MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
SECRETARIA DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
(Real Academia de Artilharia, Fortificação e Desenho – 1792)

Regina Helena Farias de Almeida

PROCESSO DE GERÊNCIA DE ASSETS NO DESENVOLVIMENTO DE SOFTWARE BASEADO EM REUSO: UMA ABORDAGEM NO CONTEXTO DO MINISTÉRIO DA DEFESA E DE SEUS COMANDOS SUBORDINADOS

Rio de Janeiro
2007

Livros Grátis

http://www.livrosgratis.com.br

Milhares de livros grátis para download.

INSTITUTO MILITAR DE ENGENHARIA

REGINA HELENA FARIAS DE ALMEIDA

PROCESSO DE GERÊNCIA DE *ASSETS* NO DESENVOLVIMENTO DE SOFTWARE BASEADO EM REUSO: UMA ABORDAGEM NO CONTEXTO DO MINISTÉRIO DA DEFESA E DE SEUS COMANDOS SUBORDINADOS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Engenharia de Software do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências da Computação.

Orientador: Ten Cel Ulf Bergmann – D.Sc.

Rio de Janeiro

2007

c2007

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 – Praia Vermelha
Rio de Janeiro – RJ CEP: 22290-270

Este exemplar é de propriedade do Instituto Militar de Engenharia, que poderá incluí-lo em base de dados, armazenar em computador, microfilmar ou adotar qualquer forma de arquivamento.

É permitida a menção, reprodução parcial ou integral e a transmissão entre bibliotecas deste trabalho, sem modificação de seu texto, em qualquer meio que esteja ou venha a ser fixado, para pesquisa acadêmica, comentários e citações, desde que sem finalidade comercial e que seja feita a referência bibliográfica completa.

Os conceitos expressos neste trabalho são de responsabilidade do(s) autor(es) e do(s) orientador(es).

A447 Almeida, Regina Helena Farias de
Processo de Gerência de *Assets* no Desenvolvimento de *Software* Baseado em Reuso: Uma Abordagem no Contexto do Ministério da Defesa e de seus Comandos Subordinados / Regina Helena Farias de Almeida.
– Rio de Janeiro: Instituto Militar de Engenharia, 2007.
218p.: il., graf., tab.

Dissertação (mestrado) – Instituto Militar de Engenharia - Rio de Janeiro, 2007.

1. Processo de Software. 2. Processo de Reuso. 3. Reutilização de Software. 4. Processo de Gerência de *Assets*. I. Instituto Militar de Engenharia. II. Título.
CDD 005.1

INSTITUTO MILITAR DE ENGENHARIA

REGINA HELENA FARIAS DE ALMEIDA

PROCESSO DE GERÊNCIA DE *ASSETS* NO DESENVOLVIMENTO DE SOFTWARE BASEADO EM REUSO: UMA ABORDAGEM NO CONTEXTO DO MINISTÉRIO DA DEFESA E DE SEUS COMANDOS SUBORDINADOS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Engenharia de *Software* do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências da Computação.

Orientador: Ten Cel Ulf Bergmann - D. Sc.

Aprovada em 10 de agosto de 2007 pela seguinte Banca Examinadora:

Ulf Bergman – D. Sc. do IME - Presidente

Antonio Francisco do Prado – D. Sc. da UFSCar

Ricardo Choren Noya – D. Sc. do IME

Rio de Janeiro

2007

A minha mãe por tudo que sou e a minha filha por tudo que quero ser.

AGRADECIMENTOS

Agradeço a todas as pessoas que me incentivaram, apoiaram e possibilitaram esta oportunidade de ampliar meus horizontes, especialmente:

- A minha mãe Aurora e a minha filha Thaís que sempre apoiaram e incentivaram os meus estudos;
- A minha família, a Dr[a] Vânia Novelli e aos meus amigos pessoais por terem permanecido ao meu lado em todos os momentos;
- Ao meu orientador Ulf Bergman, pelas diretrizes, orientações e estímulo, que foram fundamentais para a conclusão deste trabalho;
- A todos os professores, pelos conhecimentos transmitidos e aos funcionários do IME, em particular ao SG. Persegani e a Secretária Emília, pelo apoio e atenção;
- Aos integrantes da Banca Examinadora, pelo tempo dedicado à apreciação desta dissertação;
- Aos amigos de turma do IME pelo companheirismo e amizade;
- Aos amigos e Mestres Kleyna e Gurgel pelo auxílio nas questões referentes as suas dissertações;
- Ao Almirante Öberg e ao Capitão de Fragata Muradas pelo incentivo ao ingresso no curso de Mestrado em Engenharia de Sistemas de Computação;
- Aos amigos do CASOP pelas palavras de carinho e incentivo;
- Ao Centro de Apoio a Sistemas Operativos, em particular ao Capitão de Fragata Da Cruz, pelo total apoio e incentivo;
- Ao Centro de Análise a Sistemas Navais, em particular a amiga e Mestre Lúcia Nigro, pelo apoio técnico e administrativo; e
- Ao Instituto Militar de Engenharia (IME) por ter possibilitado a minha participação no curso de mestrado desta instituição.

"Dê-me um ponto de apoio e moverei o mundo".

ARQUIMEDES

SUMÁRIO

LISTA DE ILUSTRAÇÕES 11
LISTA DE TABELAS 13
LISTA DE SIGLAS 14

1 INTRODUÇÃO 18
1.1 Motivação 18
1.2 Objetivo 22
1.3 Estrutura do Trabalho 23

2 REVISÃO BIBLIOGRÁFICA 24
2.1 Reusable Software Assets 24
2.1.1 Conceito 24
2.1.2 Dimensões dos Assets 26
2.1.2.1 Granularidade 27
2.1.2.2 Variabilidade 27
2.1.2.3 Articulação 27
2.1.3 Meta-Modelo 28
2.1.4 Formato de Armazenamento 29
2.2 Processo de Desenvolvimento de Software Baseado em Reuso 30
2.2.1 Modelos para Definição de Processos 31
2.2.1.1 MOF 31
2.2.1.2 SPEM 32
2.2.2 Norma ISO/IEC 12207 33
2.2.2.1 Processos Primários 34
2.2.2.2 Processos de Suporte 35
2.2.2.3 Processos Organizacionais 36
2.2.3 Alterações Inseridas no Ciclo de Vida do Software com o Reuso 36
2.2.4 Processo de Reuso 38
2.2.4.1 Repositório 40
2.2.4.1.1 Vantagens do Repositório 41

2.2.4.1.2 Conteúdo do Repositório 42
2.2.4.1.3 Funções do Repositório 43
2.2.4.1.4 Implementação 45
2.2.4.1.5 Métodos de Classificação 46
2.2.4.2 Gerência da Qualidade 48
2.2.4.3 Serviços de Suporte ao Processo de Gerência de Assets 50
2.2.4.4 Métricas 52
2.2.4.4.1 Exemplo de Métricas 54
2.2.4.5 Riscos 57
2.2.4.5.1 Recomendações para o Sucesso do Reuso 60
2.3 Desenvolvimento de Software Baseado em Reuso no Contexto do Ministério da Defesa e de seus Comandos Subordinados 61
2.3.1 Ciclo de Vida do Software Baseado em Reuso 62
2.3.1.1 Visão Geral 64
2.3.2 Garantia da Qualidade 66
2.4 Conclusões 69

3 PROCESSO DE GERÊNCIA DE ASSETS NO DESENVOLVIMENTO DE SOFTWARE BASEADO EM REUSO 71
3.1 Estrutura do Processo de Gerência de Assets 72
3.2 Entradas e Saídas do Processo 74
3.3 Equipe de Gerência de Assets 74
3.4 Atores do Processo 75
3.5 Repositório de Dados 76
3.5.1 Método de Classificação 76
3.5.2 Estrutura de Armazenamento 78
3.6 Artefatos Gerados Durante a Execução do Processo 80
3.7 Atividades do Processo 81
3.7.1 Produção 83
3.7.1.1 Aprovar Assets (A1) 84
3.7.1.2 Armazenar Assets (A2) 87
3.7.1.3 Atualizar Assets (A3) 92
3.7.1.4 Notificar Consumidores 95

3.7.2 Gerência ... 99
3.7.2.1 Remover Asset (A6) ... 99
3.7.2.2 Cadastrar Usuário (A7) ... 101
3.7.2.3 Controlar Versão (A8 e A9) ... 106
3.7.2.4 Registrar Problemas (A10) ... 110
3.7.2.5 Calcular Métricas (A11) ... 112
3.7.3 Consumo ... 115
3.7.3.1 Buscar Assets (A4) ... 115
3.7.3.2 Recuperar Assets (A5) ... 118

4 ESTUDO DE CASO: ANÁLISE DO PROCESSO PROPOSTO ... 120
4.1 Planejamento do Estudo de Caso ... 120
4.1.1 Definição do Contexto do Estudo de Caso ... 120
4.1.2 Hipótese do Estudo de Caso ... 123
4.1.3 Condução do Estudo de Caso ... 124
4.1.4 Participantes do Estudo Caso ... 124
4.1.5 Objetivo do Estudo de Caso ... 124
4.2 Monitoração do Estudo de Caso ... 125
4.2.1 Cenários do Estudo de Caso ... 125
4.2.2 Cenário de Aprovação de Asset ... 126
4.2.3 Cenários de Armazenagem de Assets ... 126
4.2.4 Cenários de cadastro de Usuários ... 127
4.2.5 Cenários de Procura de Assets ... 128
4.2.6 Cenários de Recuperação de Assets ... 128
4.2.7 Cenário de Registro de Problema ... 129
4.2.8 Cenário de Atualização de Asset ... 130
4.2.9 Cenário de Elaboração de Relatório de Métricas ... 131
4.3 Avaliação do Estudo de Caso ... 131

5 CONCLUSÕES ... 134
5.1 Benefícios ... 135
5.2 Trabalhos Futuros ... 137

6 REFERÊNCIAS BIBLIOGRÁFICAS 138

7 APÊNDICES 144

7.1 APÊNDICE 1: Artefatos Gerados no Estudo de Caso 145

7.2 APÊNDICE 2: Ferramenta de Apoio ao Processo de Gerência de Assets..... 199

8 GLOSSÁRIO DE TERMOS TÉCNICOS E EXPRESSÕES USADAS..... 215

LISTA DE ILUSTRAÇÕES

FIG. 2.1 DEFINIÇÃO DE ASSET 25
FIG. 2.2 TIPOS DE ASSETS 26
FIG. 2.3 CORE RAS E PROFILES 28
FIG. 2.4 PRINCIPAIS SEÇÕES DO CORE RAS 28
FIG. 2.5 ESTRUTURA GERAL DO CORE RAS 29
FIG. 2.6 MAPEANDO RAS PARA ARQUIVOS .RAS 30
FIG. 2.7 META-MODELO: ARQUITETURA DE QUATRO CAMADAS 31
FIG 2.8 PROCESSOS DE CICLO DE VIDA DE SOFTWARE 34
FIG. 2.9 CICLO DE VIDA DO SOFTWARE BASEADO EM REUSO PELA IEEE 1517 37
FIG. 2.10 PROCEDIMENTOS PARA REUSO DE SOFTWARE. 39
FIG. 2.11 REPOSITÓRIO DE ASSETS E PROCESSOS ENVOLVIDOS 41
FIG. 2.12 MODELO DE IMPLEMENTAÇÃO DE REPOSITÓRIO. 43
FIG. 2.13 TAXONOMIA COMUM DE VOCABULÁRIOS PARA CLASSIFICAÇÃO 46
FIG. 2.14 O PROCESSO DE REUTILIZAÇÃO 51
FIG. 2.15 PROCESSOS DE PRODUÇÃO, GERÊNCIA E CONSUMO 52
FIG. 2.16 ORGANOGRAMA DO MINISTÉRIO DA DEFESA. 62
FIG. 2.17 PROCESSO DE DESENVOLVIMENTO BASEADO EM REUSO. 65
FIG. 2.18 PROCESSO DE QUALIDADE DE SOFTWARE 67
FIG. 3.1 INTERAÇÕES ENTRE PROCESSOS 73
FIG. 3.2 ESTRUTURA DE ARMAZENAMENTO BASEADA NA RAS 78
FIG. 3.3 ATIVIDADES EXECUTADAS E ARTEFATOS GERADOS NO PGA 82
FIG. 3.4 DIAGRAMA DE ATIVIDADES – APROVAR ASSETS 84
FIG. 3.5 DIAGRAMA DE ATIVIDADES – ARMAZENAR ASSETS 88
FIG. 3.6 DIAGRAMA DE ATIVIDADES – ATUALIZAR ASSETS 93
FIG. 37 DIAGRAMA DE ATIVIDADES – REMOVER ASSETS 100
FIG. 3.8 DIAGRAMA DE ATIVIDADES – CADASTRAR USUÁRIO 102
FIG. 3.9 DIAGRAMA DE ATIVIDADES – CONTROLAR VERSÃO - CADASTRAR 106
FIG. 3.10 DIAGRAMA DE ATIVIDADES – CONTROLAR VERSÃO – RECUPERAR 108
FIG. 3.11 DIAGRAMA DE ATIVIDADES – REGISTRAR PROBLEMAS 110
FIG. 3.12 DIAGRAMA DE ATIVIDADES – CONTROLE DE MÉTRICAS 112
FIG. 3.13 DIAGRAMA DE ATIVIDADES – BUSCAR ASSETS 116

FIG. 3.14 DIAGRAMA DE ATIVIDADES – RECUPERAR ASSETS 118
FIG. 7.1 USE CASE DO SUB-SISTEMA DE PRODUÇÃO 204
FIG. 7.2 USE CASE DO SUB-SISTEMA DE CONSUMO 205
FIG. 7.3 USE CASE DO SUB-SISTEMA DE GERÊNCIA 206

LISTA DE TABELAS

TAB. 2.1 NOTAÇÃO DOS ELEMENTOS SPEM. 32
TAB. 2.2 SERVIÇO DE GERÊNCIA DE COMPONENTES. 51
TAB. 2.3 OBJETIVOS PARA MÉTRICAS DE REUSO. 54
TAB. 2.4 MÉTRICAS DE REUSO. 55
TAB. 3.1 SERVIÇOS QUE O PROCESSO DE GERÊNCIA DE ASSETS FORNECE SUPORTE 82
TAB. 3.2 CHECKLIST.....86
TAB. 3.3 RELATÓRIO DA REVISÃO..... 87
TAB. 3.4 ASSETS PRODUZIDOS PELO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE.. 89
TAB. 3.5 ÓRGÃOS DA MARINHA DO BRASIL ENVOLVIDOS COM SOFTWARE 104
TAB. 3.6 ÓRGÃOS DO SISTEMA DE CIÊNCIA E TECNOLOGIA DO EXÉRCITO. 104
TAB. 3.7 ÓRGÃOS DA FORÇA AÉREA BRASILEIRA ENVOLVIDOS COM SOFTWARE 105
TAB. 4.1 CENÁRIOS DO ESTUDO DE CASO.....125
TAB. 4.2 ATIVIDADE APROVAR ASSET.....126
TAB. 4.3: ATIVIDADE ARMAZENAR ASSET.....127
TAB. 4.4: ATIVIDADE CONTROLAR VERSÕES - CADASTRAR.....127
TAB. 4.5 ATIVIDADE CADASTRAR USUÁRIO..... 127
TAB. 4.6 ATIVIDADE BUSCAR ASSET..... 128
TAB. 4.7 ATIVIDADE RECUPERAR ASSET..... 129
TAB. 4.8 ATIVIDADE CONTROLAR VERSÕES - RECUPERAR..... 129
TAB. 4.9 ATIVIDADE REGISTRAR PROBLEMA.....129
TAB. 4.10 ATIVIDADE APROVAR ASSET 130
TAB. 4.11 ATIVIDADE: ATUALIZAR ASSET..... 130
TAB. 4.12 ATIVIDADE CONTROLAR VERSÕES - CADASTRAR.....130
TAB. 7.1 LISTA DE ATORES E OBJETIVOS..... 203

LISTA DE SIGLAS

CAAML	Centro de Adestramento Almirante Marques Leão
CASNAV	Centro de Apoio a Sistemas Navais
CASE	Engenharia de Software Apoiada por Computador
CASOP	Centro de Apoio a Sistemas Operativos
CBSE	Engenharia de Software Baseada em Componentes
CCA BR	Centro de Computação de Aeronáutica de Brasília
CCA SJ	Centro de Computação de Aeronáutica de São José dos Campos
CCA RJ	Centro de Computação de Aeronáutica do Rio de Janeiro
CDS	Centro de Desenvolvimento de Sistemas
CIGE	Centro Integrado de Guerra Eletrônica
CITEX	Centro Integrado de Telemática do Exército
COMASSE	Comissão Assessora de Ciência e Tecnologia para a Defesa
CTEx	Centro Tecnológico do Exército
DAdM	Diretoria de Administração da Marinha
DepCT	Departamento de Ciência e Tecnologia
DGMM	Diretoria-Geral do Material da Marinha
DHN	Diretoria de Hidrografia e Navegação
DMCEI	Diretoria de Material de Comunicações, Eletrônica e Informática
DSAM	Diretoria de Sistemas de Armas da Marinha
DSG	Diretoria de Serviço Geográfico
DTM	Diretoria de Telecomunicações da Marinha
EMGEPRON	Empresa de Gerenciamento de Projetos Navais
ICSE	Conferência Internacional em Engenharia de Software
IEEE	*Institute of Electrical and Eletronics Engineers*
IME	Instituto Militar de Engenharia
IPD	Instituto de Pesquisa e Desenvolvimento
IPqM	Instituto de Pesquisa da Marinha
ITA	Instituto Tecnológico da Aeronáutica
MCT	Ministério da Ciência e Tecnologia
MD	Ministério da Defesa

MEM	Material de Emprego Militar
MPOG	Ministério do Orçamento, Planejamento e Gestão
MDICE	Ministérios do Desenvolvimento, Indústria e Comércio Exterior
OA	Órgão de Apoio
ODS	Órgão de Direção Setorial
OMG	*Object Management Group*
RAS	*Reusable Asset Specification*
SADLOG	Sistema de Apoio à DecisãoLogística
SCT	Secretaria de Ciência e Tecnologia
SCTEx	Sistema de Ciência e Tecnologia do Exército
SDTI	Sub-Departamento de Tecnologia da Informação
SILOMS	Sistema de Logística de Material e Suprimentos
SIGPES	Sistema de Gerências de Pessoal
STI	Secretaria de Tecnologia da Informação
TA	Treinador de Ataque

RESUMO

No mundo globalizado de hoje, a necessidade de se prover sistemas para o gerenciamento do grande volume de informações gerado a cada dia tornam imperativa a busca contínua por novas práticas e formas de desenvolvimento de software para a manipulação dessas informações. Isso está vinculado também à necessidade de se conseguir ganhos de qualidade, produtividade e redução de custos em tais desenvolvimentos, pois esses são fatores fundamentais no sucesso do negócio de empresas ligadas à tecnologia da informação. Nesse cenário, o reuso dos artefatos de software vem a exercer um papel importante na concretização desses objetivos. Mas como garantir que esses artefatos serão reutilizados?

