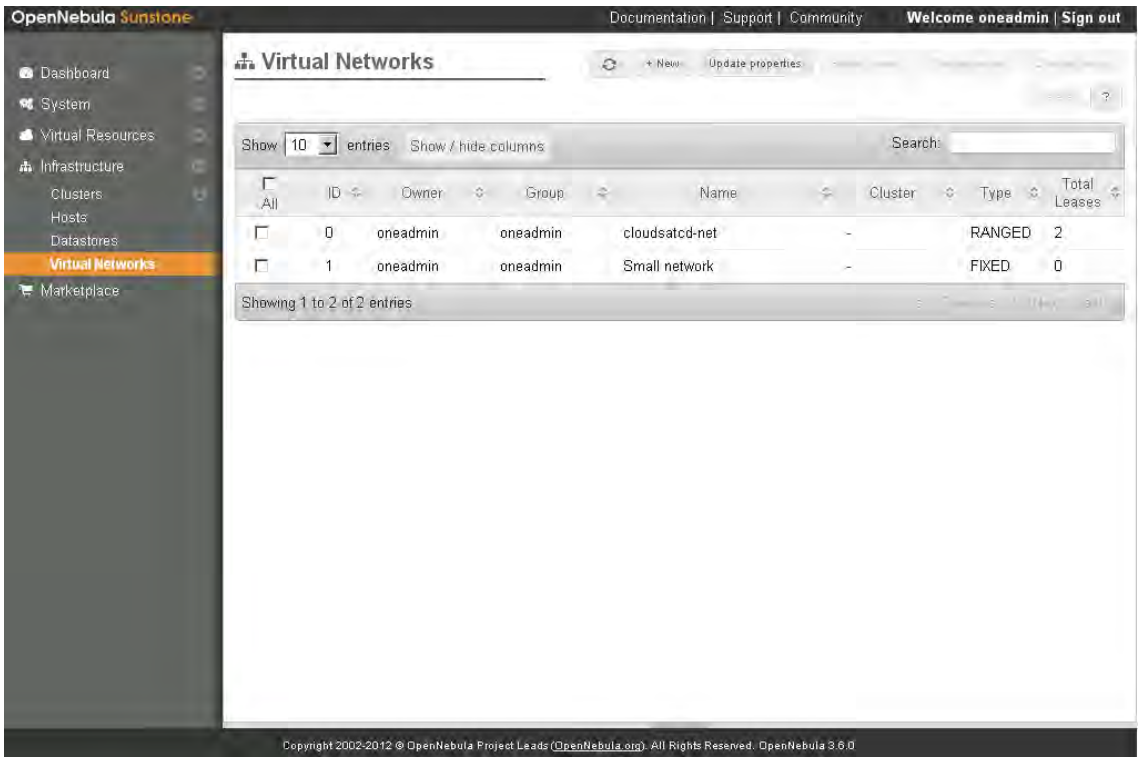


Figura B.13 - Gerenciamento de redes virtuais

As máquinas virtuais são criadas no menu correspondente a recursos virtuais (*Virtual Resources*). Entretanto, o próximo passo é criar um *template* para a nova máquina virtual com as características desejadas. O gerenciamento de templates (*Templates*) é uma das tarefas mais complexas, pois é preciso conhecer previamente os valores e instruções que serão passadas para o OpenNebula criar a VM. O Sunstone torna esta atividade menos árdua para os administradores de infraestrutura dispondo de um configurador padrão (*Wizard*) que auxilia a tarefa de criação através de campos pré-definidos para serem preenchidos com as características desejadas.

Ao pressionar o botão de nova ação (*New*), uma janela *pop-up* com o configurador aparece, conforme demonstrado na Figura B.14. Nas abas são exibidos os *hypervisores* disponíveis e, dependendo de qual será utilizado, a aba correta deve ser escolhida para iniciar a configuração do *template*. Em cada assistente alguns atributos são ocultados ou disponíveis, dependendo do *hypervisor* utilizado (TORALDO, 2012).

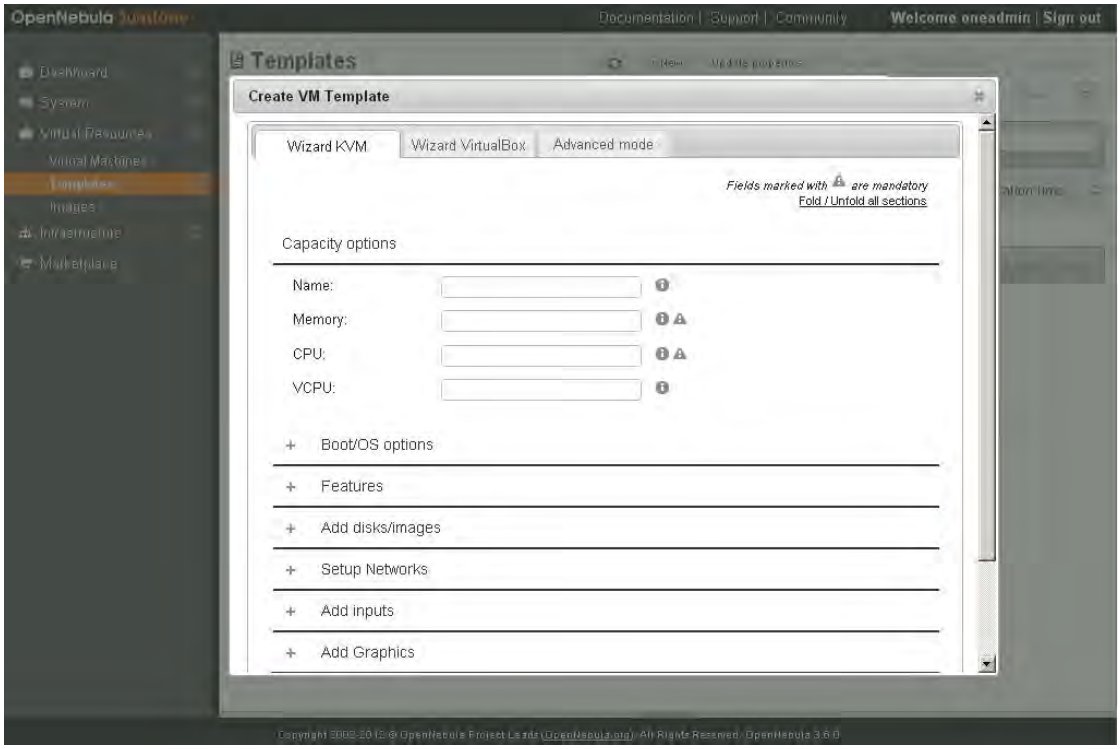


Figura B.14 - Criação de *templates* de máquinas virtuais

As principais seções do modo padrão (*Wizard*) são descritas a seguir (TORALDO, 2012):

- *Capacity options*: é usada para definir a configuração de base da VM, tal como o seu nome, a memória atribuída, número de CPUs, e número de CPUs virtuais associadas.

- *Boot/OS options:* é utilizada para definir a arquitetura de CPU da VM e opções de inicialização. O campo método de inicialização (*Boot method*) se expande caso a opção *kernel* seja selecionada, permitindo definir *kernel* personalizado.

- *Features*: permite ativar ou desativar os recursos da ACPI (*Advanced Configuration and Power Interface*) e PAE (*Physical Address Extension*) para o modelo VM específico. O recurso ACPI é usado principalmente para lidar com ações de desligamento do *host*, e PAE é usado para permitir a utilização de mais de 4 GB de memória em uma VM com arquitetura i686.

- *Add disks/images:* é utilizada para adicionar discos e imagens, permitindo conectar um ou mais discos na máquina virtual. É possível fixar uma imagem de disco personalizada, fornecendo um caminho absoluto (*PATH*), ou usando uma imagem registrada no repositório OpenNebula. Dependendo do *hypervisor*, há várias opções para o tipo de barramento (*Bus*) e atributos de *driver (Target* e *Driver)*, estes dois últimos se deixados em branco usam a opção *default* para o *driver* do gerenciador de VM. A cada disco ou imagem selecionado é preciso adicioná-lo (*Add*) para que as opções escolhidas sejam salvas

- *Setup Networks:* refere-se à instalação e configuração da rede. É possível selecionar uma rede virtual já existente, anteriormente criada em *Virtual Networks*. Pode-se especificar manualmente um endereço IP ou deixar em branco para que o OpenNebula atribua automaticamente

um endereço IP disponível com base nas configurações de rede. É possível também configurar o *firewall* da VM, por exemplo, permitindo o acesso apenas a SSH, HTTP, outros protocolos, e filtragem de portas comumente usadas, tanto para TCP (*Transmission Control Protocol*) quanto para UDP (*User Datagram Protocol*). Além disso, pacotes ICMP (*Internet Control Message Protocol*), utilizados principalmente para diagnósticos de rede podem ser descartados ou aceitos.

- Add Graphics: é utilizada para configurar uma sessão VNC ligado a qualquer instância. O campo *Listen IP* deve ser especificado como 0.0.0.0, uma vez que o padrão é 127.0.0.1, acessível apenas a partir do *host* no qual ele está sendo executado.