O grande gargalo da reutilização de *software* é a falta de um processo que garanta a pratica do reuso. Por isso, a utilização de um processo de gerência dos artefatos reusáveis é um dos principais fatores que asseguram a qualidade, armazenamento, busca e recuperação dos artefatos e o sucesso do reuso dentro de uma organização.

Este trabalho está baseado nas definições da norma IEEE Std 1517 (1999) para o processo de gerência de *assets.* O processo estabelecido abrange a utilização do meta-modelo SPEM na elaboração do processo, o esquema de classificação em facetas segundo Prieto-Diaz (1993), a estrutura de armazenamento segundo a RAS (2005), documentação segundo Sametinger (1996) e o cálculo de métricas segundo Ezran (2002). Além disso, foi especificado um controle efetivo de usuários ao repositório e um controle das diferentes versões dos *assets* armazenados

Por fim, é apresentado um estudo de caso aplicando o processo de gerência aos *assets* gerados no projeto piloto Controlador Tático, desenvolvido com base no Processo de Desenvolvimento de *Software* Baseado em Reuso no Contexto do Ministério da Defesa e de seus Comandos Subordinados de GURGEL (2004) e nos artefatos gerados no desenvolvimento do Simulador do Treinador de Ataque e no Sistema de Apoio à Decisão Logística.

Este estudo de caso mostra que a utilização de um processo de gerência de *assets* proporciona o efetivo sucesso da reutilização dentro da sistemática de desenvolvimento de sistemas baseado em reuso.

ABSTRACT

In a globalized society, the need for providing systems that are able to manage huge amount of information makes it important the constant search for new means and practices for software development. Moreover, it is important to improve quality, productivity and reduce cost of such developments since these are essential factors when it comes to the success of information technology companies. Taking this scenario into account, the reuse of software artifacts is an important issue for the achievement of those goals. But the question is how to guarantee that these artifacts will be reused.

The greatest obstacle is the lack of a process, which can guarantee the reuse practice. Therefore, the use of a process of management of the reusable artifacts is one of the main factors that assure the storage, search and recovery of artifacts and also the success of the reuse within organizations.

This work is based on the definitions of the IEEE Std 1517 (1999) rule for assets management processes. The chosen process comprises the reuse of the SPEM meta-model in the preparation of the process, the classification schema according to Prieto-Diaz (1993), the RAS (2005) storage structure, the Sametinger (1996) documentation and the Ezran (2002) metric calculations. Besides that, it was established an effective users control to the repository and a control of the different versions of assets storage. It was also proposed a supporting process tool according to the services devised to support management, production and consumption specified by Apperly (2001).

At last, it is presented a case study using the assets management process elaborated in the pilot project, Tactic Controller, based on the software development according to the reuse in the context of the Ministry of Defense and their commands subordinated to GURGEL (2004) and artifacts generated from the development of the Attack Training Simulator and from the Support to Decision and Logistic System.

This case study reveals that the reuse of one Assets Management Process can assure the real success of the reuse within the context of systems development based on the reuse.

1 INTRODUÇÃO

No desenvolvimento de software busca-se alcançar os requisitos da aplicação a ser desenvolvida de maneira rápida, fácil, com baixos custos e alta qualidade. Para que possa atingir estes objetivos, a Engenharia de Software atua na aplicação de abordagens ao desenvolvimento de software, buscando a melhora da qualidade e o aumento da produtividade (KLABUNDE, 2000).

A reutilização de software é um fator que pode levar ao aumento da qualidade e da produtividade na atividade de desenvolvimento de software (SILVA, 2000) e isto se baseia na perspectiva de que o reuso de artefatos de software já desenvolvidos e depurados reduza o tempo de desenvolvimento, o tempo de testes e as possibilidades de introdução de falhas na produção de um novo *asset*.

Por definição *Asset* é um artefato ou conjunto de artefatos que tenham sido criados com um propósito explícito de ser reutilizado em outros esforços de desenvolvimento. Neste sentido, ele é a unidade que irá ser produzida, utilizada e gerenciada para permitir o reuso (RATIONAL SOFTWARE CORPORATION, 2005).

Reuso é a chave para o paradigma do grande aumento da produtividade e qualidade no desenvolvimento de software em larga escala (SILVA,1996) e também um dos principais benefícios da tecnologia de objetos e componentes, possibilitando o rápido desenvolvimento de sistemas complexos.

1.1 MOTIVAÇÃO

Em 1968, na conferência da NATO, que é considerado o marco de nascimento da engenharia de software, a palavra reutilização de software já surgia como uma das possibilidades de se superar a crise de software. O trabalho de Mcilroy (1968) apresentado nesta conferência discutia uma proposta para a especificação de uma biblioteca de

componentes[1] de software reutilizáveis e a importância de se reutilizar componentes de software em contextos similares.

Na década de 70, pouco progresso e experimentação foram realizados na área (POYTON, LANERGAN, 1979). Como ponto significante, podemos ressaltar os projetos do Departamento de Defesa dos Estados Unidos com a linguagem de programação ADA. No início dos anos 80, a reutilização de software ganhou novo fôlego com os trabalhos conduzidos por Freeman (1980), (1983), (1984), cujo enfoque principal era na especificação de métodos e mecanismos para a operacionalização da reutilização de software, representação de componentes reutilizáveis e na organização de repositórios de componentes.

Ao longo dos anos 80 e 90, era esperada uma confirmação da reutilização de software como um grande avanço no aumento de produtividade no desenvolvimento de software.

Trabalhos importantes foram desenvolvidos, com destaque para os trabalhos de Neighbours (1981), que foi o pioneiro na definição do termo análise de domínio, Arango (1988), cuja tese de doutorado foi uma contribuição importante para a sistematização do processo de criação de componentes reutilizáveis, através da especificação do que ele chamava de sistematização da análise de domínio, criando assim o termo engenharia de domínio e também Prieto-Diaz (1987), que, através da sua classificação por facetas, apresentou uma abordagem interessante e inovadora para a busca e seleção dos componentes reutilizáveis. Apesar destes trabalhos pioneiros e a conseqüente evolução dos mesmos, em termos práticos, houve uma evolução pequena na utilização efetiva da reutilização de software (KRUEGER, 1992). Apesar das diversas pesquisas conduzidas na área, surgindo conferências importantes e específicas para tratar sobre tema, além de vários *workshops* e trabalhos apresentados em outros foros, ainda não se possui um consenso a respeito de qual a melhor abordagem a ser adotada.

Assim, passados alguns anos desde o trabalho pioneiro de Mcilroy (1968), vemos que a reutilização de software ainda não é uma prática comum entre os desenvolvedores. Mesmo com as novas tecnologias de desenvolvimento de software como a orientação a objetos, componentização, objetos distribuídos, tecnologias de recuperação de informação baseadas em técnicas inteligentes, a reutilização sistemática e organizada de software não é uma prática tão corrente.

[1] Componentes de software, segundo Hopkins (2000), é definido como um pacote físico de executável de software com uma interface bem definida, pública e com forte tendência à reutilização.

Nossa suposição para esta questão é de que a reutilização de software até hoje não se tornou uma prática comum no desenvolvimento de software, dentre outros aspectos, pela falta de técnicas padronizadas e mais direcionadas para a construção, busca e recuperação de *assets* de forma que o reutilizador possa entender, selecionar e adaptar de forma facilitada o *asset* ao seu contexto.

Durante as fases do processo de desenvolvimento de software, uma grande quantidade de esforço, em geral, é desperdiçada. Pode-se observar que alguns artefatos de software gerados, durante as fases do processo de desenvolvimento (especificações, arquitetura, projeto, ou parte de código), são similares ou até mesmo iguais, a outros confeccionados em outros projetos.

Artefato de software é um termo utilizado na nomenclatura do Processo Unificado e UML (Unified Modeling Language) (JACOBSON; RUMBAUGH; BOOCH, 1998). Neste contexto, artefato de software é qualquer produto gerado durante o ciclo de vida de desenvolvimento de um sistema, que pode ser: um documento de requisitos, modelos, arquivos de código fonte, uma documentação, um modelo conceitual, uma linguagem de domínio, uma regra de transformação, uma arquitetura de domínios, a especificação de um componente, casos de testes ou *scripts.*

Estes artefatos de *software* desenvolvidos para certos projetos, podem ser utilizados em novos projetos, com o intuito de diminuir o retrabalho realizado durante o processo de desenvolvimento de um produto de *software*.

Em uma pesquisa realizada em 29 organizações por Frakes e Fox (1993) apud Lim (1998), foi apresentado o percentual de subprodutos de *software* gerados durante o processo de desenvolvimento de *software*, e que consistiam de partes reutilizáveis. Os dados deste estudo indicaram que código e a documentação do usuário eram os bens que, na média, tinham o mais alto percentual de partes reutilizáveis nestas organizações pesquisadas.

Outra pesquisa realizada por Morisio, Tully e Ezran (MORISIO, 2002), foram analisadas 24 empresas européias que tiveram uma experiência na implantação de um programa de reutilização de *software*. Um terço aproximadamente destas iniciativas falhou, por não terem sido adotados processos específicos ao reuso, ou pelo fato dos processos convencionais, até então em vigor, não terem sido adaptados aos programas de reutilização de *assets* (além de fatores humanos responsáveis pelos fracassos). A crença de que o paradigma OO e a configuração aleatória de um repositório sejam suficientes, para o sucesso de um programa de reuso é apontado por Morisio como um dos erros mais comuns cometidos por gerentes e

líderes de projeto. É necessário, então, haver um processo regendo a reutilização de *assets* via um repositório.

Vários pesquisadores apontam a necessidade de ferramentas que forneçam suporte ao processo de reutilização de software [(MCCLURE, 1997), (LIM, 1998), (IEEE Std 1517-1999), (GUO; LUQI, 2000), (PRESSMAN, 2000), (GRISS, 2001)]. Um repositório de bens de software é uma destas ferramentas, e a principal razão para a existência de um repositório é tornar disponíveis os *assets*, permitindo a divulgação e o compartilhamento e possibilitando o acesso dos desenvolvedores que buscam candidatos potenciais à reutilização em seus sistemas.

Para que a utilização do repositório de *assets* seja eficiente, é necessário estabelecer um processo de desenvolvimento adequado. Basicamente, os processos de reutilização de software apresentados na literatura possuem três atividades comuns que são as seguintes: seleção do *asset* para reutilização, adaptação para o objetivo da aplicação e integração ao produto de software que está em desenvolvimento. (BASILI; BRIAND; MELO, 1996)

No entanto, para obter vantagens reais com a reutilização de software, é necessário disponibilizar os mecanismos para auxiliar o reutilizador nas atividades de identificação, seleção e entendimento do *asset*.

Além disso, devem ser disponibilizados recursos não só para o consumo de *assets*, mas também para a produção de *assets* e para a gerencia do repositório e das informações contidas nele [(MCCLURE, 1997), (LIM, 1998), (APPERLY, 2001)].

Desta forma, um recurso importante para apoiar o processo da reutilização de software é a existência de um processo que gerencie as atividades e que auxilie na gerencia, busca e recuperação dos *assets* armazenados em um repositório, possibilitando, desta forma, o compartilhamento e a difusão destes bens reutilizáveis.

Para isso, é importante estudar e analisar os mecanismos de classificação de informação dos *assets* em um repositório, as estratégias de busca e recuperação dos mesmos e o conjunto de informação necessária para representar e documentar o *asset* dentro do repositório.

Somente com um processo de gerência dos *assets* é que poderemos assegurar que os *assets* produzidos durante a fase de desenvolvimento de um sistema serão reutilizados na elaboração de um novo projeto.

1.2 OBJETIVO

O objetivo dessa dissertação é propor um modelo de processo de gerência de *assets* a ser utilizado no contexto do Processo de Desenvolvimento de *Software* Baseado em Reuso no Ministério da Defesa (MD) e de seus Comandos Subordinados proposto por Gurgel (2004).

- Segundo a ISO/IEC 12207 (1995) o propósito do Processo de Gerência de *Assets* é gerenciar a vida dos *assets* reusáveis desde a sua concepção até à sua retirada do sistema.

Para que a gerência de *assets* ocorra de modo efetivo o processo proposto deve assegurar que:

- Após o término do desenvolvimento de um sistema, o artefato ou conjunto de artefatos que desejamos reusar, deverá estar acompanhado de uma documentação contendo informações pertinentes e seguindo um formato pré-estabelecido;
- O artefato ou conjunto de artefatos deverá passar por um rígido controle de qualidade que possa assegurar a sua reutilização. Após essa etapa, poderemos designá-lo de *asset*, que por definição é um artefato reutilizável;
- Os *assets* serão armazenados em um repositório. Para podermos armazenar um *asset* ele deverá seguir um esquema de classificação;
- Os *assets* deverão ser catalogados;
- Haverá controle de versões;
- Haverá cálculo de métricas do reuso;
- Haverá controle de usuários;
- Haverá registro de problemas; e
- Para finalizar o processo serão definidas regras para a busca e recuperação de um *asset*. Essas regras serão as formas que poderemos selecionar um *asset*, definindo como ocorrerá a sua busca, de maneira que seja possível localizar um *asset* candidato ao reuso e recuperá-lo de acordo com as necessidades do usuário.

1.3 ESTRUTURA DO TRABALHO

Esta dissertação é composta de quatro capítulos, além desta introdução.

No capítulo 2, **Revisão Bibliográfica**, são apresentados os principais aspectos sobre *Reusable Asset Specification* (RAS). Para isso, são apresentados os conceitos e as definições sobre este assunto. Os tópicos sobre a Norma ISO/IEC 12207 e a sua extensão, IEE 1517 (1999), são abordados os processos necessários para o apoiar o desenvolvimento de s*oftware* baseado em reuso. Já o tópico sobre Processo de Reuso apresenta os conceitos e as necessidades do processo de reutilização de *software* dentro da sistemática do reuso. O final do capítulo aborda o Desenvolvimento de Software Baseado em Reuso no Contexto do Ministério da Defesa e de seus Comandos Subordinados que apresenta o desenvolvimento de software baseado em resuso proposto por GURGEL (2004) e o processo de qualidade de software prposto por MOORE (2004) dentro do contexto do Ministério da Defesa.

O capítulo 3 apresenta o **Processo de Gerência de *Assets***, detalhando suas etapas, atividades envolvidas, responsabilidades e produtos gerados.

O capítulo 4 apresenta o **Estudo de Caso**, com o objetivo de verificar a adequabilidade da proposta apresentada no capítulo 3.

Finalmente, o capítulo 5 apresenta as **Conclusões** deste trabalho, bem como as perspectivas para pesquisas futuras.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta os resultados da revisão bibliográfica realizada, com os conceitos e estudos que foram utilizados na definição da proposta. Serão apresentados os principais aspectos relativos à reutilização de software, aos processos de software e a gerência de *assets.*

No decorrer da revisão bibliográfica são encontrados os termos "*assets*" e "componentes" para indicar "artefatos reutilizáveis". Cabe ressaltar, que um componente reusável nada mais é do que um dos tipos de *assets* gerados no desenvolvimento baseado em reuso.

2.1 REUSABLE SOFTWARE ASSETS

2.1.1 CONCEITO

O conceito de *Reusable Software Assets* é centrado no reuso de software. Podemos considerar "reuso" como a maneira de capturar e explorar a experiência do negócio. Sendo assim, os *assets* encapsulam o conhecimento do negócio e é um bem de valor elevado para uma companhia (EZRAN, 2002).

Segundo a RAS (2005), *asset* é um artefato ou conjunto de artefatos que tenham sido criados com um propósito explícito de ser reutilizado em outros esforços de desenvolvimento. Neste sentido, ele é a unidade que irá ser produzida, utilizada e gerenciada para permitir o reuso. A **FIG. 2.1** ilustra a definição de um *asset.* Eles possuem regras para o seu uso que são as descrições de como eles podem ser utilizados.

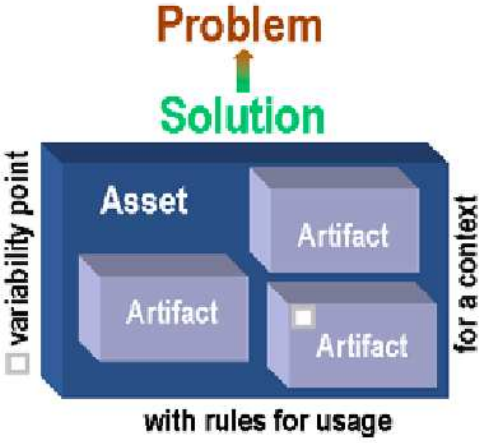


FIG. 2.1: Definição de *Asset*

Como definido anteriormente, artefato é qualquer produto gerado durante o ciclo de vida de desenvolvimento de um sistema, que pode ser: um documento de requisitos, modelos, arquivos de código fonte,, uma documentação, um modelo conceitual, uma linguagem de domínio, uma regra de transformação, uma arquitetura de domínios, a especificação de um componente, casos de testes ou *scripts*.

Segundo Gurgel (2004), independentemente do tipo de tecnologia adotada, reuso é o desenvolvimento de um sistema de software com o uso de elementos criados em contextos diferentes.

No entanto, apenas o desenvolvimento ou a aquisição de artefatos passíveis de serem reusados não garante que se faça o reuso. Para que tenhamos um reuso sistemático, devemos definir uma forma de documentação e representação que, diante de uma nova necessidade de desenvolvimento, facilite a busca, entendimento e o uso destes artefatos já desenvolvidos. Logo, o *asset* deve, além de apresentar o artefato, ou artefatos, viabilizar a sua gerência, ou seja, o seu armazenamento, controle de versões, busca, entendimento e aplicação.

A *Rational Software* e o *Reusable Asset Specification Consortium* definiram um conjunto de regras base para a categorização dos *assets* e sua representação de forma a viabilizar esta gerência.

A especificação RAS (*Reusable Asset Specification*), proposta pela OMG, define uma forma padrão de empacotamento de *assets*. Um ativo reutilizável de software (*asset*) pode ser considerado, de maneira genérica, uma coleção coesa de artefatos que solucionam um problema específico ou um conjunto de problemas, encontrados no decorrer do ciclo de desenvolvimento de software. Dentro do contexto do RAS, o pacote de um ativo reutilizável

de software é um conjunto de arquivos, que implementam a solução, e uma estrutura de meta-informação que define e descreve o ativo como um todo. A especificação da OMG define três tipos de *assets*:

- **Componente** - Tipo de *asset* que adere a uma interface bem documentada e que, tipicamente, oculta sua implementação (*black box*), tornando-a implícita;
- ***Framework*** - Empregado para solucionar um conjunto maior de problemas. Apresenta-se, normalmente, como uma coleção de ativos individuais ou *middlewares* sobre os quais aplicações são construídas. Vide definição de Gamma (GAMMA, 1995); e
- **Padrão** - Uma abstração da estrutura e do comportamento de um sistema ou parte de um sistema. Um padrão pode ser aplicado a um sistema, no sentido de orientar a estrutura e a dinâmica dos elementos do mesmo. Vide definição de Gamma (GAMMA, 1995).

2.1.2 DIMENSÕES DOS *ASSETS*

Existem três dimensões chaves que descrevem os *assets* (ativos reutilizáveis). Eles podem variar segundo essas três dimensões: granularidade, variabilidade e articulação (RAS, 2005), como mostra a **FIG. 2.2.**

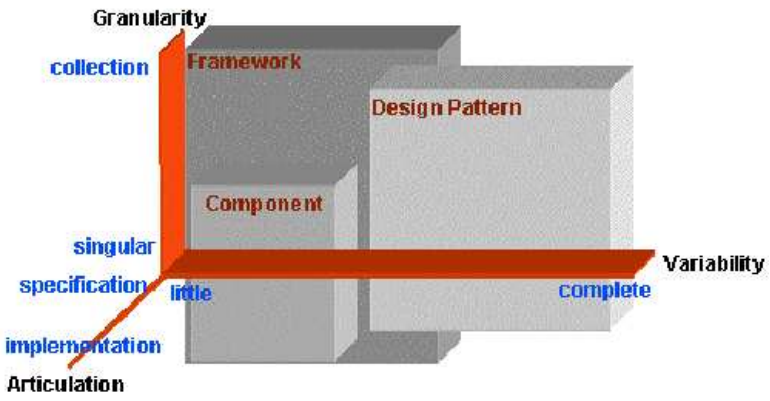


FIG. 2.2: Tipos de *Assets*

2.1.2.1 GRANULARIDADE

A granularidade descreve quantos problemas particulares ou soluções alternativas o *asset* resolve. Os *assets* mais simples oferecem somente uma solução a um problema bem definido. Conforme a granularidade vai aumentando o *asset* dirige-se a múltiplos problemas, e/ou pode oferecer várias soluções para aquele problema. Podemos ter nesta linha desde mecanismos simples com apenas uma funcionalidade até complexos *frameworks*. No geral com o aumento da granularidade vem um aumento no tamanho e na complexidade de um *asset*.

2.1.2.2 VARIABILIDADE

A variabilidade de um *asset* é outra propriedade chave de um ativo. Num extremo o *asset* pode ser invariável, isto é, não pode ser alterado. Freqüentemente estes *assets* são chamados de *black-box*, e sua implementação não pode ser vista e nem modificada. No extremo oposto estão os *assets* denominados de *white-box* (caixa-branca). Estes recursos são criados com a expectativa que os consumidores do *asset* editarão e alterarão sua execução. Os *assets white-box* também incluem artefatos de desenvolvimento tais como: requisitos, modelos, componente concreto, etc...

2.1.2.3 ARTICULAÇÃO

A articulação descreve o grau de integridade dos *assets* em fornecer a solução. Os *assets* cujos artefatos especificam uma solução mas não fornecem a solução têm um grau baixo de articulação. Os *assets* cujos artefatos especificam e executam uma solução tem um grau maior de articulação. Nesta dimensão um *asset* pode ir desde uma definição de requisitos, passando pela análise de domínio e especificação de projeto, até a implementação.

2.1.3 META-MODELO

O meta-modelo RAS é definido em duas grandes categorias, o *Core* (núcleo) RAS e os Profiles (perfis). O *Core* RAS representa os elementos fundamentais da especificação de um *asset*. Os Profiles descrevem extensões destes elementos fundamentais. Um perfil não deve alterar a definição ou a semântica dos elementos do núcleo. O *Core* RAS não é instanciável, portanto, um *asset* deve ser de um determinado Profile. O núcleo do meta-modelo RAS e seus perfis padrões, definidos pela especificação, estão ilustrados na **FIG. 2.3** e **FIG. 2.4**.

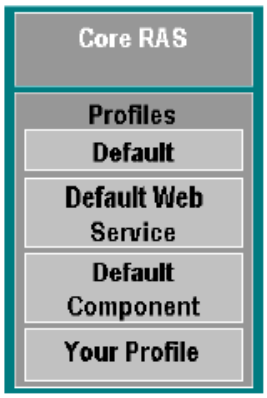


FIG. 2.3: *Core RAS e Profiles*

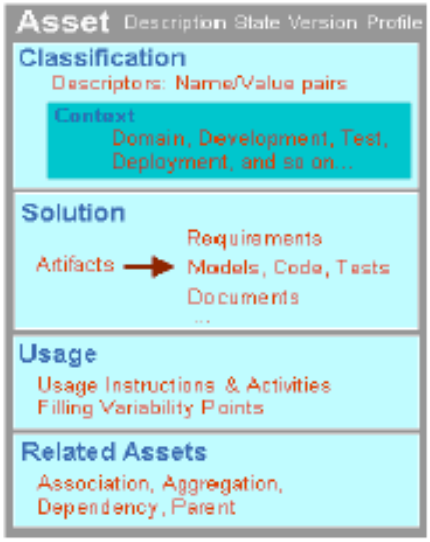


FIG. 2.4: Principais seções do Core RAS

Core RAS define a estrutura básica que todo *asset* deve conter. A **FIG. 2.5** ilustra sem maiores detalhes esta estrutura básica composta pelos seguintes elementos:

- ***Classification*** (classificação). Lista um conjunto de descritores para classificação dos *assets*, bem como uma descrição do contexto para o qual o *asset* é relevante;
- ***Solution*** (solução). Descreve os artefatos que fazem parte do *asset*;
- ***Usage*** (utilização). Contém as regras para instalação, personalização, e utilização do *asset*;
- ***Related Assets*** (*assets* relacionados). Descreve a relação deste *asset* com outros; e
- ***Profile*** (perfil). Representa o perfil do *asset*, podendo ser um dos perfis padrões do RAS, uma extensão dos mesmos ou do próprio *Core* RAS.

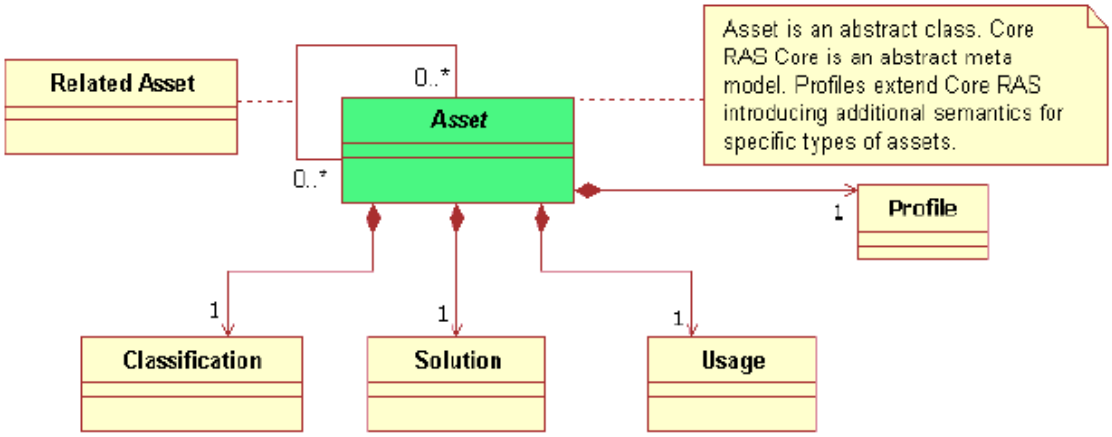


FIG. 2.5: Estrutura Geral do *Core RAS*

2.1.4 FORMATO DE ARMAZENAMENTO

A especificação RAS define um formato de armazenamento baseado em meta-informações, a respeito de um determinado *asset*, num arquivo manifesto (*rasset.xml*). Este arquivo contém toda a relação de artefatos que compõem o *asset*, bem como a localização destes artefatos. O *rasset.xml* nada mais é do que uma instância XML, definida segundo um esquema (.xsd, por exemplo) refletindo algum modelo de perfil RAS, como os que foram apresentados na seção anterior. A **FIG. 2.6** mostra o encadeamento de conceitos: o meta-modelo definido por um perfil RAS é mapeado num esquema XML, que é instanciado em

um *rasset.xml*. É importante se notar a presença obrigatória de um arquivo XSD, que contenha um esquema de perfil (ex. *defaultProfile.xsd*, *defaultComponentProfile.xsd*, *DefaultWebServiceProfile.xsd* etc.), definindo a estrutura do arquivo manifesto de acordo com o perfil em questão.

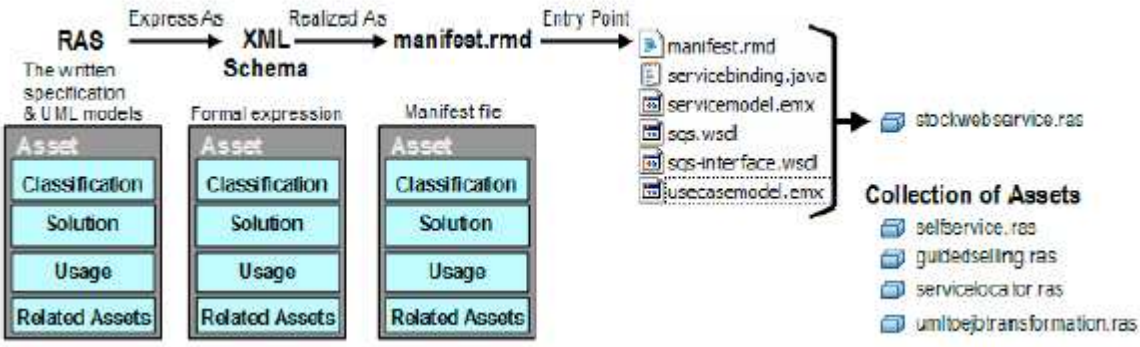


FIG. 2.6: Mapeando RAS para arquivos .ras

2.2 PROCESSO DE DESENVOLVIMENTO DE SOFTWARE BASEADO EM REUSO

Processo é um conjunto de tarefas definidas para se criar um produto. Os processos são compostos, isto é, podem ser sub-divididos em sub-processos até chegar a tarefas atômicas, também chamada de atividades, realizadas por uma pessoa num local específico. Os processos de software se referem a todas as tarefas necessárias para se produzir e gerenciar o software. Os processos de reuso correspondem ao sub-conjunto de tarefas necessárias ao reuso de software dentro da empresa (EZRAN, 2002) e a introdução de um processo de reuso tem se mostrado como o maior desafio na disseminação do conceito de reuso de componentes (BIGGS, 2004).

Os processos se fazem necessários a fim de garantir que atividades similares sejam realizadas da mesma maneira por equipes diferentes em épocas diferentes.

Vários tipos de dados são integrados em um modelo de processo para indicar quem, quando, onde, como e porque os passos são realizados. Para representar um modelo de processo é freqüentemente adotada uma *Process Modelling Language* (PML), a qual deve oferecer recursos para descrever e manipular os passos do processo.

2.2.1 MODELOS PARA DEFINIÇÃO DE PROCESSOS

2.2.1.1 MOF

Segundo FARIA (2005), o modelo MOF (MOF 2.0) da OMG define uma arquitetura para definição e descrição de meta-dados em quatro camadas. Este modelo em camadas, para representar meta-informações em quatro níveis, não é na verdade exclusividade do MOF. Sua estrutura é apresentada na **FIG. 2.7**.

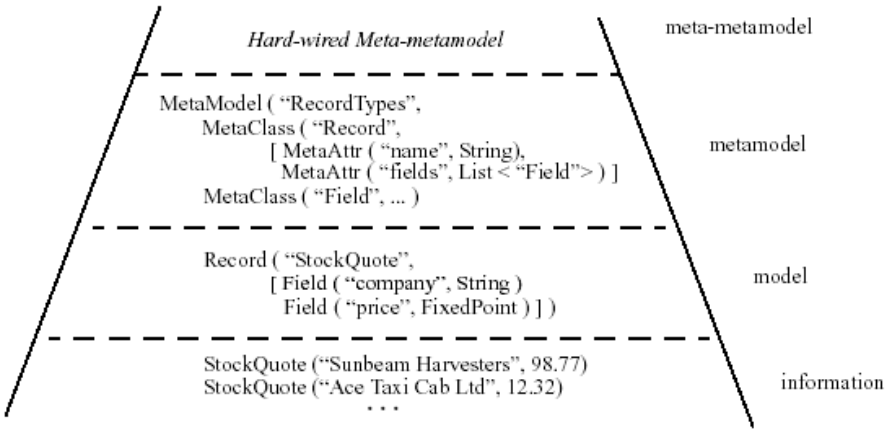


FIG. 2.7: Meta-Modelo: Arquitetura de Quatro Camadas

Os níveis podem ser interpretados conforme as descrições a seguir:

- **Informação** (M0) - Os objetos deste nível menos abstrato compõem a informação que se quer representar. Esta informação é tipicamente denominada de dados;
- **Modelo** (M1) - Este nível compreende os meta-dados que descrevem a informação do nível M0;
- **Meta-modelo** (M2) - Esta camada define a estrutura e a semântica do meta-dado do nível M1. Pode ser entendida como uma linguagem, na medida em que define uma gramática para notação de modelos; e

- **Meta-meta-modelo** (M3) - Camada mais abstrata que descreve as regras sintáticas e semânticas dos dados descritos em M2 (meta-meta-dados). Uma linguagem para definir diferentes tipos de meta-modelos.

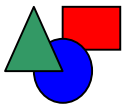
2.2.1.2 SPEM

O *Software Process Engineering Metamodel*, (SPEM 2.2) é um meta-modelo que define um padrão para especificação de processos baseado em subconjuntos do meta-modelo UML, para descrever um processo concreto de desenvolvimento de software ou uma família de processos relacionados de desenvolvimento de software. Uma ferramenta baseada no SPEM seria uma ferramenta para autoria e personalização de processos. Além de descrever um meta-modelo de processo, o SPEM é definido também como um perfil UML, isto é um pacote de elementos de modelagem personalizados para um propósito ou domínio específico, estendendo o meta-modelo através de estereótipos, propriedades e restrições.

Sendo o SPEM o meta-modelo para especificação base de qualquer processo, é necessário definir os seus constituintes básicos. Os princípios base do SPEM surgem da idéia que o processo é uma colaboração entre entidades abstratas ativas designadas por papéis de processos (meta-classe *process role*) que realizam atividades (meta-classe *activity*) em entidades tangíveis e concretas chamadas produtos do trabalho (meta-classe *work products*).

O SPEM define estereótipos para seus elementos de modelagem como alguns descritos na **TAB. 2.1**:

TAB. 2.1: Notação dos elementos SPEM.

ESTERIÓTIPO	DEFINIÇÃO	NOTAÇÃO
WorkProduct	é uma descrição de algo que contém informação ou é uma entidade física produzida ou usada por atividades do processo.. Ex: documento, código fonte, modelos, planos, etc.	

Activity	descreve uma parte do trabalho realizado por um *ProcessRole*: suas tarefas, operações e ações executadas por um papel ou de que forma o papel deve auxiliar	
ProcessRole	define responsabilidades em relação a WorkProducts específicos e define papéis que realizam e auxiliam atividades específicas.	
ProcessPackage	notação especial para pacotes no contexto SPEM	
Document	notação específica para diferentes tipos de *WorkProducts*	

2.2.2 NORMA ISO/IEC 12207

A norma internacional ISO/IEC 12207, fornece uma estrutura comum de processos para ser utilizada como referência no desenvolvimento de produtos e serviços de software, no contexto da sistematização e gestão dos processos de trabalho de apoio ao chamado ciclo de vida do software.

Segundo a norma ISSO/IEC 12207, os processos de ciclo de vida de *software* podem ser empregados para adquirir, fornecer, desenvolver, operar e manter produtos de *software*. Ela agrupa as atividades, que podem ser executadas durante o ciclo de vida de *software*, em cinco processos fundamentais (processos principais), oito processos de apoio (processos de suporte) e quatro processos organizacionais (processos organizacionais) (ver **FIG. 2.14**). Nos processos principais estão enquadrados os clientes, fornecedores, operadores, utilizadores e a equipe de desenvolvimento. Já a equipe de suporte está inserida nos processos de suporte e organizacionais.