- *Add context variables*: adiciona múltiplas variáveis de contexto passadas e disponibilizadas para o *script* contextualização.

Ao concluir o *template* e criá-lo (*Create*), o mesmo será submetido ao repositório. A última aba para criação de *templates* de VM permite criá-los de forma avançado (*Advanced Mode*), e pode ser usado para colar diretamente um arquivo de *template*, tal qual seria submetê-lo através de linha de comando (*onetemplate*) (TORALDO, 2012).

A seção do Sunstone mais utilizada corresponde às máquinas virtuais (*Virtual Machines)*, também dentro do menu de recursos virtuais (*Virtual Resources*). As principais operações de controle, administração e monitoramento de instâncias de VMs são realizadas nesta seção.

Para instanciar uma nova VM o botão de adição de recursos (New) deve ser pressionado, abrindo uma janela *pop-up* com o formulário correspondente, conforme ilustrado na Figura B.15. Esta ação é extremamente simplificada devido a sua integração com o repositório de *templates*, de forma que é essencial especificar um *template*, assim como o nome da VM e a quantidade de instâncias desejadas, numa única ação, agilizando o processo de criação.

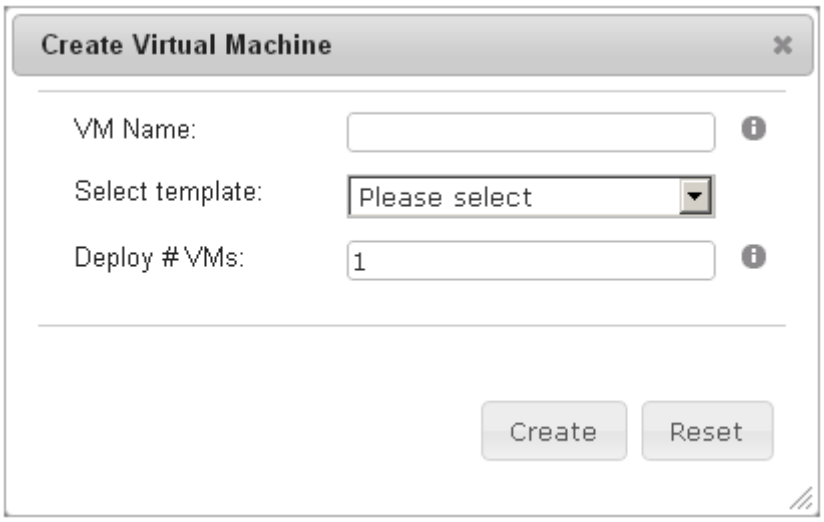


Figura B.15 - Criação de máquinas virtuais

Ao criar várias instâncias os nomes das VM são gerados automaticamente anexando um ID incremental no final do nome especificado na caixa de diálogo (*VM Name*) (TORALDO, 2012)

No topo da página encontram-se os botões de ação aplicados às VMs criadas. É possível atualizar as propriedades (*Update properties*), alterar o dono (*Change owner*) e o grupo (*Change group*), desligar (*Shutdown*) e excluir (*Delete*) uma ou mais máquinas virtuais.

Há também um grupo de ações contido dentro do botão referente a ações prévias (*Previous actions*). Dentre as opções disponíveis neste botão estão as ações para implantar (*Deploy*), migrar (*Migrate*), migrar sem interromper o funcionamento (*Live migration*), travar (*Hold*), liberar (*Release*), suspender (*Suspend*), retomar (*Resume*), parar (*Stop*), reiniciar (*Restart*), reenviar (*Resubmit*), reinicializar (*Reboot*), restabelecer (*Reset*) e cancelar (*Cancel*). Todas essas ações equivalem a comandos disponíveis através da linha de comando, com o utilitário *onevm*.

As VMs instanciadas são listadas na tabela principal contida no centro da página com as principais informações sobre cada uma, tais como dono, grupo,

nome, nome do *host*, endereço IP alocado e estado atual. Se uma máquina virtual em execução foi configurada com gráfico VNC habilitado e foi iniciada com êxito, um ícone colorido para iniciar uma sessão VNC diretamente a partir do navegador web (com suporte a *websockets*) é exibido. Se o ícone estiver cinza, a máquina ainda não está em um estado de execução ou o *template* não contém uma seção gráfica com suporte VNC habilitado (TORALDO, 2012).

A Figura B.16 contextualiza a página descrita anteriormente e ilustra duas VMs instanciadas em CloudSatCD, uma com o sistema operacional FreeBSD 9 x86 e outra com Ubuntu Server 12 x86, ambas em execução (*RUNNING*) e com suporte a VNC ativado.

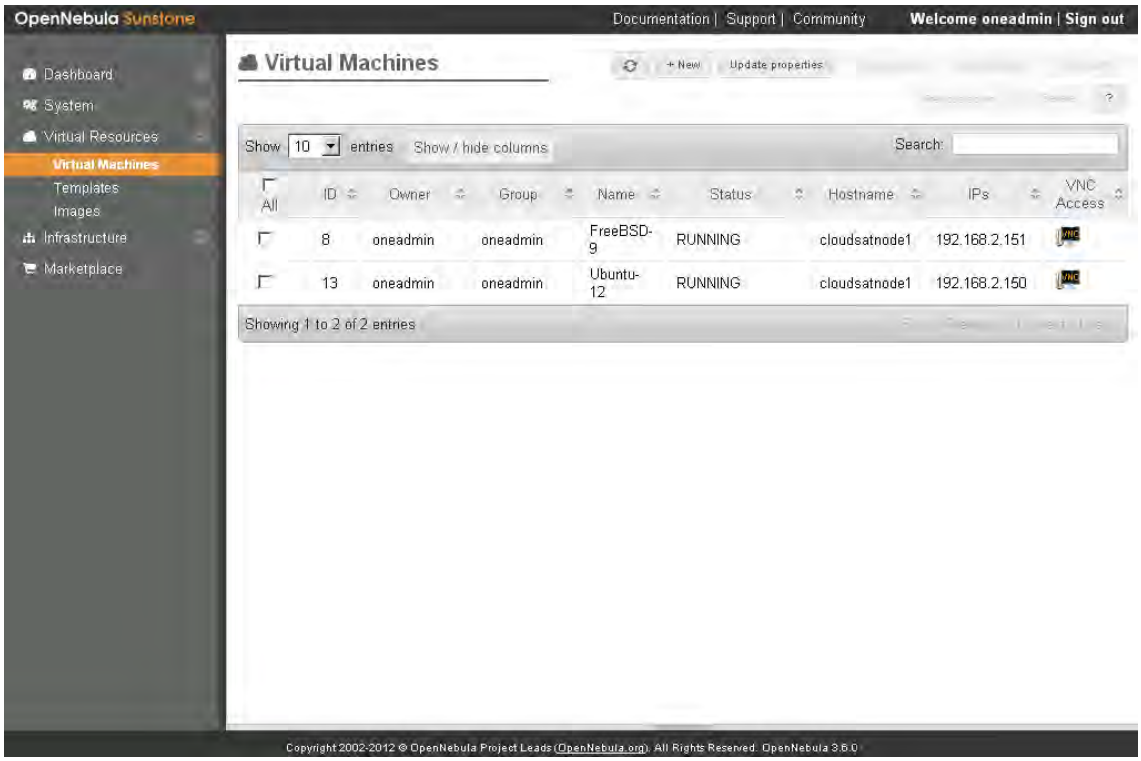


Figura B.16 - Gerenciamento de máquinas virtuais

Ao clicar no ícone do VNC uma janela *pop-up* aparece permitindo que o usuário se conecte diretamente ao console principal da VM e possibilitando executar comandos e ações como se estivesse em frente ao monitor. Há

também um botão especial [*Ctrl+Alt+Del*] que envia essa combinação de teclas para a VM (TORALDO, 2012), conforme pode ser visto na Figura B.17.

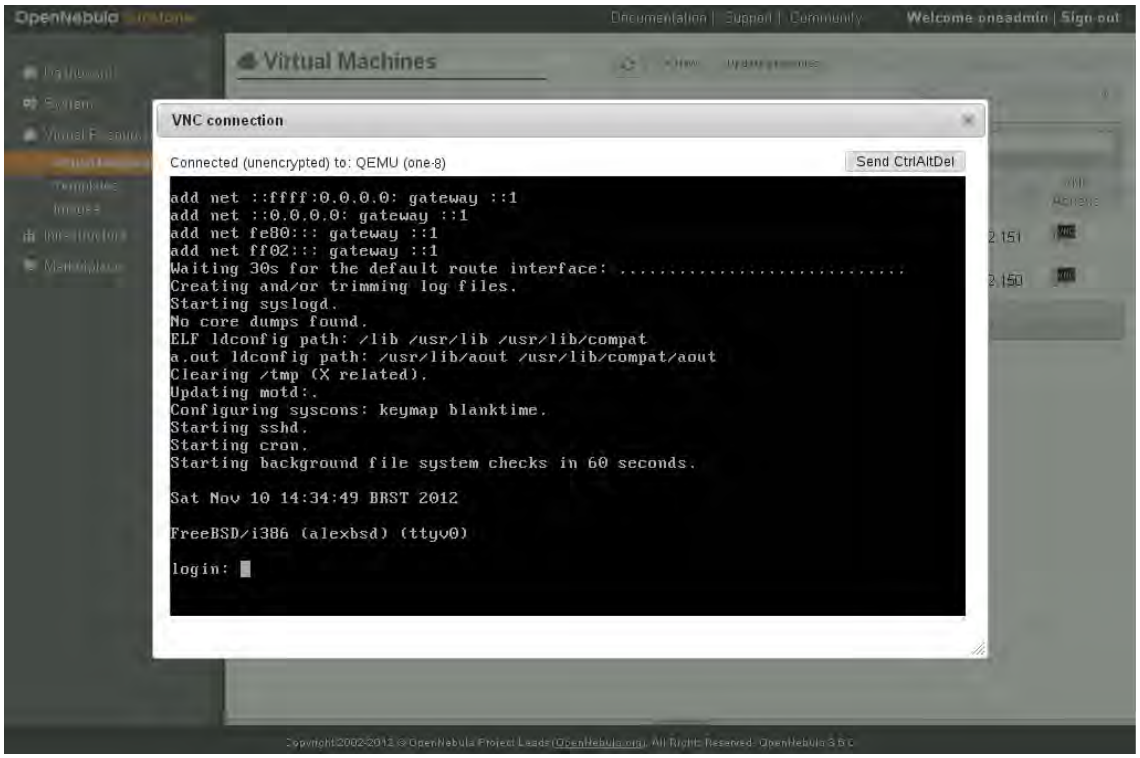


Figura B.17 - Acesso à máquina virtual via navegador no OpenNebula

Mesmo com este recurso do noVNC disponibilizado pelo OpenNebula Sunstone, é possível acessar as máquinas virtuais através de um cliente VNC comum, utilizando a endereço de rede do *host* e a porta para a VM requisitada.

Informações adicionais sobre uma VM específica podem ser obtidas clicando em sua respectiva linha, pois um painel de fundo aparece na parte inferior da tabela contendo abas com dados: da VM (*VM Information*), do disco (*Disk & Hotplugging*), do *template* utilizado (*VM Template*), dos registros de atividades (*VM log*), do histórico (*History information*), e de gráficos de monitoramento (*Monitoring information*).

A Figura B.18 ilustra um exemplo de painel de informações adicionais para uma determinada VM, neste caso, a que roda o sistema operacional FreeBSD.

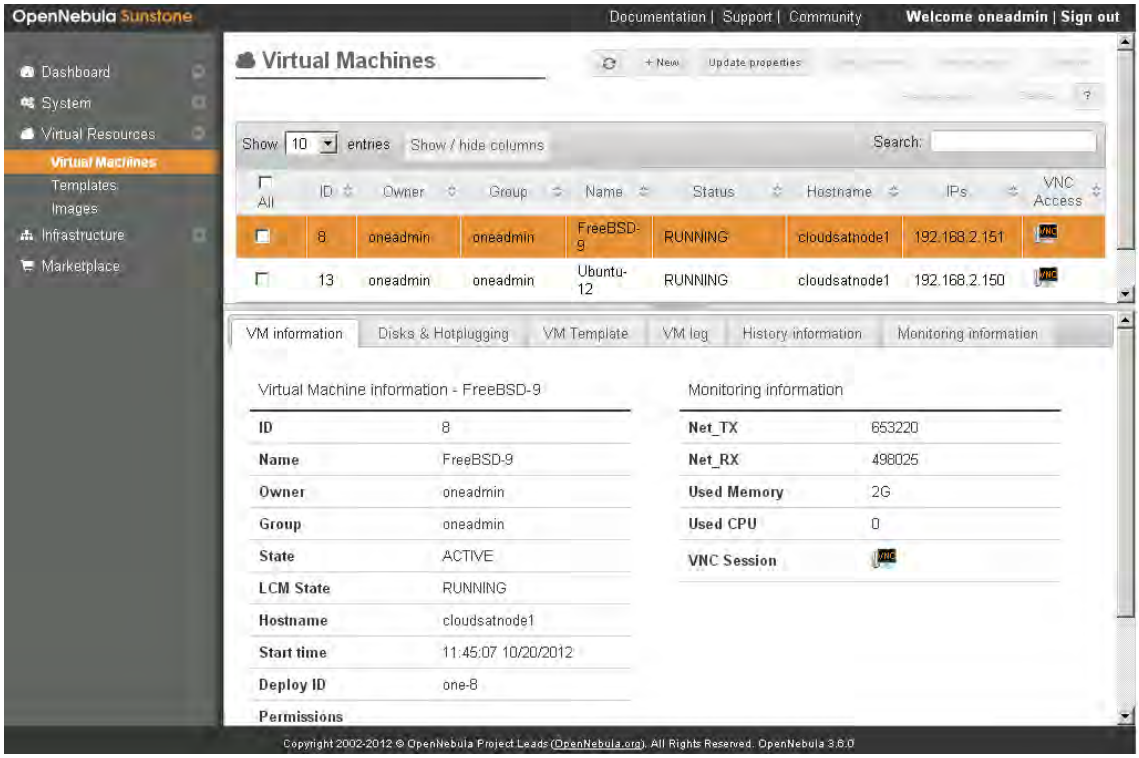


Figura B.18 - Informações adicionais sobre as máquinas virtuais

Tal como acontece com os *hosts*, os dados disponíveis não estão em tempo real, mas foram recentemente recuperados pelo *driver* de gerenciamento de informação, conforme estipulado em alguns parâmetros no arquivo de configuração do OpenNebula.

Com as implantações das máquinas virtuais feitas, a infraestrutura subjacente da nuvem CloudSatCD está concluída, e pode ser constantemente monitorada, de maneira geral, a partir do painel de controle do Sunstone (*Dashboard*), conforme demonstrado na Figura B.19, onde os principais dados gerais são exibidos. Caso sejam necessárias informações mais detalhadas, basta navegar entre os diversos menus da interface web de gerenciamento e selecionar os recursos desejados, ou recuperar essas informações via linha de comando no terminal.

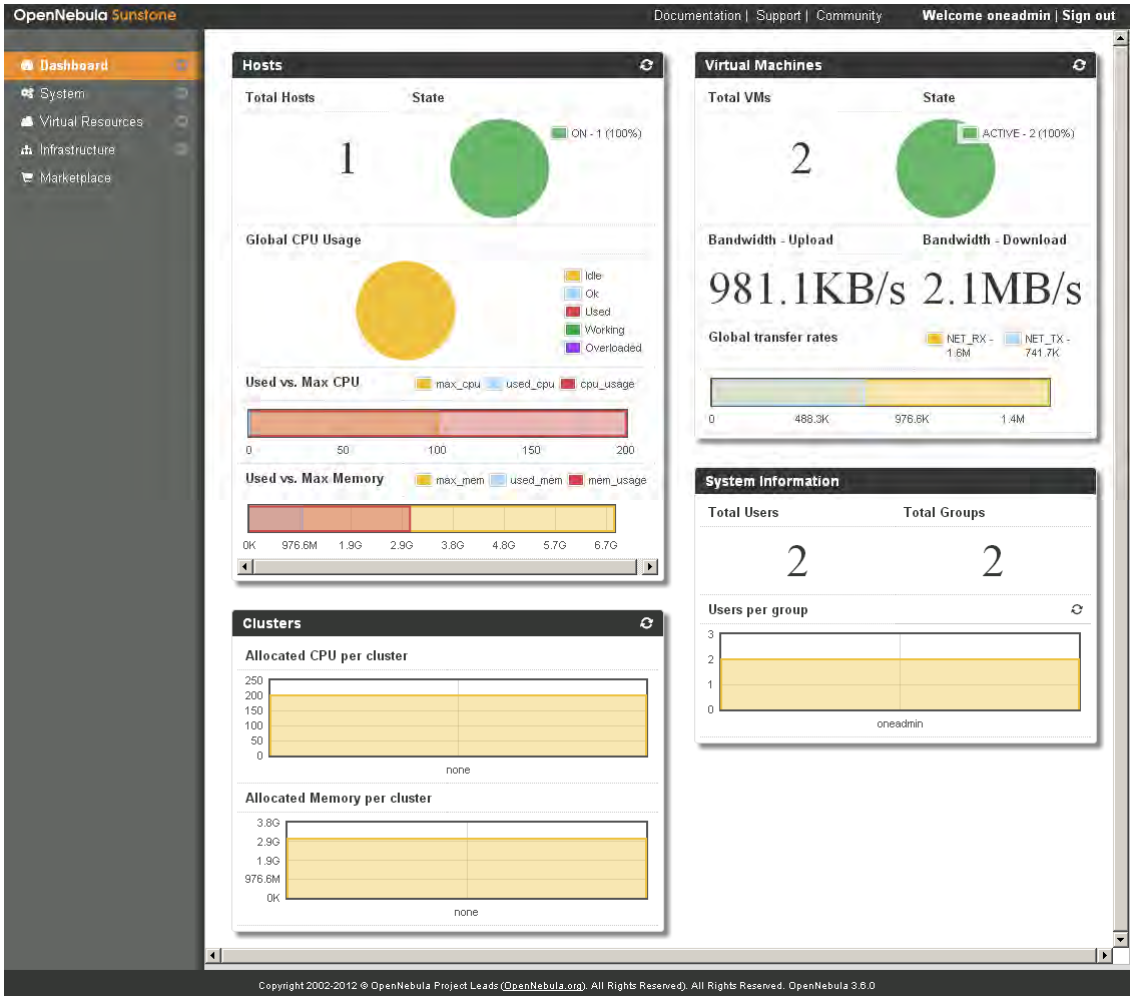


Figura B.19 - Painel de controle do OpenNebula com CloudSatCD em funcionamento

Como a proposta de CloudSatCD é prover apenas software para os usuários finais e o OpenNebula é uma aplicação voltada para IaaS (*Infrastructure as a Service*), as ferramentas responsáveis por prover o ambiente de *e-Engineering* estão disponibilizadas nas VMs criadas, ou seja, o ambiente operacional.