Essa norma, no entanto, não especifica os seguintes elementos: detalhes de implementação, detalhes de documentação, qual o modelo de ciclo de vida, ou o método de desenvolvimento de software que será utilizado.

FIG. 2.8: Processos de Ciclo de Vida de Software (ISO/IEC 12207).

2.2.2.1 PROCESSOS PRIMÁRIOS

Os processos primários do ciclo de vida do software constituem um conjunto de cinco processos que atendem as partes fundamentais (pessoa ou organização) durante o ciclo de vida de software. Os Processos Primários abrangem os seguintes processos:

- **Processo de Aquisição** - define as atividades de aquisição da organização (a organização que adquire um sistema, produto de software ou serviço);
- **Processo de Fornecimento** - define as atividades do fornecedor da organização (a organização que fornece o sistema, o produto de software ou o serviço de software ao cliente);
- **Processo de Desenvolvimento** - define as atividades da equipe de desenvolvimento da organização (a organização que define e desenvolve o produto de software);
- **Processo de Operação** - define as atividades de operação da organização (a organização que fornece o serviço de operação de um sistema informatizado); e

- **Processo de Manutenção** - define as atividades de manutenção do software da organização (a organização que fornece o serviço de manter o produto de software; isto é, gerir as modificações do produto de software para mantê-lo atualizado e operacional).

2.2.2.2 PROCESSOS DE SUPORTE

Os processos de suporte do ciclo de vida do software constituem um conjunto de oito processos. Um processo de apoio auxilia um outro processo como uma parte integrante, com um propósito distinto, e contribui para o sucesso e qualidade do projeto de software.

Os Processos de Suporte abrangem os seguintes processos:

- **Processo de Documentação** - define as atividades para registrar as informações produzidas por um processo do ciclo de vida;
- **Processo de Gestão de Configurações** - define as atividades de gestão de configuração;
- **Processo de Garantia da Qualidade** - define as atividades para assegurar de forma objetiva que os produtos e os processos de software estão conforme com os requisitos especificados e conforme o planejado. As revisões conjuntas, as auditorias, a verificação, e a validação podem ser usadas como técnicas da garantia de qualidade;
- **Processo de Verificação** - define as atividades (para o cliente, o fornecedor, ou parte interessada) para verificar os produtos de software;
- **Processo de Validação** - define as atividades (para o cliente, o fornecedor, ou parte interessada) para validar os produtos de software;
- **Processo de Revisão** - define as atividades para avaliar o estado e os produtos de uma atividade;
- **Processo de Auditoria** - define as atividades para determinar a conformidade com os requisitos, planejamento e contrato; e
- **Processo de Resolução de Problemas** - define um processo para analisar e resolver problemas (incluindo não-conformidades), qualquer que seja a sua natureza ou origem no decorrer do desenvolvimento, operação, manutenção ou outro processo.

2.2.2.3 PROCESSOS ORGANIZACIONAIS

Os processos organizacionais de ciclo de vida constituem um conjunto de quatro processos. Eles são empregados por uma organização para estabelecer e implementar uma estrutura, constituída de processos de ciclo de vida e pessoal associados, e melhorar continuamente a estrutura e os processos. Os Processos Organizacionais abrangem os seguintes processos:

- **Processo de Gestão** - define as atividades básicas de gestão, incluindo a gestão de projeto, durante o ciclo de vida;
- **Processo de Infra-Estrutura** - define as atividades básicas para estabelecer a infra-estrutura e apoio de um processo de ciclo de vida;
- **Processo de Melhoria** - define as atividades básicas ao desempenho da organização (quer seja cliente, fornecedor, equipes de desenvolvimento e manutenção, ou gestor de outro processo) para estabelecer a medição, controlo e melhoria do ciclo de vida; e
- **Processo de Formação** - define as atividades para prover pessoal adequadamente treinado.

2.2.3 ALTERAÇÕES INSERIDAS NO CICLO DE VIDA DO SOFTWARE COM O REUSO

Segundo GURGEL (2004), após as modificações sugeridas na proposta de revisão da ISO/IEC 12207-1995, temos o acréscimo de três processos: processo de engenharia de domínios, processo de gerência de *assets* e processo de gerência do programa de reuso, conforme havia sido definido pela IEEE 1517 (1999) (parte escurecida na **FIG. 2.9**).

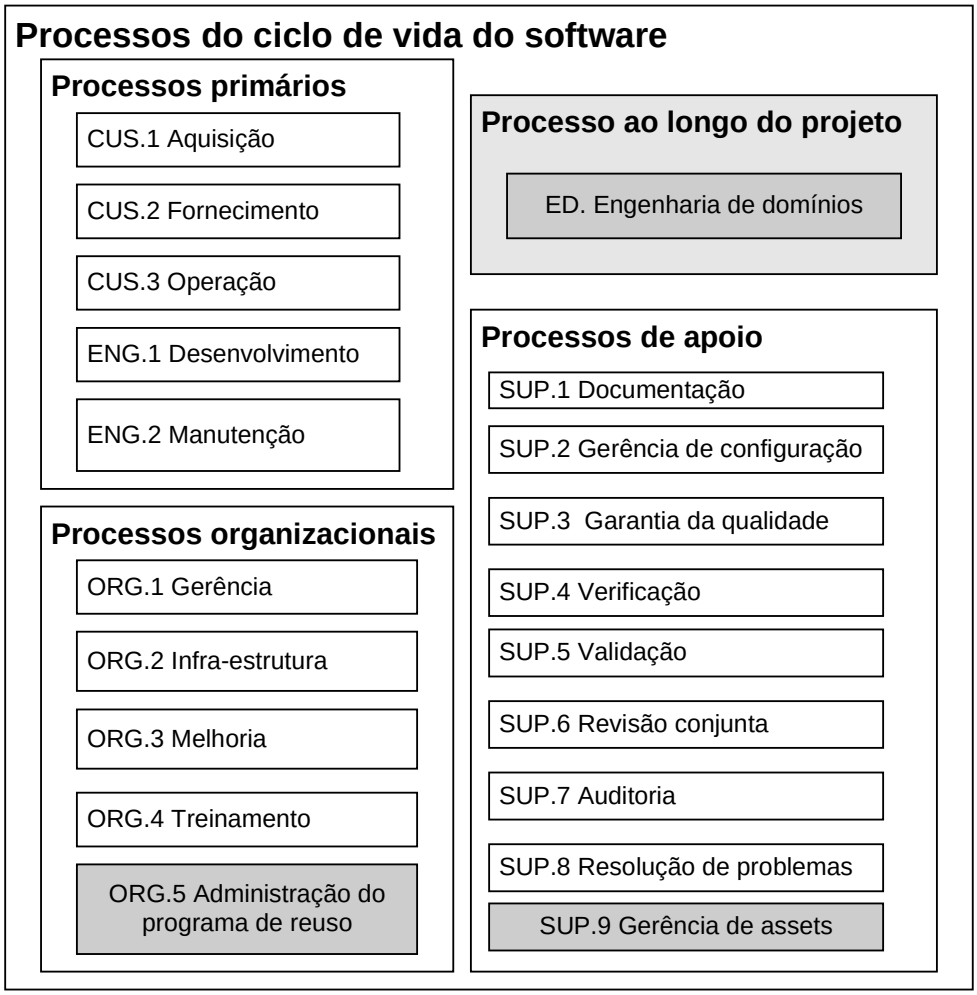


FIG. 2.9: Ciclo de vida do software baseado em reuso pela IEEE 1517 (1999).

- **Processo de Engenharia de Domínio** – envolve a criação e manutenção de modelos de domínio, arquiteturas de domínio e outros *assets* de domínio que tenham sido definidos;
- **Processo de Gerência de *Assets*** - responsável por aplicar procedimentos administrativos e técnicos a fim de: identificar, definir, certificar, classificar e representar os *assets*; guardar modificações, migrações e versões dos *assets*; guardar e reportar o status dos *assets*; estabelecer e controlar o armazenamento e fornecimento dos *assets*; e a forma de cancelamento dos *assets*; e
- **Processo de Gerência do Programa de Reuso** - define as e tarefas do administrador do plano de reuso e é utilizado para planejar estabelecer, gerenciar, controlar e monitorar o programa de reuso da Organização.

2.2.4 PROCESSO DE REUSO

Segundo MOORE (2004), o processo de reuso deve maximizar reuso e minimizar esforços básicos de desenvolvimento. Então, o primeiro passo no processo é encontrar e selecionar *assets* adequados. Para efetivo reuso deve ser mais fácil achar *assets* que os desenvolver para único uso (JACOBSON et al., 1997). Ao encontrar os *assets* adequados não significa ter achado exatamente o que precisa. Localizar *assets* semelhantes pode ser suficiente. Depois que os *assets* foram encontrados, eles devem ser entendidos para serem reusados. Achar e entender estão relacionados, porque para selecionar um *asset* para reuso, deve-se saber o que ele faz. Entender fica mais importante quando o *asset* tiver que ser modificado. Assim, a documentação adequada é fundamental para este passo. Naturalmente, restrições legais podem proibir a disponibilidade da implementação de um *asset* e podem permitir seu reuso somente como uma caixa preta (IEEE1517, 1999).

Construir um sistema de *software* sem precisar modificar os *assets* é o ideal. Geralmente, pelo menos alguns dos *assets* devem ser adaptados às necessidades específicas do sistema de *software* a ser construído. Os *assets* podem ser modificados mudando características internas ou acrescentando novas características. Quando um *asset* oferece a funcionalidade exigida, ele deve ser incorporado no sistema de *software*. Porém, os *assets* existentes podem não ser suficientes para construir um novo sistema. Portanto, alguns *assets* devem ser construídos do zero para serem inseridos no repositório de *assets* reutilizáveis, facilitando o seu reuso em projetos futuros (FICHMAN, 1999).

Desta forma, desenvolver para reuso implica em construir modelo de domínio e arquitetura, construir novos *assets*, ou alterar *assets* existentes para melhorar sua reusabilidade. Para apoiar desenvolvimento para reuso, devem ser acrescentadas ao ciclo de vida do *software* as seguintes atividades (IEEE1517, 1999):

- Executar análise de domínio;
- Executar projeto de domínio; e
- Construir *assets* reutilizáveis.

Na **FIG. 2.10** é apresentado um resumo das etapas do processo de reuso. O processo de reusar um *asset* existente requer as seguintes atividades: Busca, Entende e Adapta. A atividade Desenvolve é necessária quando nenhum *asset* reutilizável pode ser identificado. A atividade Integra deve ser feita mesmo que um *asset* existente seja reusado ou um novo tenha

sido desenvolvido. Finalmente, o novo *asset* ou o *asset* modificado deve ser generalizado e classificado para ser armazenado no repositório de *assets* para futuro reuso (FRAKESb, 1996).

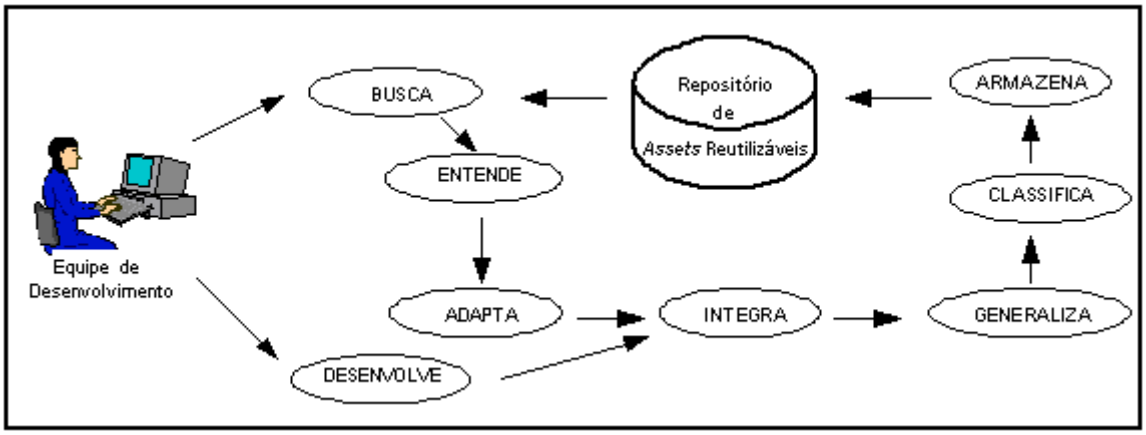


FIG. 2.10: Procedimentos para reuso de *software.*

É importante que a documentação seja considerada uma parte essencial do *asset*, pois sem sua própria documentação um *asset* é inútil. Nem pode ser recuperado quando for necessário e nem pode ser reusado e adaptado com esforço razoável. Desta forma, a documentação de cada *asset* deve ser clara, concisa e auto-suficiente para agilizar e viabilizar os seguintes aspectos: a avaliação de *assets* em um conjunto de possíveis candidatos, a compreensão da funcionalidade de um *asset*, o uso de um *asset* em um certo ambiente e a adaptação de um *asset* para necessidades específicas (SAMETINGER, 1996).

Porém, antes de se pensar na implantação do processo de reuso, o primeiro e mais importante passo é obter suporte e comprometimento da alta gerência da organização para o programa de reuso. Isso é crucial pois o programa de reuso pode afetar todas as áreas do processo de produção de software na empresa e o suporte da alta gerência precisa ser obtido para permitir que mudanças nos processos existentes sejam implantadas (ou até mesmo mudanças na política da empresa, se necessário) (BIGGS, 2004).

Junto com o processo de reuso, surgem outros pontos a considerar a fim de se obter sucesso:

- Definir novos papéis para coordenar o reuso entre equipes diferentes;
- Permitir o treinamento das equipes envolvidas; e

- Prover incentivos apropriados para o reuso, a fim de evitar argumentos do tipo: “Por que prejudicar o cronograma do meu projeto apenas para que o que eu faço seja reutilizado por outra equipe depois?”

2.2.4.1 REPOSITÓRIO

O repositório é o local onde os *assets* são armazenados e deve conter um catálogo com todos os *assets* disponíveis, bem como ser de fácil acesso e acessível a todas as pessoas interessadas, ou participantes do projeto. Deve-se levar em conta também que o repositório contém informações intelectuais importantes da empresa e, por isso, a forma de acesso e a identificação de cada acesso devem ser controladas e gerenciadas.

Geralmente quando se fala em repositório de *assets* pensa-se logo num repositório centralizado, mas não existe nenhuma regra ou razão que defina a necessidade da empresa em centralizar o armazenamento dos *assets*. Abaixo alguns fatores a considerar na implantação do repositório:

- Número de desenvolvedores, equipes e ambientes de desenvolvimento da organização; e
- Número de *assets* e número de domínios de aplicação aos quais os *assets* pertencem.

Nos casos em que haja poucos *assets*, ou poucos desenvolvedores, e exista boa informação acerca de onde se encontram os *assets* a serem reutilizados, pode-se questionar a real necessidade da centralização do repositório.

À medida que o número de *assets* cresce, é necessário definir e manter de forma eficiente um repositório com um catálogo dos *assets*. Isso irá evitar situações em que o desenvolvedor não sabe quais *assets* ele pode reutilizar, não sabe onde encontrar um determinado *asset* ou não sabe onde armazenar um *asset* (EZRAN, 2002).

Um repositório de *assets* pode ser gerenciado (seu controle está centralizado em uma única pessoa) ou não gerenciado (não há um responsável pelo seu controle), a diferença entre eles está no fato que, neste último, a eficácia do reuso reside no próprio desenvolvedor, que deve saber onde encontrar cada *asset*, que pode estar misturado em torno de outros artefatos não reusáveis, e ainda correr o risco de utilizar uma versão obsoleta, ou não adequada, do *asset*.

Existe também o perigo de, à medida que o repositório cresce, ele acabar se tornando um repositório de *assets* sem utilidade, ou com uma funcionalidade sub-utilizada, onde vários *assets* executam a mesma função, ou simplesmente não irão executar nenhuma função no futuro, pois nunca serão reusados. Até mesmo nos repositórios para uso pessoal existe a necessidade de se eliminar *assets* duplicados ou sem funcionalidade para se assegurar à utilidade do repositório em longo prazo (NORTON, 2003).

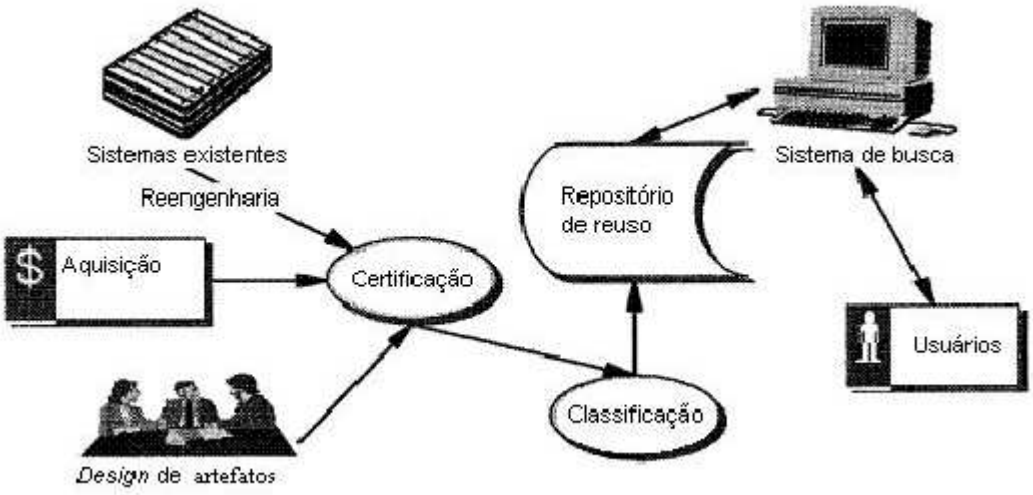


FIG. 2.11: Repositório de *assets* e processos envolvidos

2.2.4.1.1 VANTAGENS DO REPOSITÓRIO

Michel Ezran apresentou as seguintes vantagens em se adotar o uso de um repositório centralizado de *assets* (EZRAN, 2002):

- Lugar único para armazenar os *assets*, bem como pesquisar por *assets* existentes;
- Forma padronizada de documentar, pesquisar e enumerar os *assets*; e
- Forma padronizada de controlar mudanças e melhorias nos *assets*.