APÊNDICE C - AMBIENTE OPERACIONAL

O ambiente operacional é composto pelos serviços de software disponibilizados pelas máquinas virtuais na nuvem CloudSatCD. Este ambiente segue o modelo de serviços SaaS e é o responsável por prover ao usuário final o ambiente de *e-Engineering* com recursos para apoiar a fase de projeto conceitual de satélites.

A preparação dos sistemas operacionais utilizados nas VMs inclui a instalação dos pacotes básicos e sua respectiva atualização, assim como customização das configurações do ambiente de rede (interfaces) e SSH para acesso seguro entre as máquinas e administração remota via terminal.

A VM contendo o sistema operacional FreeBSD 9 é o *core* da infraestrutura e nela são executados a maioria dos serviços disponibilizados por CloudSatCD. Esse sistema operacional é o responsável por cuidar da arquitetura subjacente ao *groupware* e ao gerenciador de requisitos. A VM contendo o sistema operacional Ubuntu Server 12 é o responsável pelo gerenciamento da VPN (*Virtual Private Network*) entre a acessos externos criptografados e o *core* da infraestrutura, fazendo adicionalmente o redirecionamento de requisições HTTP via *proxy*.

C.1 Máquina Virtual 1 - Concentrador de Serviços

A configuração inicial do FreeBSD tem como ponto de partida a instalação do ambiente web. O servidor web Apache e da plataforma PHP com suporte a MySQL constituem o primeiro passo da prototipação dos ambiente de serviços de CloudSatCD. O FreeBSD dispõe de um sistema avançado de instalação de pacotes, denominado *ports*, que além de baixá-los automaticamente, permite que módulos e extensões sejam habilitados/desabilitados previamente no momento da instalação, deixando-os altamente customizáveis e fáceis de atualizar posteriormente. O *ports* também é capaz de analisar e baixar dependências entre pacotes, quando necessário. A maioria dos pacotes de

CloudSatCD foram instalados seguindo este sistema, salvo quando explicitamente seja mencionado ser proveniente do site do desenvolvedor.

A instalação do módulo SSL e geração de certificados digitais para acesso via protocolo HTTPS (*HyperText Transfer Protocol over SSL*) vêm a seguir. Com o *webserver* operacional, o passo seguinte é a instalação do banco responsável por gerir os dados das aplicações, o MySQL. Um *front-end* web para gerir a base de dados via rede, o phpMyAdmin, também foi instalado. Utilizando o sistema de *ports*, essas tarefas não exigem grande esforço.

A construção do servidor de aplicações (*appserver*) para execução de aplicativos baseados em Java é o próximo passo. O processo inicia-se pela instalação da plataforma Java, feita pelo *ports*. O Glassfish e o módulo Quercus para execução de PHP dentro da plataforma Java foram instalados a partir dos pacotes baixados diretamente do site do desenvolvedor (*source*). No caso do Glassfish é necessário que algumas variáveis de ambiente Java tenham sido exportadas previamente e que o sistema “conheça” o caminho (*path*) para elas. A instalação é concluída após ser feita a criação e inicialização do domínio que será utilizado por CloudSatCD.

O módulo Quercus necessita que suas bibliotecas (*.jar*) sejam extraídas do container (*.war*) e movidas para o domínio do Glassfish, ou feita implantação do container diretamente via interface web administrativa (*deploy*), além de alterar o arquivo de configuração do domínio, adicionando uma *servlet*, ou seja, uma classe Java utilizada para ampliar as capacidade do servidor e permitir o acesso a aplicações PHP.

As Figuras C.1 e C.2 exibem, respectivamente, um trecho do código necessário para habilitar o módulo Quercus no Glassfish e a implantação do container Quercus via interface administrativa.

```
<servlet>
    <servlet-name>Quercus Servlet</servlet-name>
    <servlet-class>com.caucho.quercus.servlet.QuercusServlet</servlet-class>
    <init-param>
        <param-name>ini-file</param-name>
        <param-value>/usr/local/etc/php.ini</param-value>
    </init-param>
</servlet>

<servlet-mapping>
    <servlet-name>Quercus Servlet</servlet-name>
    <url-pattern>*.php</url-pattern>
</servlet-mapping>
```

Figura C.1 - Trecho de código para habilitar PHP em Java com Quercus

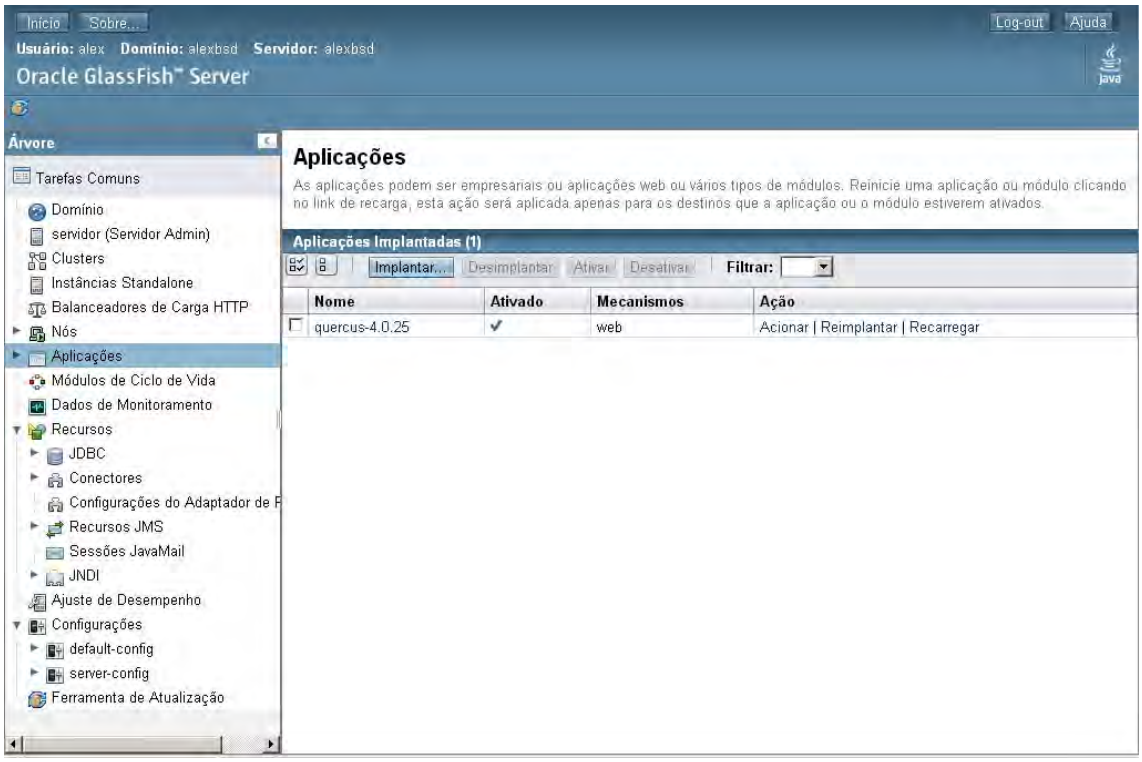


Figura C.2 - Interface administrativa do Glassfish com Quercus implantado

A fim de testar a implantação e seu funcionamento, foi criada uma página em PHP com a função *phpinfo* dentro do domínio utilizado pelo Glassfish com Quercus. A Figura C.3 ilustra um trecho da resposta do navegador web ao invocar a URL da página PHP dentro do servidor de aplicações.

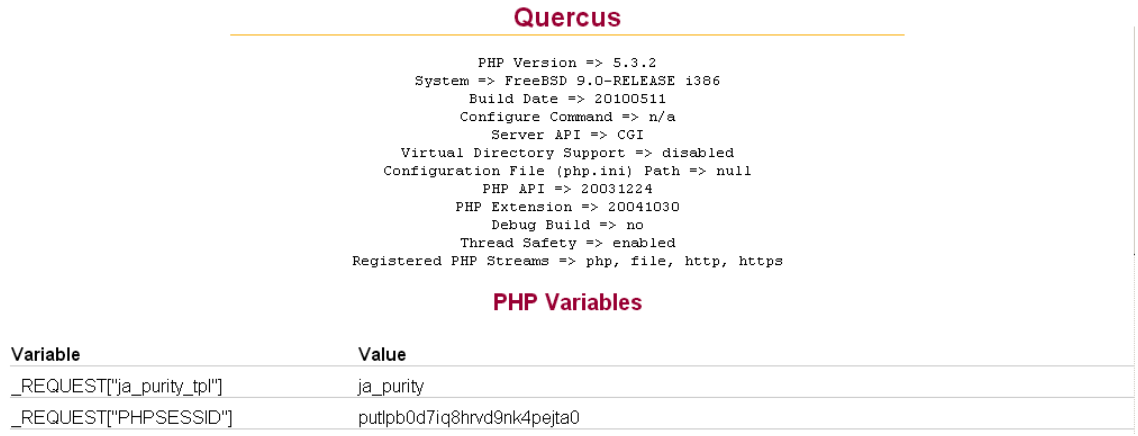


Figura C.3 - PHP executando no Glassfish através do Quercus

O Apache ServiceMix é a próxima instalação a ser feita. Visando facilitar a instalação e a preparação do ambiente, o Apache Maven, uma ferramenta para gerenciamento e automação de projetos em Java de modelo de configuração baseado no formato XML (APACHE MAVEN, 2012), foi instalado também. Baixado manualmente, a instalação do Apache ServiceMix e de seus componentes exige que variáveis de ambientes, já anteriormente exportadas, tenham sido declaradas. Esse software é uma caixa de ferramentas completa responsável habilitar a infraestrutura para implementações futuras baseadas em SOA.

Antes de iniciar a instalação do *groupware* é preciso instalar alguns softwares não inclusos que farão parte da solução, pois este deixa de forma flexível esta escolha para o administrador. Entre estes softwares estão: o Postfix, um servidor SMTP (*Simple Mail Transfer Protocol*) responsável por ser o servidor de envio de e-mail do ambiente, e o Courier-IMAP, um servidor que provê acesso às caixas de e-mail através do protocolo IMAP (*Internet Message Control Protocol*).

O Postfix é um agente de transferência de e-mails constituído por vários módulos que permitem a integração com outros softwares de maneira clara e simples (POSTFIX, 2012). O Courier-IMAP é software que permite que os

usuários tenham acesso às mensagens de suas caixas de entrada no servidor (COURIER-IMAP, 2012). Com ambos, a solução final poderá ser capaz de enviar e receber e-mails de forma integrada ao *groupware*. Adicionalmente foram incluídos o ClamAV (CLAMAV, 2012) e o SpamAssassin (SPAMASSASSIN, 2012) , antivírus e antispam, respectivamente, integrados entre si, através de um *script* (*clamav-filter.sh*) e ao par Postfix/Courier-IMAP de modo que formam um servidor de e-mail, provendo mais segurança aos usuários contra mensagens indesejadas e protegendo seus dispositivos. A Figura C.4 mostra um trecho de um dos arquivos de configuração do Postfix (*master.cf*) utilizado para integrar os softwares.

```
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (100)
# ==========================================================================
#smtp      inet  n       -       n       -       -       smtpd
smtp  inet    n       -       n       -       -       smtpd
        -o content_filter=clamav:clamav

clamav unix    -       n       n       -       -       pipe
       flags=Rq user=clamav argv=/usr/libexec/postfix/clamav-filter.sh -f ${sender}
  --  ${recipient}
```

Figura C.4 - Trecho do arquivo *master.cf* para integração no Postfix

Com a base do sistema pronta é possível iniciar a instalação e configuração da solução de *groupware*, o EGroupware, que é o centro do ambiente colaborativo proposto. Esse processo é iniciado através do sistema de *ports* do FreeBSD e concluído através da interface web autoexecutável que faz testes e checa por dependências e/ou inconsistências nas configurações do servidor web. Essa interface também pode ser responsável pela criação da base de dados automaticamente, ou a mesma pode ser criada manualmente, seja via linha de comando ou via phpMyAdmin, devendo já estar preparada para receber a base do *groupware*, ou o instalador deste não permite o avanço da instalação.

Uma vez resolvidas as questões de dependências e inconsistências nas configurações o passo seguinte é a customização do ambiente, que envolve a geração de uma nova interface de acesso (tela de *login*), onde esta será uma

das responsáveis também por autenticar usuários ao sistema de SaaS do *groupware* e redirecioná-los aos recursos que têm permissão de acesso de acordo com a ACL (*Access Control List*) em que estão inseridos, ou seja, os usuários têm acesso apenas às informações a que tem os privilégios de acesso necessários.

A Figura C.5 exibe a página inicial customizada da solução de *groupware*, *CloudSatCD Groupware Tool*, em sua interface de *login*.

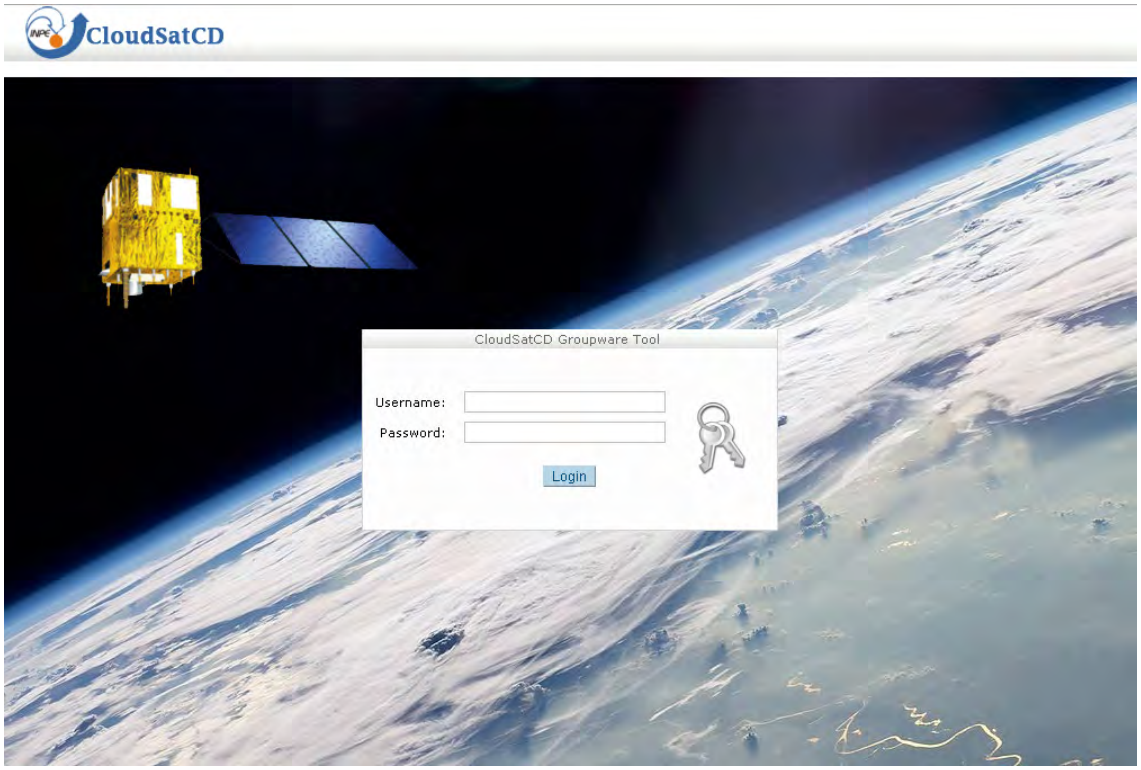


Figura C.5 - Tela de *login* de *CloudSatCD Groupware Tool*

As ACLs permitem que sejam concedidos permissões e direitos de acesso a dados de forma independente, seja por usuário específico, de forma individual, ou por grupo, de forma que o grupo inteiro tenha os mesmos privilégios. Caso não tenham sido concedidos os privilégios de acesso devidos, então não é possível ler, adicionar, editar ou excluir uma entrada no sistema.

O sistema inicializa por padrão com três grupos pré-definidos: (1) *Admins*, que constitui o grupo dos administradores do sistema com permissões totais, privilégios irrestritos e controle geral sobre todo o sistema e acesso a todas as aplicações; (2) *Default*, que constitui o grupo com permissões padrão para a maioria dos usuários, com acesso a algumas aplicações definidas pelo administrador, sendo estas passíveis de alterações; e (3) *NoGroup*, que constitui o grupo “anônimo”, onde a princípio não há privilégios de acesso às aplicações.