2.2.4.1.2 CONTEÚDO DO REPOSITÓRIO

Já é sabido que um repositório de *assets* deve possuir, obviamente, *assets* cujo reuso seja freqüente nos projetos desenvolvidos pela empresa. Mas é preciso considerar que os diferentes *assets* disponíveis podem não ser iguais em termos de linguagem de programação, formato de arquivos, granularidade e documentação. Além disso, como visto anteriormente, o repositório deve conter um catálogo de forma a facilitar a busca e identificação dos *assets* pelo desenvolvedor. Sendo assim, pode-se concluir que o repositório será composto de um catálogo e de vários *assets*. Como os *assets* armazenados podem ser diferentes entre si, para que o catálogo possa referenciar os diversos *assets* existentes, cada *asset* deve ser composto de uma parte comum, a qual será referenciada no catálogo. Essa parte comum é a sua ficha cadastral, a outra parte seria o corpo do *asset*, que na verdade engloba todo o conjunto de artefatos que compõem o *assets* original (EZRAN, 2002).

A ficha cadastral do *asset*, também chamada de meta-dado do *asset* (NORTON, 2003), deve conter uma série de campos a fim de facilitar a sua busca no catálogo:

- Nome único de identificação do *asset*;
- Autor;
- Data de criação;
- Data da última atualização;
- Versão atual;
- Lista de palavras chave associadas ao *asset*; e
- Breve descrição do *asset* (área de uso ou tecnologia envolvida).

Detalhes de usabilidade, instalação e solução são descritos em documentos à parte, pertencentes ao corpo do *asset*;

- Localização do *asset* (o *asset* pode estar armazenado dentro do repositório ou externamente, em algum sistema de controle de versão da empresa ou até mesmo externamente à empresa, num servidor *web* de outra companhia, por exemplo).

Um modelo básico de um repositório de *assets* apresentado por EZRAN (2002) pode ser visto na **FIG. 2.12**.

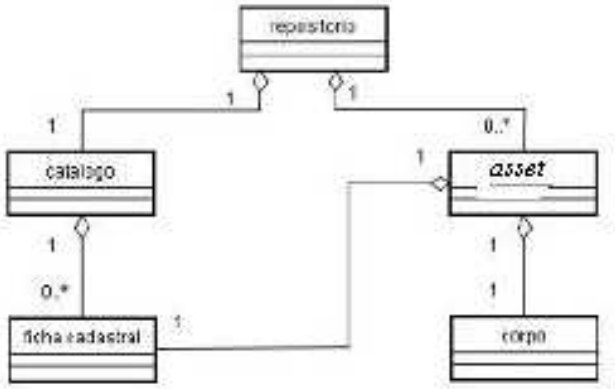


FIG. 2.12: Modelo de implementação de repositório

Esse modelo de implementação de repositório foi elaborado usando-se a notação da linguagem UML, de forma que seja possível implementá-lo em qualquer linguagem de programação. Seguindo-se a notação dos *assets* e o relacionamento entre eles, vemos que o repositório é composto por um catálogo e um conjunto de 0 ou mais *assets*, sendo que o catálogo e cada *asset* estão associados a um único repositório. O catálogo, por sua vez, é composto de 0 ou mais fichas cadastrais, e cada ficha cadastral pertence à somente um catálogo. Cada *asset* possui uma ficha cadastral e um corpo associado. A mesma ficha cadastral que é parte do *asset* é a que faz parte do catálogo e se constitui no elo de ligação entre o catálogo e os *assets* do repositório. Sendo assim, a partir do repositório construir o catálogo e, através das fichas cadastrais existentes no catálogo, chega-se ao *asset*, cujo corpo se constitui no artefato ou conjunto de artefatos de software a ser reusado pelo usuário do repositório.

2.2.4.1.3 FUNÇÕES DO REPOSITÓRIO

Basicamente, a função do repositório é listar seus *assets* e prover meios de acesso a eles, mas pode ser aprimorado a fim de prover várias outras funções de forma a melhorar a sua utilização, tanto por parte dos desenvolvedores quanto da equipe de gerenciamento do repositório (EZRAN, 2002).

São apresentados alguns exemplos de funções possíveis de serem implementadas no repositório:

- **Identificação e descrição do *asset***: cada *asset* deve ser descrito por um único nome e possuir um conjunto padrão de atributos;

  Exemplo: o conjunto padrão de atributos poderia ser composto por nome, domínio, categoria, versão, autor e descrição;
- **Inserção e remoção de *assets***: possibilidade do gerenciador de reuso inserir novos *assets* e retirar *assets* obsoletos. Essa função pode ser estendida aos próprios usuários do repositório. A inserção do corpo de um *asset* deve estar atrelada à correspondente inserção da correta descrição do *asset*, conforme o item anterior. Uma extensão dessa funcionalidade é o cenário em que uma nova versão de um *asset* já existente é adicionada ao repositório;
- **Visualização do catálogo**: deve ser possível a visualização de todo o catálogo, contendo a descrição de todos os *assets*. É particularmente útil para encontrar rapidamente o *asset* quando o repositório é pequeno (possui poucas dezenas de *assets*);
- **Mecanismo de busca**: deve ser possível a busca por palavras chave, fornecidas pelo usuário, contidas em qualquer parte da ficha cadastral do *asset*. O resultado é a apresentação das fichas cadastrais dos *assets* encontrados. Uma extensão é a busca diretamente no corpo do *asset*;
- ***Download**:* uma vez encontrado o *asset*, deve ser possível obter o *asset* completo (o corpo do *asset*) de maneira simples, rápida e segura;
- **Organização do catálogo**: como o repositório pode conter muitos *assets*, a busca textual pode ser muito demorada ou ineficiente (ou ambos). É interessante que o catálogo seja organizado também por palavras chave e sinônimos na descrição do *asset*. Em alguns repositórios essa organização ocorre através de menus hierárquicos (em forma de árvore), e a pesquisa textual pode ser restrita a algumas áreas do catálogo, tornando-se mais rápida e eficiente;
- **Histórico:** o repositório deve conter a descrição do histórico de cada *asset* (autor, data de criação, lista de modificações). Outros dados podem ser registros de melhorias necessárias, resultados de integração e teste do *asset*, exceções ao reuso. Esses dados podem estar na ficha cadastral;

- **Estatísticas**: o repositório deve coletar dados e registrar automaticamente quais os *assets* mais acessados, quando, quantidade e tipo de buscas, número de *assets*;
- **Controle de acesso**: como os *assets* podem conter informações confidenciais, devem ser restritos a grupos específicos, bem como a gerência do repositório deve ser feita por uma única pessoa (ou um grupo pequeno de pessoas). O acesso deve ser controlado também por motivos de se garantir a integridade do repositório (minimizar os riscos de remoção indevida de *assets*, por exemplo);
- **Controle de versão**: o repositório deve conter mecanismos de controle de múltiplas versões de um mesmo *asset*;
- **Controle de mudanças**: procedimentos devem ser definidos para propor, discutir, aceitar e implementar mudanças nos *assets* do repositório;
- **Notificação de mudanças**: mudanças nos *assets* (adição, remoção) bem como mudanças no próprio repositório (novas funções, procedimentos) devem ser notificadas automaticamente a todos os usuários em potencial; e
- **Notificação de atualizações**: identificar de forma única cada usuário que esteja reusando cada *asset* do repositório, de forma a notificar esses usuários sobre atualizações (devido a correções de falhas, por exemplo) nos *assets* que estejam reusando.

Nota-se que nem todas são necessárias, dependendo do contexto em que o repositório é utilizado dentro da empresa. Por exemplo, se o repositório é usado apenas em projetos internos dentro de uma mesma empresa, um controle de acesso pode mostrar-se desnecessário, ou se existirem poucos *assets* no repositório, o mecanismo de busca pode ser eliminado, já que é mais simples apresentar sempre a lista completa dos *assets*.

2.2.4.1.4 IMPLEMENTAÇÃO

A implementação de um repositório deve obedecer a uma estrutura planejada de forma a facilitar seu uso e manutenção ao longo de sua existência. Como muitas empresas estão centradas num modelo de redes (senão na própria Internet), é imprescindível que o repositório seja acessível de qualquer nó de rede dentro da empresa. Sendo assim, um modelo de arquitetura 3-camadas seria o recomendado para a implementação do repositório, pois essa

arquitetura permite conciliar os requisitos de usabilidade, manutenibilidade e disponibilidade requeridas pelo repositório (EZRAN, 2002).

2.2.4.1.5 MÉTODOS DE CLASSIFICAÇÃO

Segundo ROSSI (2004), em uma biblioteca ou em um repositório, é necessário que os as*sets* sejam armazenados de modo que facilite sua busca, sua seleção e sua recuperação.

Desta forma, é importante discutir os métodos de classificação e recuperação da informação dos *assets* em um repositório ou biblioteca, já que são recursos essenciais para a implementação e aceitação do processo de reutilização. Os métodos mais citados na literatura sobre reutilização de software, são: métodos de biblioteca tradicional, métodos baseados no conhecimento e método de hipertexto.

Os métodos tradicionais de classificação e recuperação de informação são classificados de acordo com o tipo de vocabulário, conforme o apresentado na **FIG. 2.13**. Segundo a taxonomia apresentada por Frakes e Gandel (1990), estes métodos podem ser de vocabulário controlado ou vocabulário não controlado.

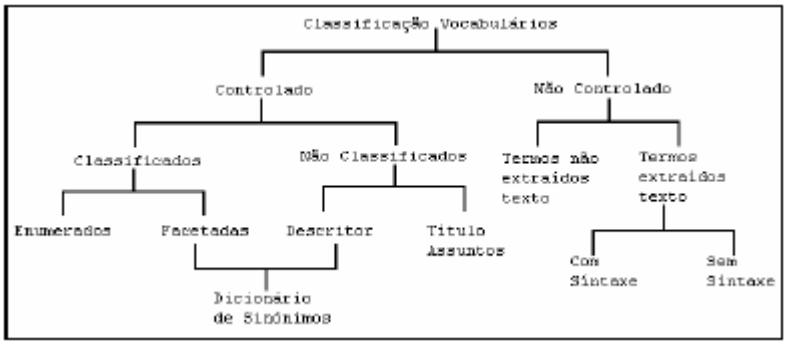


FIG. 2.13: Taxonomia comum de vocabulários para classificação

Entende-se por vocabulário controlado, quando os termos utilizados por indexadores e pesquisadores são restritos a um conjunto comum; deste modo, é assegurado que os termos usados pelos participantes dos projetos são os mesmos. Ao contrário, vocabulário não controlado ocorre quando pouca ou nenhuma restrição é imposta em relação à seleção ou sínteses de termos, e o relacionamento semântico entre termos não é limitado.

Os métodos de vocabulário controlado podem ser classificados ou não classificados.

Os métodos classificados possuem sua estrutura apresentada através da forma enumerada e de facetas. Os métodos não classificados são menos estruturados e são baseados em palavras-chave que podem ser descritores e títulos-assunto. Os métodos de vocabulário não controlado apresentam metodologias que permitem flexibilidade tanto na recuperação quanto na classificação, como no método de recuperação de texto-livre.

No método de classificação enumerada, um dado ou área de interesse é dividido em sucessivas classes mais restritas e assim, resultando em uma árvore de hierarquia estruturada (FRAKES; GANDEL, 1990). Pode-se citar como exemplo de aplicação deste método, o sistema decimal de Dewey, que é um sistema de classificação de livros ou outras publicações, onde as classes são designadas por um número de três dígitos e as subdivisões são mostradas por números depois do ponto decimal.

O método de classificação por facetas é baseado na análise de um assunto em função de seus termos básicos para a organização dos assuntos em facetas ou classes elementares básicas. Desta forma, o assunto em um domínio é descrito pela síntese destas classes ou facetas básicas. As facetas são consideradas como perspectivas, visões ou dimensões de um domínio de aplicação em particular (PRIETO-DÍAZ; FREEMAN, 1987).

O método de classificação em descritores utiliza listas, ordenadas alfabeticamente, de palavras simples que possam ser usadas para descrever a informação sobre um item.

Já o método de classificação de título-assunto utiliza listas, ordenadas alfabeticamente, de termos em linguagem natural e frases que podem ser usadas para descrever a informação sobre um item.

Os métodos classificação em descritores e de título-assunto são muitos relacionados, pois, em ambos os casos, o vocabulário de termos é limitado a uma lista prescrita. A principal diferença entre os métodos é que o método de classificação de descritor tende a utilizar freqüentemente palavras simples, enquanto que, método de classificação de título-assunto utiliza frases.

O recurso de dicionários de sinônimos é um mecanismo para descrever a variedade de relacionamentos entre termos dentro de uma listagem alfabética, ou seja, sua finalidade é restringir vocabulário agrupando os termos que possuam o mesmo significado em um termo (FRAKES; GANDEL, 1990).

No método de vocabulário não controlado ou análise de texto-livre, as palavras de texto são analisadas pela sua freqüência no texto e as palavras-chaves relevantes são derivadas automaticamente, através de suas propriedades estatísticas e posicionais. Deste modo, os

resultados são índices automáticos constituídos de uma associação dos termos escolhidos para representar ou definir um item (PRIETO-DÍAZ, 1991).

Neste método, os índices podem ser definidos manualmente, ou podem ser extraídos a partir do texto descritivo do documento relacionado ao componente, removendo as palavras que pertencem a "*stop list*", tais como artigos, conjunções, pronomes e outros (e, a, o, um, uma, se, qual, que, por que, etc.), o usuário especifica a consulta usando palavras-chaves que são aplicadas aos índices para encontrar uma combinação com um documento (HENNINGER, 1997).

2.2.4.2 GERÊNCIA DA QUALIDADE

Segundo MOORE (2004), a Gerência da Qualidade é responsável pela determinação e o planejamento do trabalho. Ela deve zelar pela qualidade em termos de determinar o que fazer para que o todo alcance os resultados propostos (CROSBY, 1996). Para isto, um Comitê de Qualidade deve ser formado por consultores da qualidade para apoiar as equipes de desenvolvedores/montadores de projetos nas avaliações da qualidade dos artefatos produzidos durante todo processo de ciclo de vida do projeto, determinando de que modo o produto deve ser verificado, inspecionado, testado e controlado no decorrer de sua vida dentro e fora da organização (ROCHA et al., 2001). Este comitê deve, também, ajudar na identificação dos atributos de qualidade necessários para que cada um dos projetos alcance seus objetivos e a equipe de desenvolvedores/montadores enfoque no desenvolvimento de *software* baseado no reuso (IEEE1517, 1999).

A garantia da qualidade somente pode ser assegurada se existir um padrão de especificação confiável e seguido, e também controlado de forma sistemática. O grupo de garantia da qualidade de *software* deve utilizar técnicas, ferramentas, métodos, padrões, normas, processos e bases de dados para apoiar os diversos projetos de *software*. Ele deve apoiar a equipe de desenvolvimento de *software* por meio de verificações e validações, que permitem avaliar os artefatos produzidos (ABNT12207, 1998; ISO15504, 1998; ROCHA et al., 2001; FIORINI et al., 1998).

A avaliação deve ser planejada e realizada, conforme o planejamento por revisores, verificadores ou testadores treinados e sem vínculo operativo no artefato examinado (PAULK

et al., 1993; CROSBY, 1996; ABNT12207, 1998; ISO15504, 1998; ROCHA et al., 2001; SEI, 2002). As Normas CMM, CMMI, ABNT 12207 e ISO 15504 recomendam que os executores das atividades de verificação e validação não devem ter participação na elaboração do artefato a ser examinado (PAULK et al., 1993; SEI, 2002; ABNT12207, 1998; ISO15504, 1998). As tensões entre autores e revisores dos artefatos são apontadas na abordagem da Gerência pela Qualidade Total[2] (GQT) como grandes fontes geradoras de conflitos que comprometem a qualidade do produto (KAN, 1995). Esta fonte geradora de conflito deve ser minimizada através de esclarecimentos durante a etapa de treinamento da revisão e, também, com a introdução de um líder que exerce o papel de moderador nas reuniões, de forma a eliminar ou reduzir o número de conflitos (WIEGERS, 1995).

Cada atividade de verificação e validação deve apresentar dois resultados: produto de *software* é aceito ou rejeitado, e o registro de não-conformidades e de cálculos (ABNT12207, 1998). Acumulando os resultados dos cálculos e analisando-os, o Comitê de Qualidade pode determinar exatamente o estado do produto de *software*. Ele reúne os dados provenientes das atividades de verificação e validação, e os registra de maneira a ser de utilidade prática para todos os interessados. A gerência de reuso e de projeto precisa de dados acurados das tendências, para saber quando e onde deve agir (IEEE1517, 1999).

O controle da qualidade deve produzir um documento contendo todos os problemas identificados, e estes devem ser resolvidos. Após a cada alteração, é necessário repetir o controle da qualidade (ABNT12207, 1998; FIORINI et al., 1998).

Ishikawa propôs sete ferramentas básicas de estatística para controle da qualidade (PALADINI, 1994) que são úteis para o planejamento e controle de projetos de desenvolvimento de *software*. Dentre elas se destacam o Diagrama de Causa-Efeito, o *Checklist* e o Diagrama de Pareto.

O Diagrama de Causa-Efeito, conhecido como Gráfico Espinha-de-Peixe ou Gráfico de Ishikawa, identifica todos os fatores de causa de uma característica da qualidade em um único gráfico. Sua forma é similar à espinha de peixe, onde o eixo principal mostra a característica da qualidade, e as espinhas representam os fatores que afetam esta característica (PALADINI, 1994).