É possível adicionar e especificar grupos, permitindo gerir os direitos de mais de um usuário ao mesmo tempo. Como exemplo, pode-se criar um grupo para um determinado projeto ou departamento, pois provavelmente a maioria dos usuários pertencentes ao projeto ou departamento em questão usará os mesmos aplicativos, possibilitando selecionar e gerenciar as aplicações disponíveis para todo o grupo e não para indivíduos. Da mesma forma, pode-se atribuir direitos de acesso para os grupos, permitindo assim que todos os membros compartilhem arquivos entre si.

Se um usuário é membro de grupos diferentes, as aplicações serão consolidadas. Isto significa, por exemplo, que se um usuário é membro de dois grupos e ambos os grupos têm acesso ao calendário, o usuário também terá acesso ao calendário. Se um dos dois grupos não tem permissão de acesso ao calendário, o usuário ainda terá permissão de acesso, uma vez que o outro grupo permite o acessar o calendário. É possível ainda permitir direitos de acesso a aplicações nos níveis de usuário.

A criação das políticas de acesso às aplicações, áreas do sistema e configurações-padrão para os usuários é o próximo passo. O administrador pode definir configurações mandatórias e opcionais, permitindo que os usuários alterem as opcionais conforme desejado. Adicionalmente, é preciso que o administrador do sistema crie as contas dos usuários e insira-os nos grupos correspondentes para que as ACLs com as permissões pré-estabelecidas de

acordo com a política de acesso sejam aplicadas corretamente, restringindo o usuário àquilo que este pode acessar, incluindo projetos e aplicações.

Com as devidas alterações feitas, o sistema inicializa e exibe sua página inicial após a autenticação do usuário registrado, conforme demonstrado na Figura C.6. No canto superior esquerdo são exibidos links rápidos para a página inicial (*Home*) do usuário, configuração de preferências pessoais (*Preferences*), ajuda baseada no manual de utilização do software (*Manual/Help*) e para sair do sistema (*Logout*). No canto superior direito encontram-se informações gerais como o nome do usuário, data, número de usuários logados e um menu rápido para adição de itens em aplicativos. Na barra logo abaixo (Barra de Serviços) são exibidos ícones e links para os aplicativos e serviços disponíveis, de acordo com a ACL que o usuário está inserido. No centro da página são exibidos de forma rápida um Calendário e Notícias, podendo estes ser alterados por outros aplicativos, desde que permitido pelo administrador e configurado pelo usuário.

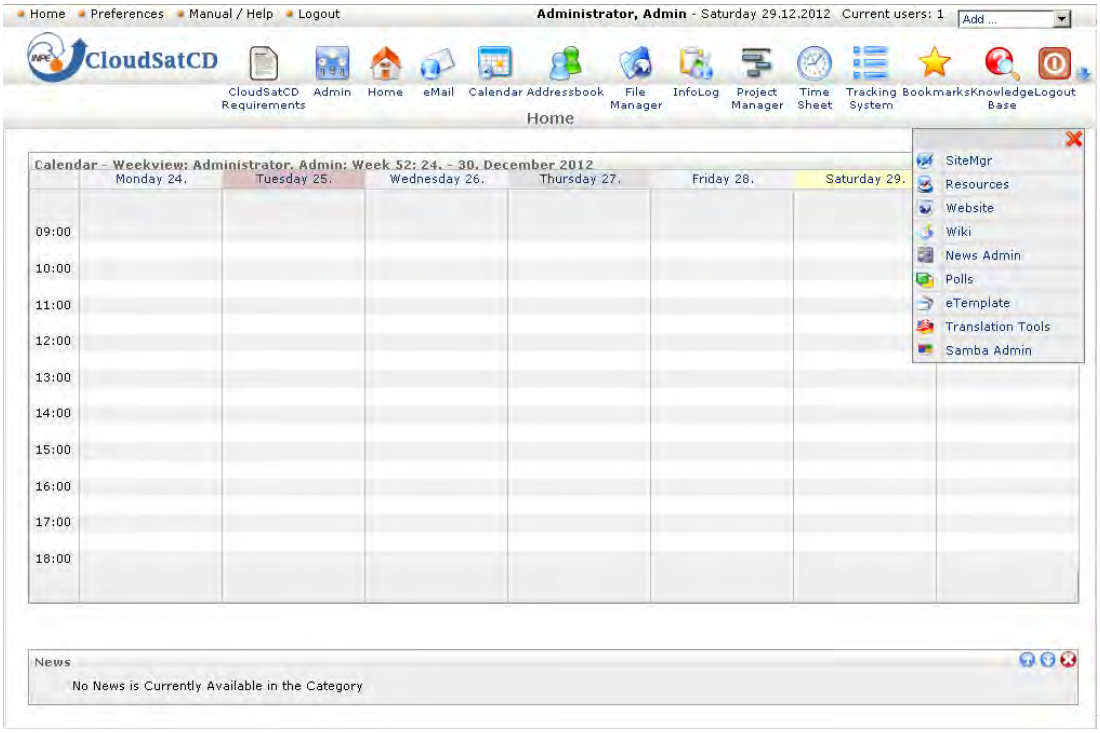


Figura C.6 - Tela inicial de *CloudSatCD Groupware Tool*

Como o usuário autenticado na Figura C.6 é o administrador do sistema, pertencente ao grupo *Admins*, esta tela inicializa com todos os serviços disponíveis, incluindo o módulo de administração, todos descritos no Capítulo 5, seção 5.2.1. Também existe um link adicional no canto esquerdo da Barra de Serviços, denominado *CloudSatCD Requirements*, que leva rapidamente o usuário ao Gerenciador de Requisitos de CloudSatCD, cuja instalação é abordada mais adiante.

A forma como o sistema exibe seus serviços pode ser alterada pelo usuário, desde que o administrador permita tal ação. Neste contexto existem três tipos de preferências que indicam o grau de customização possível: (1) Forçadas (*Forced Preferences*), em que o administrador delimita o que deseja que seja imutável, ou seja, o usuário não pode alterar as configurações convencionadas, normalmente formadas por especificações e preferências administrativas; (2) Padrão (*Default Preferences*), em que o sistema inicia com tais configurações por padrão previamente definidas, normalmente formadas por preferências gerais e comuns; e (3) Pessoais (*Your Preferences*), onde o administrador deixa totalmente a cargo do usuário alterar as configurações, normalmente formadas por preferências individuais para uso do sistema.

Concluídos os preparativos do ambiente de *groupware* inicia-se a instalação do RTH (*Requirements and Testing Hub*), o software responsável pelo gerenciamento de requisitos, gerenciamentos de testes e rastreamento de *bugs*.

A instalação, também feita via *ports*, é concluída rapidamente e não exige dependências adicionais, levando em consideração os pacotes já instalados anteriormente. A configuração mínima visa à criação da base de dados utilizada pelo software, podendo ser feita via linha de comando ou via phpMyAdmin, e sua respectiva definição no principal arquivo de configuração do programa.

A customização do ambiente, que envolve primeiramente a geração de uma nova interface de acesso (tela de *login*), visando à padronização, é o passo seguinte. Tal qual a interface do *groupware*, essa será uma das responsáveis também por autenticar usuários ao sistema de SaaS do gerenciador de requisitos e redirecioná-los aos recursos que têm permissão de acesso de acordo com a ACL (*Access Control List*) em que estão inseridos, ou seja, os usuários têm acesso apenas às informações a que tem os privilégios de acesso necessários. A Figura C.7 exibe a página inicial customizada do gerenciador de requisitos, *CloudSatCD Requirements Tool*, em sua interface de *login*, padronizada de forma a ficar semelhante à tela inicial do *groupware*.