O *Checklist* é um formulário que contém itens para serem verificados conforme as necessidades específicas de seus usuários, e por isso, apresentam extrema flexibilidade de

[2] GQT é um sistema que combina técnicas de controle da qualidade e modelos organizacionais, onde a qualidade é reconhecida como uma vantagem estratégica com o apóio total da alta-administração.

elaboração, utilização e interpretação. Seu principal objetivo é orientar a coleta de dados que vão ser analisados posteriormente (PALADINI, 1994).

O Diagrama de Pareto, representado por barras de freqüência, é utilizado para classificar as causas de defeitos que atuam em um processo específico de acordo com seu grau de importância (PALADINI, 1994). A análise de Pareto retrata o princípio 80-20 (20% das causas são responsáveis por 80% dos defeitos). Este diagrama é bastante aplicado na Qualidade de Software, porque os defeitos do software ou densidade dos defeitos não seguem uma distribuição uniforme (HAZAN, 1999).

2.2.4.3 SERVIÇOS DE SUPORTE AO PROCESSO DE GERÊNCIA DE ASSETS

O processo de reutilização refere-se a um conjunto de fases que incluem métodos utilizados para produção, intermediação e consumo de componentes reutilizáveis para o desenvolvimento de sistemas de software (LIM, 1998).

Desta forma, uma organização que deseja promover a reutilização, deve entender seu processo de desenvolvimento para aplicar as mudanças necessárias para suportar incorporação das fases necessárias para a reutilização.

O processo de reutilização consiste de quatro fases principais, apresentadas na **FIG. 2.14** [(JACOBSON; GRISS; JONSSON, 1997), (LIM, 1998)]:

- **Gerência da infra-estrutura de reutilização**: planejar e executar os processos de reutilização; para isso, inclui um conjunto de atividades relacionadas ao planejamento, iniciação, fornecimento de suporte, monitoração e coordenação das outras fases do processo de reutilização;
- **Produção de componentes reutilizáveis**: identificar e fornecer os artefatos reutilizáveis apropriados para as necessidades dos projetistas de software;
- **Intermediação (suporte) dos componentes reutilizáveis**: apoiar a reutilização, através de qualificação e certificação, configuração, manutenção e divulgação dos componentes de software reutilizáveis;
- **Consumo dos componentes reutilizáveis**: utilizar a biblioteca e as ferramentas associadas, identificar e recuperar os componentes necessários, integrar os

componentes no sistema que está sendo construído, fornecer retorno e requisitar novos componentes para responsável pela biblioteca de componentes.

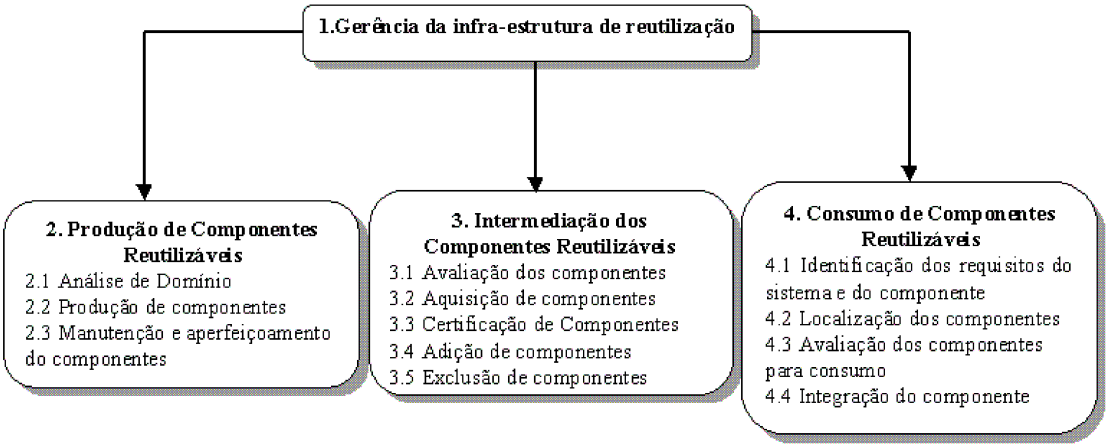


FIG. 2.14: O processo de reutilização extraído de Lim (1988, p.399).

Em cada uma destas fases são realizados os serviços para que a reutilização de software aconteça dentro do processo de desenvolvimento. Segundo Apperly (2001), a relação de serviços que devem ser oferecidos para fornecer suporte à gerência, produção e consumo de componente de software são os apresentados na **TAB**. **2. 2.**

TAB. 2.2: Serviço de Gerência de Componentes segundo Apperly (2001, p.515).

Segmento	Serviços
Produção	Divulgar os componentes projetados informalmente ou não documentados
	Utilizar especificações de componentes com ponto de início para projeto
	Divulgar componentes com especificações
	Re-divulgar componentes e especificações
	Notificar consumidores de novos componentes ou problemas
Gerência	Gerenciar a biblioteca do repositório
	Gerenciar os usuários do repositório
	Gerenciar o catálogo
	Assegurar a qualidade dos componentes
	Gerenciar os componentes
	Tornar os componentes disponíveis
	Gerenciar as versões dos componentes
Consumo	Buscar componentes que necessita
	Reportar falhas identificadas
	Especificar componentes para desenvolvimento
	Utilizar e reutilizar especificações de componentes
	Utilizar e reutilizar componentes
	Dispor componentes
	Registrar interesse em componentes
	Receber novas notificações de componentes
	Revisar novos componentes
	Atualizar especificações de componentes e componentes

.

Desta forma, um ambiente para apoiar o processo de reutilização de *software* deve disponibilizar recursos para auxiliar nas tarefas de cada pessoa que compõe este processo, baseando-se nos serviços propostos por Apperly (2001). O processo de reutilização (APPERLY, 2001) está dividido nos processos de produção, gerência e consumo e é executado pelos seguintes papéis conforme **FIG. 2.15**:

- **Produção -** produtor: implementa os componentes;
- **Consumo** - consumidor: busca e usa os componentes; e
- **Gerência** - gerenciador de *assets*: interface entre produtor e consumidor

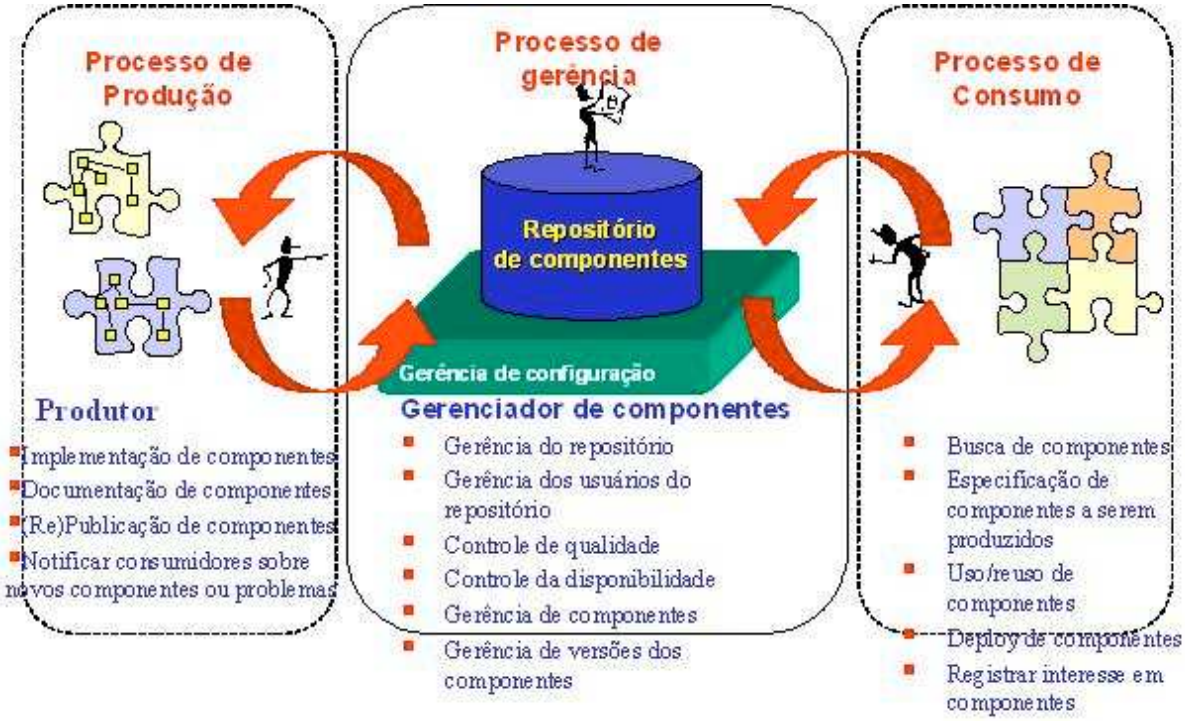


FIG. 2.15: Processos de produção, gerência e consumo segundo Apperly (2001).

2.2.4.4 MÉTRICAS

Segundo SANCHES (2005), o ponto de partida para qualquer programa de métricas de reuso é definir exatamente o que medir. Sem ter essa resposta plenamente consolidada o que será medido serão apenas números com pouca ou nenhuma importância. Coletar números é fácil, produzir medidas úteis não, e muitos programas de métricas falham porque não

conseguem diferenciar essas duas situações (EZRAN, 2002). Aqui se aplica claramente uma frase dita por Watts Humphrey: “Você pode obter praticamente qualquer número que você queira (...) mudando apenas a maneira como você conta”.

Para a introdução de métricas em reuso não se deve pensar em redução de custos, mas na redução no tempo de implementação ou na melhoria da qualidade do software produzido.

O nível de medida de reuso requer um entendimento preciso do que é contado como reuso ou não. Não existe um padrão consolidado, cada organização deve adotar um modelo e usá-lo consistentemente. Por exemplo: somente *assets* vindos do repositório são contados como reuso. Isso marca a diferença entre reuso oportunístico e reuso planejado. Às vezes valores absolutos de medidas podem não ser significativa, mas as variações desses valores observados ao longo dos projetos podem representar informações úteis e significativas para o processo de reuso (EZRAN, 2002).

Segundo Moore (2004), as métricas são necessárias para avaliar a robustez da arquitetura, em termos das mudanças de requisitos requererem mudança na arquitetura. Também são necessárias métricas para determinar a perda da reutilização em virtude de sucessivas manutenções. A perda de reutilização ocorre quando uma versão de *asset* é particionada em várias versões coexistentes, em virtude da necessidade de cada uma dessas versões necessitarem de atenção específica pela gerência de configuração e ao realizar novas alterações (ALMEIDA et al., 2003).

Para coletar os dados, é importante uma definição clara de como contar o que é reusado e como é reusado. Essa definição poderia ter a seguinte forma: um *asset* é qualquer “artefato reutilizável” vindo do repositório e utilizado sem modificações para construir, direta ou indiretamente, um produto desenvolvido num dado projeto. Esse *asset* deve ser contado apenas uma vez por projeto, mesmo que tenha sido reusado mais de uma vez no mesmo projeto (EZRAN, 2002). Dessa definição extraem-se pontos importantes a serem considerados:

- **“Utilizado sem modificações”:** se o *asset* precisar ser modificado, então há um trabalho de desenvolvimento envolvido e isso não deve ser considerado como reuso;
- **“Vindo do repositório”:** somente artefatos qualificados, documentados e desenvolvidos para reuso podem ser contados oficialmente. Serviços ou componentes disponíveis no sistema operacional não são contados como reuso, pois estão disponíveis livremente e não foram desenvolvidos visando um repositório para reuso. Por outro lado, componentes ou aplicativos avaliados e adquiridos de terceiros,

especificamente para o projeto em desenvolvimento e inseridos no repositório para posterior reuso devem ser considerados nas métricas. Os componentes vindos de dentro do próprio projeto não são representativos para as métricas. A lógica para isso é que a identificação, o desenvolvimento e a reutilização de módulos dentro de um mesmo projeto é apenas uma boa prática de *design* que cada projeto deveria adotar de forma padrão, sem a necessidade de técnicas de reuso planejado ou repositórios; e

- **"Contado apenas uma vez":** assim como internamente ao projeto um *asset* reusado várias vezes é desenvolvido apenas uma vez, o *asset* vindo do repositório deve ser contado apenas uma vez.

2.2.4.4.1 EXEMPLO DE MÉTRICAS

Um programa de métricas de reuso deve identificar que atributos de quais entidades (ou *assets*) serão medidos. Até mesmo simples projetos podem ser compostos de centenas de atributos e medi-los todos pode não ser possível devido a limitações de prazo e custo e também não necessário porque nem todos os atributos podem ser úteis.

Desse modo, o programa de métricas é iniciado a partir de certos objetivos, que são definidos a partir de um modelo baseado nas seguintes questões: propósito, atributo, entidade, ponto de vista e contexto. O programa de métricas deve começar com poucos objetivos, que vão sendo refinados e estendidos ao longo do tempo. Assim sendo, podem definir os objetivos iniciais para o programa de métricas de reuso. Esses objetivos são apresentados na **TAB 2.3:**

TAB 2.3: Objetivos para métricas de reuso.

Objetivo	Propósito	Atributo	Entidade	Ponto de vista	Contexto
O-1	Caracterizar	A utilização	Do repositório	Do ponto de vista do gerenciador de reuso	No contexto de um departamento da empresa
O-2	Avaliar	A extensão do reuso	Em um projeto	Do ponto de vista do gerente de projeto e do gerenciador de reuso	No contexto de um departamento da empresa

Definidos os objetivos, para cada um definem-se uma ou mais métricas, correspondentes aos objetivos apresentados na **TAB. 2.3**. Essas métricas são apresentadas na **TAB. 2.4**.

TAB. 2.4: Métricas de Reuso.

Objetivo	Métrica	
O-1	M-11	Número de componentes no repositório
	M-12	Número de acessos (sem *downloads*) ao repositório, por período
	M-13	Número de *downloads* do repositório, por período
	M-14	Número de buscas por palavras chave, sem sucesso, por período
	M-15	Número acumulado de reuso por componente
	M-16	Número acumulado de reuso para todos os componentes no repositório
O-2	M-21	Nível de reuso por projeto
	M-22	Número de acessos (sem *downloads*) ao repositório, por projeto
	M-23	Número de *downloads* do repositório, por projeto

O objetivo O-1 define a necessidade do gerenciador de reuso monitorar a atividade de reuso através da utilização do repositório.

O objetivo O-2 se propõe a compreender como um projeto reusa os *assets* do repositório e é de interesse tanto do gerenciador de reuso quanto do gerente de projeto.

A literatura ainda destaca algumas métricas importantes:

- **Métricas de processo:** produtividade do trabalhador, eficácia em termos de tempo, custo de produção, taxa de defeitos detectados, volume de mudanças de uma versão para outra;
- **Métricas de biblioteca ou repositório:** freqüência de acesso, tamanho do repositório, custo de certificação, qualidade e quantidade de componentes;
- **Métricas de arquitetura:** estabilidade;
- **Métricas do componente:** tamanho do componente, complexidade da interface, coesão do sistema de componente e interface, nível de usabilidade do componente e

interface, qualidade, custo dos sistemas padrões de componente, dificuldades em integrar componentes;

- **Métricas da aplicação:** volume de reuso, qualidade, volume de defeitos e suas causas, perfil da falta.

A quantidade de reuso está relacionada com o volume de reuso ou taxa de reuso (JACOBSON et al., 1997). A taxa de reuso é definida como a razão da soma dos tamanhos dos *assets* dividido pelo tamanho total da aplicação. A taxa de reuso é usada para monitorar o reuso e, também, para estimar como o reuso vai incidir nas medidas de qualidade, tempo e custo.

Frakes e Terry classificam as métricas e os modelos de reuso como (FRAKESb et al., 1996): Modelos de Custo / Benefício de Reuso, Modelo de Avaliação da Maturidade, Métricas de Quantidade de Reuso, Modelo de Fracasso, Métricas de Reusabilidade e Métrica de Bibliotecas de Reuso.

Os Modelos de Custo / Benefício de Reuso incluem a análise econômica de custo / benéfico assim como o custo da qualidade e produtividade. O Modelo de Avaliação da Maturidade categoriza programas de reuso pelo quanto à organização está avançada na implementação do reuso sistemático. As Métricas de Quantidade de Reuso são usadas para avaliar e monitorar um esforço de melhoria do reuso pelo acompanhamento do percentual de reuso de objetos do ciclo de vida. O Modelo de Fracasso é usado para identificar reuso e impedimentos de reuso em uma organização. As Métricas de Reusabilidade indicam a probabilidade que um *asset* é reutilizável. As Métricas de Bibliotecas de Reuso são usadas para gerenciar e acompanhar o uso de um repositório de reuso. O critério de avaliação para indexar esquemas de bibliotecas de reuso é: custos, busca pela efetividade, suporte para entendimento e eficiência As organizações freqüentemente necessitam desses modelos e métrica nesta ordem apresentada por Frakes e Terry (FRAKESb et al., 1996).

A maioria dos modelos econômicos de engenharia de *software* precisa ser ajustada para incluir reuso, e customizada para cada reuso específico. Muitos autores têm modificado os modelos de custos, como COCOMO (BOEHM et al., 1988), que são usados para estimar tempo e esforço e para o desenvolvimento tanto de componentes quanto de aplicações usando componentes (POULIN, 1997).

Jacobson et al. (1997) apresentam como usar um modelo de esforço e custo de reuso simplificado, essencialmente reformulando a análise de Gaffney e Durek (GAFFNEY et al., 1989) e aplicando as recomendações de Poulin (POULIN, 1997). Ele também adiciona um

fator de utilização de biblioteca, pois em uma biblioteca de sistemas de componentes que contém vários sistemas de componentes, é esperado que ocorra mais reuso para os sistemas de aplicação. Jacobson ainda relata que alguns componentes podem ser mais reusados do que outros.

2.2.4.5 RISCOS

No nível de projeto, o simples reuso tende a minimizar os riscos de projeto porque menos partes do produto necessitam ser desenvolvidos pelo desenvolvedor e a qualidade dos componentes é maior que a dos componentes ditos não-reusáveis. No nível organizacional, entretanto, um conjunto grande de riscos existe, sobretudo devido à dificuldade em se prever quais e como os componentes serão reusados. Tais riscos, apresentados abaixo, existem e devem ser minimizados através da implantação de um processo de reuso.