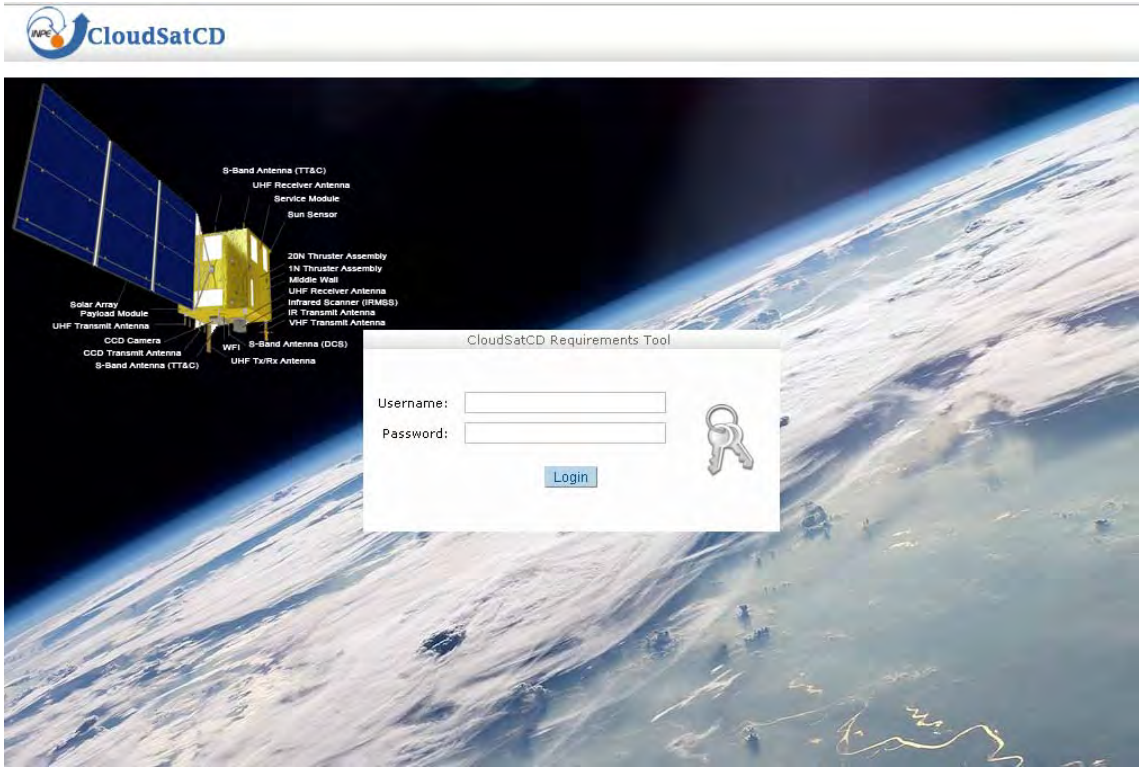


Figura C.7 - Tela de *login* de *CloudSatCD Requirements Tool*

O sistema inicializa com quatro grupos pré-definidos: (1) *RTH Administrator*, que constitui o grupo dos administradores com permissões totais, privilégios irrestritos e controle geral sobre todo o sistema, usuários e acesso a todas as aplicações; (2) *Manager*, que constitui o grupo dos gerentes de projeto com

permissões e privilégios de controle geral sobre projetos, assim como acesso a todas as aplicações, desde que definidas pelo administrador, com exceção do menu de usuários, disponível apenas para leitura; (3) *Developer*, que constitui o grupo dos desenvolvedores em que os usuários podem ter permissões padrão para acesso a aplicações definidas pelo administrador ou gerente, sendo estas passíveis de alterações, onde não há privilégios de acesso para exclusão de testes e é permitido apenas leitura para o menu de gerenciamento e para o menu de usuários; e (4) *User*, que constitui o grupo de usuários "comuns", com permissões padrão para a maioria dos usuários, em que estes podem ter acesso e direitos a algumas aplicações definidas pelo administrador ou gerente, sendo estas passíveis de alterações, onde não há privilégios de acesso para exclusão de testes e é permitido apenas leitura para o menu de gerenciamento e para o menu de usuários, tal qual o grupo *Developer*.

Diferentemente da forma como funciona o *groupware*, a ferramenta *CloudSatCD Requirements Tool* não permite a criação de novos grupos, devendo os usuários ficarem restritos a um dos grupos pré-definidos. Entretanto, é possível adicionar e especificar direitos de acesso a aplicações em nível de usuário, ou seja, de forma individual, desde que respeitadas as limitações impostas pelo grupo ao qual ele pertence.

Uma particularidade desse sistema é que o administrador por definição do próprio software é sempre parte de um dos outros três grupos, da mesma forma que um usuário, sendo de quaisquer destes grupos, pode vir a se tornar um membro do *RTH Administrator*, desde que conferida a ele essa permissão, concedida por outro administrador. Dessa forma um usuário não pode fazer parte de grupos diferentes, a menos que um deles seja o grupo dos administradores e, neste caso, prevalecem as permissões administrativas.

De maneira análoga à ferramenta de *groupware*, é preciso que o administrador do sistema crie as contas dos usuários e insira-os nos grupos correspondentes para que as ACLs com as permissões pré-definidas de acordo com a política

de acesso sejam aplicadas corretamente, restringindo o usuário àquilo que seu grupo pode acessar, incluindo projetos e aplicações.

Com o objetivo de padronizar as interfaces internas do software, praticamente todos os arquivos foram editados a fim de deixá-los com o *layout* visualmente semelhante em relação ao *layout* do *groupware* e, com as devidas alterações feitas, o sistema inicializa após a autenticação do usuário registrado.

Assim como no *groupware*, que serviu de base para a customização do *layout* utilizado no gerenciador de requisitos, no canto superior esquerdo são exibidos links rápidos para a página inicial (*Home*) do usuário, gerenciamento dos projetos (*Manage*), ajuda baseada no manual de utilização do software (*Manual/Help*) e para sair do sistema (*Logout*). No canto superior direito encontram-se informações gerais como o nome do usuário e data. Na barra logo abaixo, a Barra de Serviços, são exibidos ícones e links para os aplicativos e serviços disponíveis. No centro da página são exibidos de forma rápida Notícias com informações sobre as funções do software, definidas pelo administrador, e Resumos Gerais do Conjunto de Testes, Testes e Requisitos, de acordo com o projeto selecionado. Logo abaixo há uma caixa de seleção para alternar entre projetos e uma Barra de Links Rápidos, que contém os mesmos links da Barra de Serviços.

A Figura C.8 ilustra a tela inicial do gerenciador de requisitos customizado após *login* do usuário.

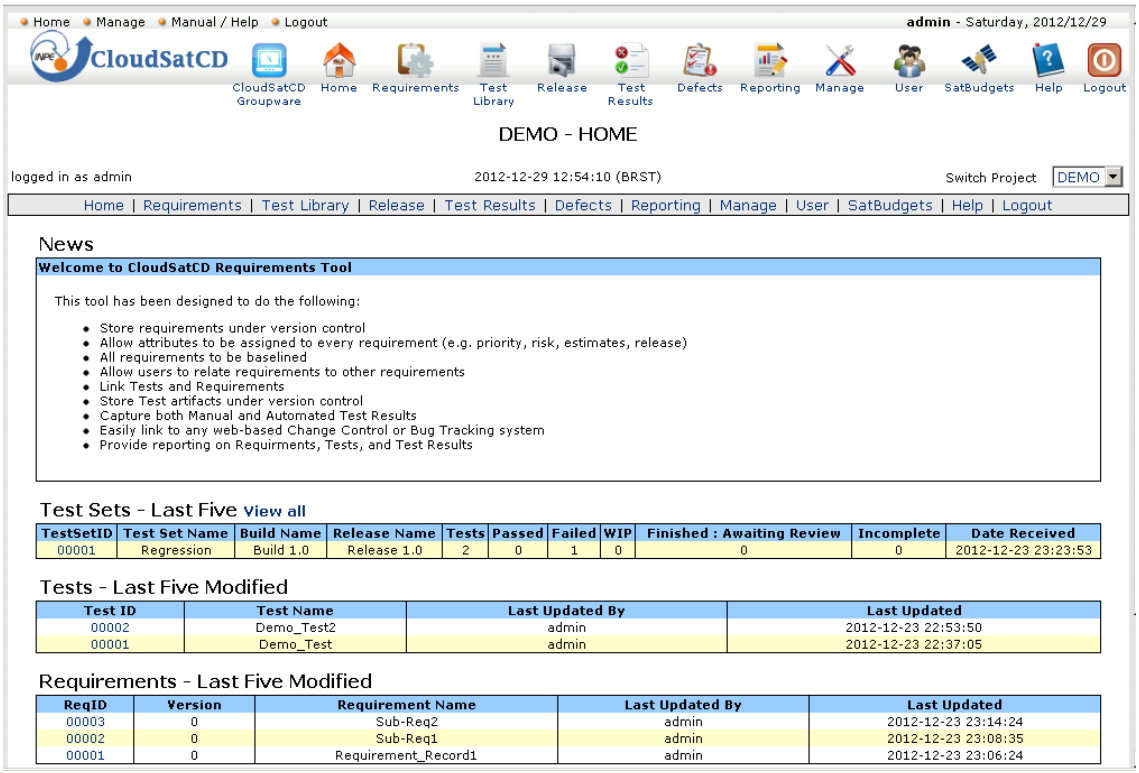


Figura C.8 - Tela inicial de *CloudSatCD Requirements Tool*

Semelhantemente ao que ocorre na interface interna do *groupware* de CloudSatCD, há um link adicional no canto esquerdo da Barra de Serviços, denominado *CloudSatCD Groupware*, que leva rapidamente o usuário àqueles serviços. As aplicações disponíveis neste software também são descritas no Capítulo 5, seção 5.2.2.

Além da padronização das interfaces foi inserido no canto direito da Barra de Serviços e da Barra de Links Rápidos um botão/link para utilização dos serviços prestados pelo SatBudgets. A descrição da integração entre esse software e CloudSatCD está descrita no Capítulo 5, seção 5.1, assim como seu funcionamento está descrito na seção 5.2.2.1.

Com a solução de serviços praticamente concluída e operacional, o passo seguinte é a disponibilização do ambiente para parceiros externos e ligação

entre CloudSatCD-SatBudgets, uma vez que a proposta dos serviços providos por CloudSatCD está inserida num contexto de rede privada/local.