- **Nenhum componente é produzido:** A produção de componentes requer esforço adicional no sentido de identificar dependências externas, além do design e implementação com técnicas mais avançadas. Essas atividades devem ser diferenciadas do processo normal de produção de software, caso contrário corre-se o risco de não se produzir componentes reusáveis (EZRAN, 2002);
- **Componentes existem mas não são encontrados**: Os componentes devem ser catalogados e armazenados de forma centralizada, sob o risco dos desenvolvedores não encontrarem o que procuram (EZRAN, 2002). Esse risco aumenta proporcionalmente ao aumento do número de desenvolvedores e de componentes**;**
- **Perde-se muito tempo para se compreender e avaliar o componente reusável**: Os usuários (desenvolvedores) devem ser capazes de avaliar rapidamente os componentes (sob o ponto de vista técnico e funcional) a fim de decidirem se irão reusá-los ou não. Devem também possuir suporte para realizar tal análise, já que não necessariamente o usuário precisa entender os detalhes de implementação de cada componente. Para isso uma boa ficha cadastral do componente se faz necessária e, se possível, também o suporte dentro da empresa fornecido pelos responsáveis pelo componente reusável (EZRAN, 2002);

- **Componentes não são reusáveis devido a sua baixa qualidade:** Uma vez produzidos, a qualidade e reusabilidade dos componentes devem ser garantidas aos futuros desenvolvedores. Isso implica em boa documentação (atualizada), adequação aos padrões da empresa e adequação a normas técnicas vigentes. Conforme previsto por William Frakes, um processo de certificação para homologar os componentes reusável pode se fazer necessário (FRAKES, 1996);
- **A empresa não está preparada para o reuso, ou o reuso pode não ser apropriado para sua linha de negócios:** Antes de começar o desenvolvimento voltado ao reuso, a empresa deve definir se há um potencial suficiente para reuso e como o reuso será praticado nas diversas áreas da empresa. Nesse caso recomenda-se uma introdução progressiva do processo de reuso (EZRAN, 2002);
- **Não é possível definir se o reuso está sendo vantajoso, ou se o desenvolvimento do reuso está no caminho certo:** No nível departamental é difícil determinar como o reuso está melhorando os negócios da empresa. Os processos de reuso, custos e benefícios devem ser monitorados a fim de permitir uma melhoria contínua no processo e nos investimentos em reuso, ou seja, um processo de gerenciamento do reuso deve ser definido (EZRAN, 2002);
- **Necessidade de aplicar novos processos e técnicas e métodos avançados** (*design* de objetos, análise de domínio e *frameworks*, por exemplo) **para se alcançar o reuso:** Os times de desenvolvimento devem ter o suporte adequado a essas atividades. Um processo de suporte ao reuso é interessante a fim de centralizar os projetos, prover suporte técnico e gerenciar os componentes comuns (EZRAN, 2002);
- **Desperdício de esforço:** A criação de um componente custa cerca de 2 a 3 vezes mais que a criação de um componente não-reusável (TRACZ, 1995). É a chamada "Regra dos Três": antes de se receber algum benefício do reuso de um componente, deve-se reusá-lo pelo menos três vezes. Michel Ezran levantou estudos que apontam para um custo de desenvolvimento para reuso 50% maior em relação ao desenvolvimento não voltado ao reuso, porém afirma que o custo de reusar um componente chega a 70% do custo de desenvolvimento do mesmo componente, o que indica a necessidade de se reusar o componente no mínimo 5 vezes para que o reuso seja viável economicamente (EZRAN, 2002). Poulin apresenta, em seus estudos, valores variando de 1,1 a 3,7 como a quantidade de reusos necessários (POULIN, 1996). Mesmo que não se tenha certeza do número futuro de reuso, pode ser

interessante desenvolver o componente, mas deve-se estar ciente de que isso terá um custo maior que o desenvolvimento de um componente não reusável, e que esse custo poderá nunca ser recuperado (FRAKES, 1994);

- **Esforço mal direcionado:** Se existir um grupo separado para o desenvolvimento de componentes reusáveis, corre-se o risco do grupo chegar a desenvolver componentes que nunca sejam reusados. Exemplo: se o grupo tiver desenvolvido 4 componentes e tiver se enganado acerca dos requisitos e modelagem de 1 componente, de modo que este nunca seja reusado, então seguindo a Regra dos Três, mostrada no item anterior, cada componente deverá ser reusado pelo menos 5 vezes para que o custo seja compensado (MCCONNELL, 1996);
- **Evolução da tecnologia:** O próprio reuso planejado se torna um risco no sentido de que ele é uma estratégia de médio-longo prazo. Além de ter que reusar um componente várias vezes para que ele seja vantajoso, o componente deve ser reusado antes que a tecnologia no qual ele seja baseado se torne obsoleta. Num mundo onde a tecnologia evolui rápida e substancialmente, deve-se levar em consideração que o investimento num componente reusável hoje poderá se perder em cerca de 2 anos (MCCONNELL, 1996). Esse risco se agrava se o componente tiver alguma dependência de algum hardware ou software específico;
- **Superestimar os ganhos de tempo proporcionados pelo reuso:** Não se deve assumir que o reuso de código irá proporcionar grandes reduções de tempo no cronograma. Se somente código for reusado, somente o tempo de codificação será reduzido. Se o reuso for estendido a outros componentes então outras etapas do processo podem ser reduzidas também. Deve-se observar que o reuso de um componente também demanda tempo para encontrá-lo e entender como usá-lo. Recomenda-se planejar um quinto do tempo para o reuso de um componente, comparado ao tempo que seria usado para desenvolvê-lo completamente (TRACZ, 1995); e
- **Síndrome NIH:** O sucesso do reuso pode estar em risco caso o desenvolvedor seja acometido pela síndrome NIH, do inglês *Not Invented Here*, cujo sintoma é acreditar de forma irracional que o software desenvolvido por terceiros não é tão eficiente nem possui tão boa qualidade quanto o software desenvolvido pessoalmente (NORTON, 03). O software desenvolvido por terceiros deve ser avaliado e, caso passe nos critérios de aceitação, deve ser utilizado no projeto, pois seguramente os riscos

inerentes ao desenvolvimento interno serão maiores que os riscos assumidos ao se adotar o software externo.

2.2.4.5.1 RECOMENDAÇÕES PARA O SUCESSO DO REUSO

- **Concentrar no reuso de componentes de um domínio específico:** Se a linha de negócios da empresa se concentra em aplicativos financeiros, o programa de reuso deve se concentrar no desenvolvimento de componentes financeiros que sejam reusáveis (FRAKES, 1994);
- **É melhor focar o esforço na criação de componentes pequenos, bastante especializados, ao invés de se tentar criar grandes componentes, super genéricos, tentando abranger o maior número de aplicações possíveis:** O desenvolvedor de grandes componentes raramente consegue prever todos os requisitos futuros e quem olhar esses grandes componentes e ver que eles não atendem a todas as suas necessidades vai acabar desistindo de reusá-lo. Dados do Laboratório de Engenharia de Software da NASA sugerem que se um componente, para ser reusado, tiver que ser modificado em mais de 25%, então é mais barato desenvolvê-lo completamente (MCCONNELL, 1996);
- **Durante o desenvolvimento, focar em encapsulamento e abstração da informação, a base do *design* orientado a objetos:** Estudos mostram que o simples uso de linguagens orientadas a objeto não faz com que o reuso fique mais fácil de ser alcançado. Muitos projetos bem sucedidos em termos de reuso foram feitos com linguagens tradicionais de programação, como C e Cobol (MCCONNELL, 1996);
- **Prover uma boa documentação sobre o componente, sobretudo acerca de suas limitações, ou lista de falhas conhecidas:** É normal que um software seja liberado com algumas falhas de baixa severidade, segundo o contexto em que ele é usado, mas no caso de componentes reusáveis, uma falha de baixa severidade pode se tornar de altíssima severidade se o componente for reusado num contexto diferente (MCCONNELL, 1996);
- **Desenvolver componentes praticamente livres de falhas:** Se o desenvolvedor passa a encontrar falha nos componentes a serem reusados, ele vai desistir de reusar e todo

o programa de reuso da empresa pode ficar comprometido devido à falta de confiança nos produtos. Mesmo que o desenvolvedor corrija a falha, isso pode encarecer o custo de desenvolvimento de software pois, ao invés do desenvolvedor original estar depurando o componente, a empresa estará remunerando um outro desenvolvedor para fazer o mesmo serviço de forma bem menos eficiente, já que ele não está familiarizado com o código do componente (MCCONNELL, 1996); e

- **Focar na qualidade dos componentes e não na quantidade:** É melhor possuir uma biblioteca pequena de componentes e bastante confiável, do que uma grande biblioteca, com componentes de baixa qualidade (MCCONNELL, 1996).

2.3 DESENVOLVIMENTO DE SOFTWARE BASEADO EM REUSO NO CONTEXTO DO MINISTÉRIO DA DEFESA E DE SEUS COMANDOS SUBORDINADOS

Segundo Gurgel (2004), o Ministério da Defesa (MD) foi criado em 1º de janeiro de 1999 e representou a criação de um órgão único para integrar e centralizar os esforços dos antigos três Ministérios Militares, que foram convertidos em Comandos (MINISTÉRIO DA DEFESA, 2003). Sua estrutura administrativa está dividida em Secretarias e Departamentos, conforme pode ser observado no organograma apresentado na **FIG. 2.16**.

Nesta estrutura, é importante observar a existência do Departamento de Ciência e Tecnologia (DepCT) subordinado à Secretaria de Logística e Mobilização, que possui a missão de desenvolver atividades de ciência e tecnologia, em especial de projetos de integração tecnológica que transcendam os objetivos específicos e as disponibilidades dos orçamentos dos Comandos Militares (DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA, 2003), ou seja, houve, desde a criação do Ministério, a definição do DepCT como órgão centralizador dos assuntos de ciência e tecnologia nas Forças Armadas.

Para o cumprimento desta tarefa, foi criada, pelo Decreto Nº 3.569/2000, a Comissão Assessora de Ciência e Tecnologia para a Defesa (COMASSE). Trata-se de um órgão colegiado presidido pelo DepCT que tem como responsabilidade propor e consolidar os projetos de Pesquisa e Desenvolvimento (P&D). Esta comissão realiza a integração dos esforços não somente das Forças Singulares, mas também deste com o Sistema Nacional de Ciência e Tecnologia e com a Indústria Nacional de Defesa. Para tal, é constituída de

representantes dos Comandos Militares, dos Ministérios da Ciência e Tecnologia (MCT), do Orçamento, Planejamento e Gestão (MPOG), e do Desenvolvimento, Indústria e Comércio Exterior (MDICE).

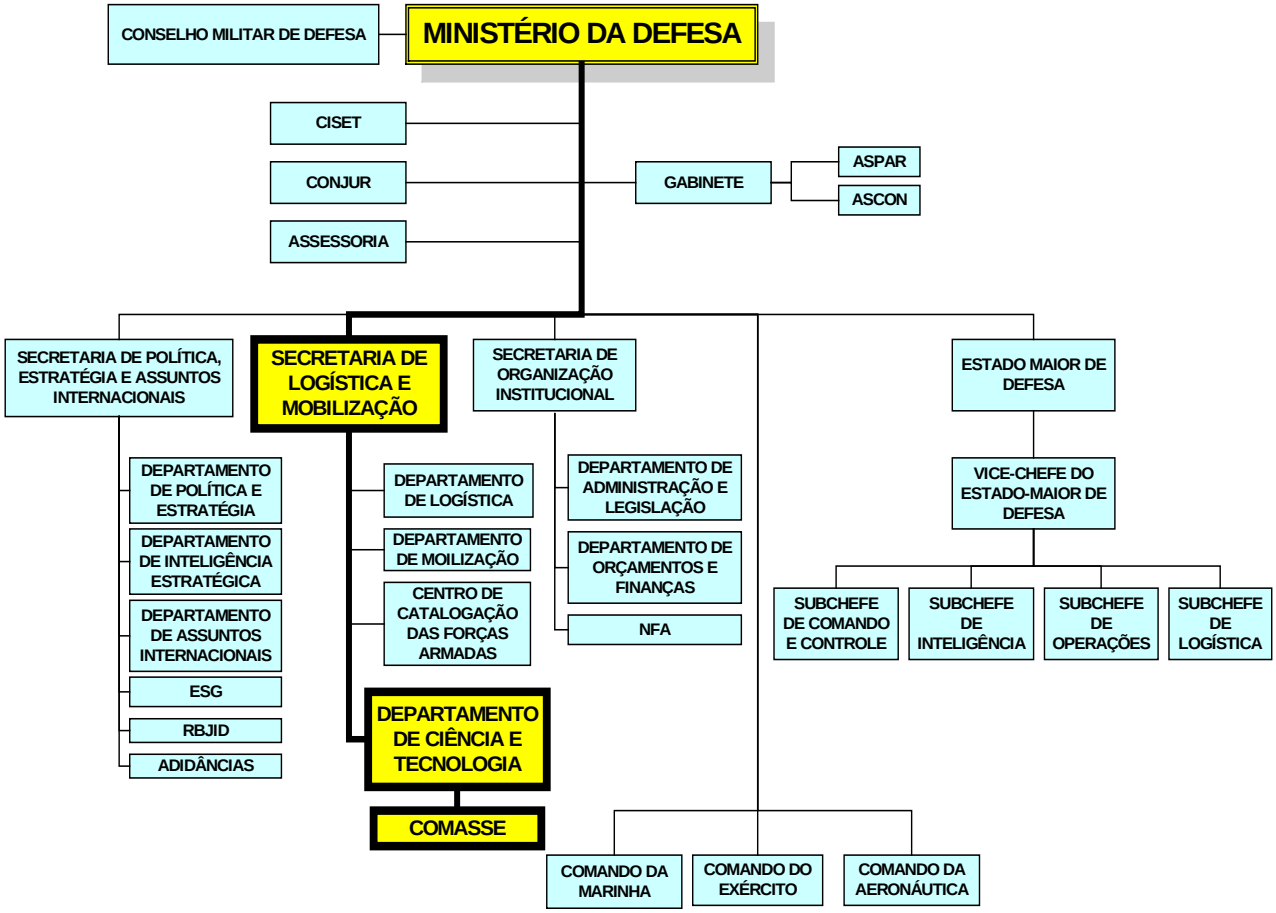


FIG. 2.16 - Organograma do Ministério da Defesa.

2.3.1 CICLO DE VIDA DO SOFTWARE BASEADO EM REUSO

Segundo GURGEL (2004), o conceito por trás do novo paradigma de reuso é a necessidade de se criar um ambiente de desenvolvimento onde famílias de sistemas relacionados em um domínio sejam projetadas de forma a possuírem uma estrutura em comum que é um dos pilares para o reuso (THE ARMY STRATEGIC SOFTWARE REUSE, 1992). Os três princípios fundamentais intrínsecos a este conceito são:

1. Focar o reuso em domínios específicos e explorar estes domínios para suportar reuso em larga escala, assegurando que o reuso passe a ser tratado como parte inseparável da engenharia de software;
2. Empregar arquiteturas orientadas ao reuso para estimular o investimento em componentes que povoem estas arquiteturas; e
3. A estratégia para implementar estas mudanças seja baseada em um reuso sistemático, onde existe definido um processo para captar oportunidades de reusabilidade.

Desta forma, temos a necessidade da definição de um ciclo de vida do software baseado em reuso para gerar este ambiente de desenvolvimento. Para que isto seja feito, três princípios básicos devem ser incorporados a este ciclo de vida a fim de promover e viabilizar o reuso (IEEE Std 1517, 1999):

1. Os temas de reuso devem governar as decisões de desenvolvimento de software: tomando proveito de *assets* já existentes, identificando oportunidades e preparando-se para o reuso.
2. Os requisitos de reuso mudam o foco de um projeto isolado para conjunto de projetos: alterando a forma de definir requisitos a fim de compartilhar com outros sistemas de software, presentes ou futuros, as possibilidades de reaproveitar sistemas legados e definir planos a nível estratégicos para o desenvolvimento de software, além da exploração de pontos comuns e acomodações de variabilidades entre os produtos de software para identificar possíveis assets.
3. A visão do reuso muda a forma de ver o desenvolvimento de software, dividindo-o em duas partes: desenvolvimento com o reuso, que identifica possibilidades de se utilizar assets para gerar um sistema de software, e desenvolvimento para o reuso, com a engenharia de domínios e o desenvolvimento de assets reutilizáveis.

O Ministério da Defesa, conforme apresentado, possui uma estrutura hierarquizada, centralizada, e com vários órgãos de desenvolvimento em seus Comandos, sendo que cada um destes órgãos atua como especialista em um conjunto de domínios que representa a sua área de atuação. O que se propõe a fim de incorporar as características necessárias ao ciclo de vida do software a ser empregado no MD é aproveitar esta estrutura organizacional existente e centralizar os processos de aquisição, gerência de *assets* e gerência de reuso a fim de atender a estes princípios definidos e com isto garantir o desenvolvimento de uma estrutura comum entre os sistemas desenvolvidos ou comprados. Analisemos como a centralização de cada um destes processos influencia no ciclo de vida:

A centralização da gerência de *assets* possibilita a padronização dos *assets* gerados, facilita a busca de *assets* candidatos passíveis de serem utilizados nos processos de engenharia de Domínios e de desenvolvimento e disponibiliza estes *assets* aos órgãos de desenvolvimento descentralizados.

- A centralização do processo de aquisição e da gerência de reuso induzem a realização do reuso e formalizam o apoio da organização ao programa de reuso. Em paralelo, possibilita que aquisições efetuadas junto ao mercado, ou seja empresas externas à organização, atendam aos mesmos preceitos e sejam capazes de gerar não somente o aplicativo, mas também *assets* reusáveis.
- A visualização dos órgãos descentralizados de desenvolvimento como fornecedores, sendo tratados da mesma maneira que uma empresa de desenvolvimento externa à organização, estimula a competitividade ao mesmo tempo em que permite que seja mantido e que se tome proveito da larga experiência que estes órgãos possuem nas suas áreas de atuação onde são especialistas de domínio.
- A gerência de reuso centralizada também auxilia a resolver problemas de definição de domínios horizontais entre órgãos de desenvolvimento.