C.2 Máquina Virtual 2 - Gerenciador de VPN

A outra máquina virtual que foi criada na infraestrutura, utilizando o sistema operacional Ubuntu, será utilizada como servidor de VPN (*Virtual Private Network*). Esse recurso permitirá a conexão de usuários externos ao ambiente de maneira segura, criando um túnel de dados criptografado entre o cliente e a solução, sob a Internet.

CloudSatCD provê dois tipos de VPN: (1) entre organizações parceiras e o sistema, formando uma VPN com conexão servidor-servidor; e (2) entre usuários “avulsos” e o sistema, formando uma VPN com conexão cliente-servidor. A diferença entre elas é que a VPN servidor-servidor é baseada em certificados digitais assinados pelo servidor e necessitam de instalação de um cliente, enquanto a VPN cliente-servidor é baseada no protocolo PPTP (*Point-to-Point Tunneling Protocol*), nativo em praticamente todos os dispositivos e sistemas operacionais da atualidade.

A VPN servidor-servidor é gerenciada pelo OpenVPN. Sua instalação é relativamente simples, bastando baixar os pacotes diretamente do site ou a partir do próprio sistema operacional. As configurações demandam mais trabalho principalmente em função da geração dos certificados digitais que devem ser assinados pelo servidor, juntamente com as chaves pública e privada.

A especificação da faixa de endereçamento IP que será utilizada pelos clientes e as configurações que estes obterão ao se conectar vêm a seguir, assim como outros parâmetros como o tipo de interface, porta, protocolo, localização dos certificados, entre outros. A Figura C.9 ilustra uma parte do código do principal arquivo de configuração do OpenVPN (*/etc/openvpn/server.conf*).

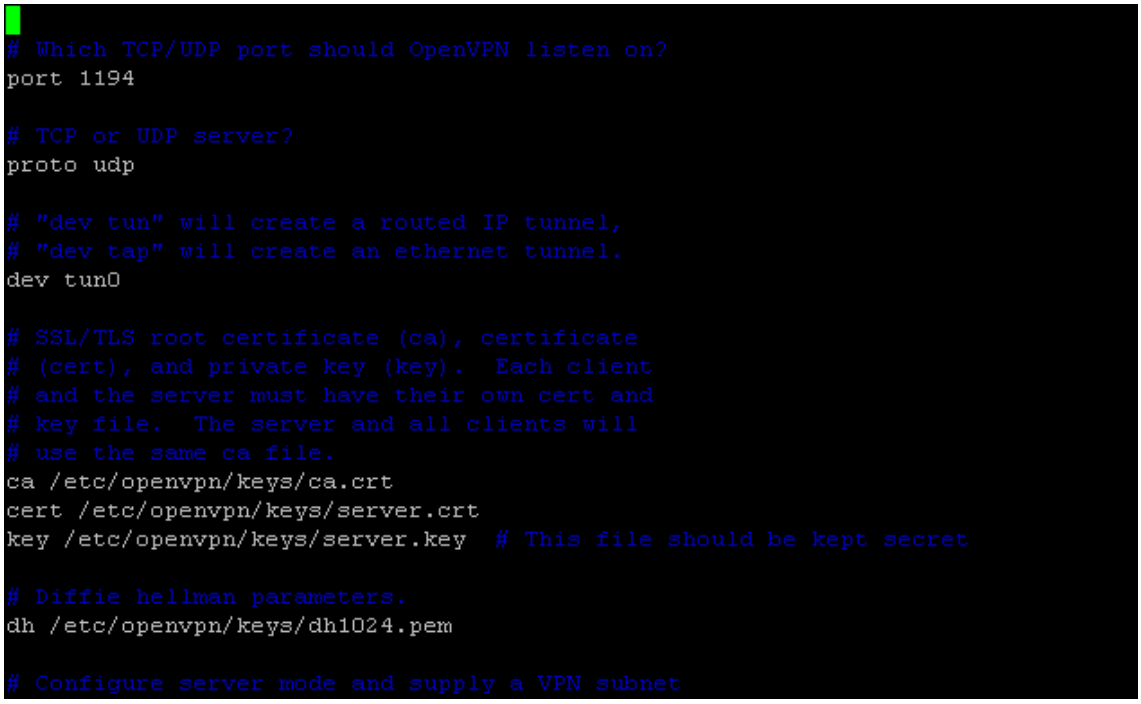


Figura C.9 - Trecho do arquivo para a implementação do OpenVPN

Após concluir as configurações o servidor de VPN pode ser iniciado e, então, é criada uma interface de rede virtual baseada na especificação incluída no arquivo de configuração do software (*tun0*), sendo este dispositivo o responsável pelo gerenciamento das conexões da VPN.

Uma vez concluído este processo e o servidor ter iniciado corretamente, é necessário adicionar regras de redirecionamento e roteamento entre as interfaces de rede do sistema, a fim de possibilitar o tráfego de pacotes; além das configurações e regras de *firewall*.

Neste tipo de solução um novo certificado deve ser gerado para cada cliente, que necessita obter previamente quatro arquivos gerados no servidor para poder se conectar: (1) certificado público da autoridade certificadora (*ca.crt*); (2) chave privada para assinatura de pacotes TLS (*ta.key*); (3) certificado público gerado para o usuário (*client.crt*); e (4) certificado privado gerado para o usuário (*client.key*). Com tais arquivos, o cliente, seja *host* ou servidor,

consegue se conectar aos serviços de CloudSatCD, desde que possua um software cliente instalado que trabalhe com OpenVPN ou um *gateway* VPN capaz de se conectar ao OpenVPN do servidor da infraestrutura; em ambos os casos sendo necessário também um arquivo de configuração com as especificações da conexão.

A fim de evitar instalações de clientes VPN em usuários “avulsos”, da própria organização, ou seja, membros da instituição com necessidade de acesso externo, CloudSatCD disponibiliza outra solução de VPN – menos segura, porém eficiente – baseada no protocolo PPTP, nativo em praticamente todos os sistemas operacionais modernos, inclusive móveis (Ex: *smartphones*).

A instalação do PopTop é relativamente simples também por ser um pacote presente na distribuição Ubuntu. As principais configurações são encontradas em três arquivos, responsáveis por: (1) configurações gerais, incluindo a faixa de endereçamento que será atribuída aos clientes (*/etc/pptpd.conf*); (2) configurações específicas do *daemon* (*/etc/ppp/pptpd-options*); e (3) dados para conexão dos clientes, como nome de usuário e senha (*/etc/ppp/chap-secrets*). A Figura C.10 exibe um trecho do arquivo de configurações avançadas da VPN (*/etc/ppp/pptpd-options*).

```
# Add an entry to this system's ARP [Address Resolution Protocol]
# table with the IP address of the peer and the Ethernet address of this
# system.  This will have the effect of making the peer appear to other
# systems to be on the local ethernet.
# (you do not need this if your PPTP server is responsible for routing
# packets to the clients -- James Cameron)
proxyarp

# Debian: do not replace the default route
nodefaultroute


# Logging

# Enable connection debugging facilities.
# (see your syslog configuration for where pppd sends to)
debug
```

Figura C.10 - Trecho do arquivo para configuração do PopTop

Já devidamente configurado o serviço de VPN pode ser iniciado. Terminado o processo, o *daemon* cria uma nova interface virtual (*ppp0*) para gerenciar a VPN e, como o roteamento e o repasse de pacotes entre interfaces já foram habilitados anteriormente, basta incluir as novas regras de redirecionamento. A cada novo usuário é preciso apenas incluir seus dados de *login* nas configurações do software e reiniciá-lo para que entrem em vigor.

Com os serviços e VPN configurados o próximo passo é a configuração do módulo de *proxy* para HTTP. CloudSatCD possui uma particularidade em relação às VPNs tradicionais e formas de acesso: a VPN estabelecida é apenas com a máquina virtual servidora de VPN, que não atua como *gateway* VPN e, por conseguinte, não gera acesso à rede. Os serviços de CloudSatCD são acessados via repasse de requisições HTTP, feitos através de um módulo *proxy*, mascarando o endereço das VMs ao mesmo tempo em que permite acesso seguro e criptografado.

O repasse de requisições HTTP é feito através da instalação do servidor web Apache e do módulo *proxy* (*proxy_http*), que responderá às solicitações encaminhando-as para o destino correto. Para que isto seja possível são necessárias também algumas regras de redirecionamento no *firewall* local. A Figura C.11 ilustra um trecho do código de configuração do módulo descrito.

```
<VirtualHost *:80>
    ProxyRequests On
    ProxyPreserveHost On
   # ProxyHTMLEnable On
    ProxyPass / http://alexbsd/
    ProxyPassReverse / http://alexbsd/
 </VirtualHost>

<VirtualHost *:8080>
    ProxyRequests On
    ProxyPreserveHost On
   # ProxyHTMLEnable On
    ProxyPass / http://alexbsd:8080/
    ProxyPassReverse / http://alexbsd:8080/
 </VirtualHost>
```

Figura C.11 - Trecho de código da implementação do *proxy* HTTP

Dessa maneira os usuários externos não têm qualquer contato com a rede, apenas com o servidor de VPN, que trata e encaminha os acessos externos para os endereços corretos.