O ciclo de vida de software proposto a seguir não busca especificar todas as etapas necessárias para o desenvolvimento de software baseado em reuso, mas apresentar a definição e distribuição de papéis, processos e atividades envolvidas, assim como os artefatos e *assets* a serem gerados. Ele é um modelo de processo a ser seguido na implantação de um reuso sistemático com o proveito das características próprias do contexto considerado.

2.3.1.1 VISÃO GERAL

Conforme pode ser visto na **FIG 2.17**, o ciclo de vida do software baseado em reuso inicia com o cliente, que é um setor que possui a necessidade de um novo aplicativo, acessando diretamente à Gerência de *Assets*, no órgão central denominado Gerência de Reuso, a fim de verificar a existência de *assets* candidatos. Em seguida, ele prepara e envia para este órgão, a solicitação de um novo aplicativo com as Necessidades do Cliente (definindo o domínio, funcionalidades e características desejadas) e o Glossário com os termos usados no domínio da aplicação.

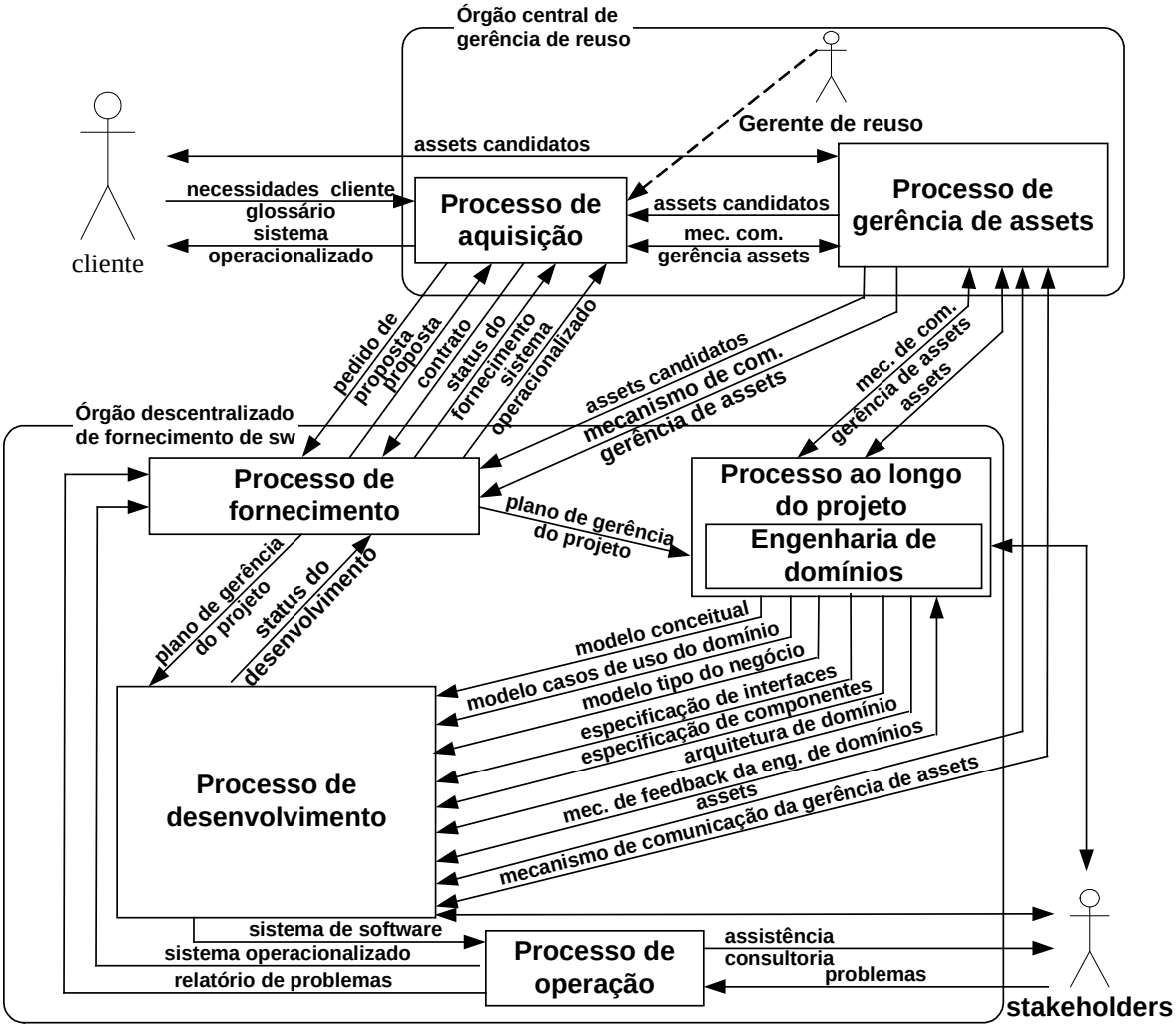


FIG. 2.17 - Ciclo de vida do software baseado em reuso no contexto do MD (GURGEL, 2004).

Esta gerência realiza o processo de aquisição analisando os *assets* existentes e as possibilidades de atendimento, prepara o plano de aquisição, e, uma vez autorizado, realiza o contrato com um dos órgãos de fornecimento descentralizados, interno ou externo à organização. No contrato, deverão constar as necessidades do cliente, glossário, *assets* candidatos e critérios de aceitação, além de outras informações julgadas necessárias.

O órgão descentralizado, seguindo o processo de fornecimento, analisa as opções de fornecimento e gera o plano de gerência do projeto. Este plano é enviado às equipes de engenharia de domínio e de desenvolvimento.

A equipe de engenharia de domínios, seguindo o processo de engenharia de domínios ou processo ao longo dos projetos, conforme denominado pela norma IEEE 1517, analisa os *assets* do domínio existentes verificando se atendem aos requisitos da aplicação. Caso não existam *assets* referentes ao domínio, ou estes não atendam, serão projetados e desenvolvidos novos *assets* de domínio, ou novas versões. Importante definir este processo como iterativo, pois a cada nova aplicação ou retorno do mecanismo de feedback da engenharia de domínio, deverá ser realizada uma revisão dos *assets* deste domínio e enviadas as alterações à gerência de *assets*.

A equipe de desenvolvimento é dividida em desenvolvedores e montadores. Os desenvolvedores implementam novos *assets* de sistema necessários à montagem do aplicativo, e não disponíveis, segundo a especificação da engenharia de domínios e os envia à gerência de *assets.* Os montadores, uma vez tendo recebido o plano de gerência e os *assets*, realizam a montagem e teste do aplicativo. Após isto, o aplicativo é encaminhado à equipe de operacionalização de sistemas.

A equipe de operacionalização, seguindo o processo de operação, irá disponibilizar o sistema ao usuário, instalando-o e preparando sua documentação de operação, dando suporte e resolvendo eventuais problemas que surjam durante esta operação.

O gerente de *assets*, segundo o processo de gerência de *assets*, será o responsável por fornecer e armazenar os *assets* utilizados durante todo o ciclo de vida do software, assim como registrar problemas e informações advindas dos processos através do mecanismo de comunicação com o gerenciador de *assets.*

2.3.2 GARANTIA DA QUALIDADE

Segundo MOORE (2004), o processo de Garantia da Qualidade de *Software* tem como propósito estabelecer um conjunto de diretrizes para o planejamento e a execução de um processo de avaliação de um produto de *software* ao definir as atividades que devem ser executadas.

O Processo de Garantia da Qualidade de *Software* (MOORE, 2004) se inicia juntamente com o Plano de Aquisição[3] pelo Grupo de Garantia da Qualidade de *Software* Corporativo através do artefato Necessidades do Cliente e Glossário[4] fornecido pelo Cliente no Processo de Aquisição (veja **FIG. 2.17**). A partir deste artefato, o Grupo de GQS Corporativo deve criar e documentar um Manual da Qualidade de Projeto, onde devem ser especificados os objetivos para assegurar a qualidade dos *assets* e processos utilizados para desenvolver os produtos de *software* (artefatos) que vão compor estes *assets*, além dos objetivos para garantir a satisfação do cliente. Neste manual também deve estar especificado o escopo da avaliação (das verificações, revisões e medições a serem realizadas no produto), os métodos utilizados e as responsabilidades de todos envolvidos no processo de avaliação.

Na **FIG. 2.18** são apresentadas as interações do Processo de Garantia da Qualidade de *Software* (MOORE, 2004) com os processos de GURGEL (2004), especificando os artefatos de entrada/saída com seus respectivos papéis e os responsáveis pelo PGQS.

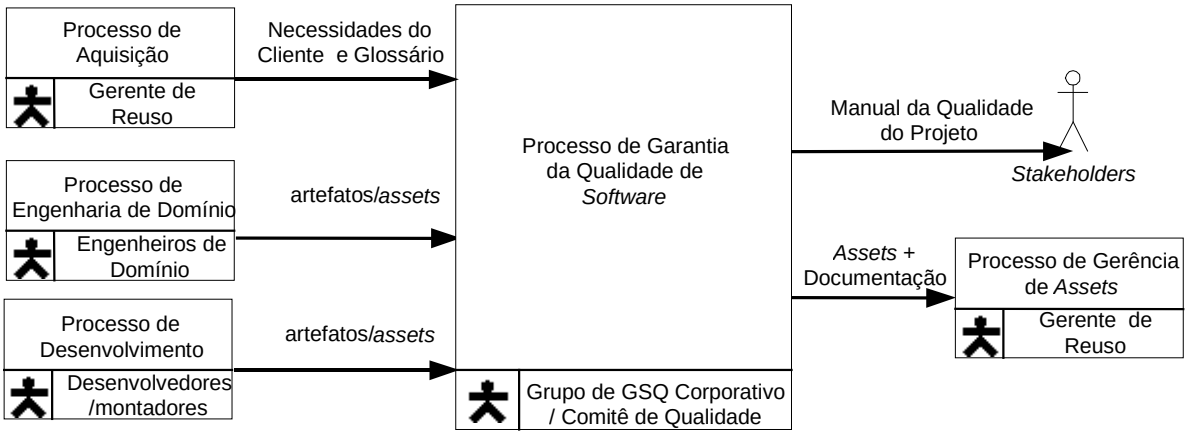


FIG. 2.18 - Processo de Garantia da Qualidade de *Software*.

O Grupo de GQS Corporativo envia este manual junto com o Pedido da Proposta ao Comitê da Qualidade do Órgão de Fornecimento de *Software* interno ou externo à organização para que este comitê possa estabelecer o Grupo de Garantia de Qualidade de *Software*, e definir um Plano e Cronograma de Garantia da Qualidade para o projeto em

[3] Artefato da atividade Preparação do Plano de Aquisição do Processo de Aquisição do Processo de Desenvolvimento de *Software* baseado em Reuso para MD de GURGEL (2004).

[4] O artefato Necessidades do Cliente e Glossário define o domínio do problema, funcionalidades e características desejadas, e glossário de termos do domínio da aplicação.

questão. Tanto o plano como o cronograma devem ser integrados pelo Plano de Gerência de Projeto[5].

Para cada referência estabelecida pelo Plano de Gerência de Projeto, este processo deve especificar os itens a serem verificados, revisados e armazenados de acordo com o Manual da Qualidade do Projeto e o Plano de Garantia da Qualidade do *Software* do Projeto (PGQSP). Este plano deve ser implementado e mantido durante a vigência do Contrato do Projeto[6] (ABNT12207, 1998; ABNT9000, 2001), e deve estabelecer os seguintes itens:

Responsável pelas atividades de GQS no projeto: identifica o nome do consultor de GQS alocado ao projeto pelo Comitê de Qualidade;

Padrões e procedimentos para as atividades de GQS: o consultor de GQS deve utilizar, na execução das atividades de GQS no projeto de *software* e na realimentação acerca destas atividades, os padrões e procedimentos estabelecidos no Plano de Gerência do Projeto versão <X.X>. As responsabilidades e autoridade do grupo de GQS devem estar definidas no Documento de Implementação de GQS do Órgão Militar;

Padrões e procedimentos de projeto: informa os padrões e procedimentos que devem ser seguidos pelo projeto de *software* a ser verificado / revisado pelo verificador / revisor no Plano de Gerência do Projeto;

Ferramentas: informa a ferramenta, caso haja, que deve ser utilizada para registro e acompanhamento das atividades de GQS. Esta ferramenta deve fornecer o apoio necessário para a documentação e o rastreamento dos itens de não-conformidade até sua conclusão;

Treinamento: necessidade de algum treinamento específico em GQS requerido por alguma circunstância no contexto do projeto. Esse item é opcional;

Histórico de versões: informa dados de histórico das versões anteriores e da atual do artefato, tais como data da versão, número seqüencial da versão, breve descrição do motivo da versão, autor responsável pela versão, o nome do revisor e por quem foi aprovado;

Cronograma de verificações / revisões: informa o cronograma para as verificações / revisões de GQS no projeto e o esforço em homens-hora empregado pelo consultor de GQS e pela equipe de desenvolvimento que participam dessas atividades.

As não-conformidades às necessidades do cliente detectadas no Processo de Controle da Qualidade devem ser registradas nos Relatórios de Verificação ou Revisão para análise e avaliação. Além disso, após a elaboração de cada relatório, um questionário de solicitação de

---

[5] Este artefato é produto da atividade Preparação do Plano de Gerência de Projeto do Processo de Fornecimento proposto por GURGEL (2004).

[6] Este artefato é produto do Processo de Aquisição proposto por GURGEL (2004).

sugestões de melhoria da qualidade deve ser respondido e comentado para capturar e aplicar as lições aprendidas no controle da qualidade de forma a contribuir para melhorar este processo. O preenchimento deste questionário pelo executor da verificação / revisão não deve ultrapassar a quinze minutos.

Os registros das atividades e das tarefas de Garantia da Qualidade, todos os artefatos produzidos para avaliação e os resultados obtidos, os registros dos problemas detectados e a resolução destes problemas que surgiram durante todo o ciclo de vida de desenvolvimento do Projeto, inclusive o Plano e Cronograma de Garantia da Qualidade, devem ser mantidos e disponibilizados a todos interessados (*stakeholders*). Para tal, um Repositório de Qualidade do Projeto deve ser criado para guarda e acesso desses documentos e registros.

É importante ressaltar que todos os modelos dos artefatos do Processo de Garantia da Qualidade de *Software* devem estar disponíveis no Repositório de Qualidade do Projeto a toda equipe de desenvolvimento/montadores a qualquer momento para que, assim, estes modelos de artefatos possam contribuir como suporte na elaboração dos produtos de *software* das atividades dos processos de Engenharia de Domínio e Desenvolvimento de GURGEL (2004).

2.4 CONCLUSÕES

Este capítulo apresentou a Revisão Bibliográfica onde foram discutido os seguintes conceitos:

- ***Reusable Software Assets* (RAS)** – especificação definida pela OMG que descreve os tipos de *assets* existentes e o seu Meta-Modelo onde é apresentado os elementos fundamentais da especificação e as suas extensões;
- **Processo de Desenvolvimento de Software Baseado em Reuso** - esse tópico abordou o conceito de processo e a linguagem de modelagem de processos SPEM utilizada neste trabalho. Apresentou também a norma ISSO/IEC 12207 e a sua extensão, a norma IEE 1517 (1999), que fornece uma estrutura comum de processos para ser utilizada como referência no desenvolvimento de produtos e serviços de *software.* Por fim foi apresentado o processo de reuso e as implicações e necessidades para se desenvolver sistemas para/com reuso, tais como:
  - Definição de um repositório;

- Métodos de classificação de *assets;*
- Catalogação de *assets*;
- Gerência da Qualidade;
- Seviços de suporte ao processo de gerência de *assets*;
- Elaboração de Métricas; e
- Risco na implatanção do reuso.

▪ **Desenvolvimento de Software Baseado em Reuso no Contexto do Ministério da Defesa e de seus Comandos Subordinados** – por útimo, esse tópico apresentou a estrutura hierárquica do Ministério da Defesa, o conceito de reuso dentro desta estrutura, o Processo de Desenvolvimento de Software (GURGEL, 2004) e o Processo de Qualidade (MOORE, 2004) dentro desse contexto.

3 PROCESSO DE GERÊNCIA DE *ASSETS* NO DESENVOLVIMENTO DE *SOFTWARE* BASEADO EM REUSO

Com o objetivo de garantir o reuso dos *assets* gerados no ciclo de vida do desenvolvimento de software baseado em reuso, este trabalho apresenta uma proposta de um Processo de Gerência de *Assets* também denominado de PGA. O processo proposto estabelece um *framework* para auxiliar nas atividades de gerência, produção e consumo de *assets*, com o explícito propósito de contribuir para a qualidade, produtividade e sucesso do projeto de reuso dentro do contexto do Ministério da Defesa e de seus Comandos Subordinados.

Gurgel descreve sobre o contexto do Ministério da Defesa desde sua criação em 1999 à formação do Departamento de Ciência e Tecnologia nas Forças Armadas e, também, apresenta a atual estrutura dos Comandos da Marinha, Exército e Aeronáutica (GURGEL, 2004). O Ministério da Defesa é uma instituição de grande porte, hierarquizada e com núcleos de desenvolvimento descentralizados conforme apresentado em GURGEL (2004).

Este trabalho estabelece um modelo de processo que garanta a reusabilidade dos *assets* gerados no Processo de Desenvolvimento de *Software* Baseado em Reuso de GURGEL (2004) e certificados pelo Processo de Garantia da Qualidade de *Software* de Moore (2004).

GURGEL (2004) e MOORE (2004), baseados nos produtos de *software* de (CHEESMAN et al., 2000), geraram os artefatos dos processos de Engenharia de Domínio e Desenvolvimento do Controlador Tático, que foram os artefatos utilizados para a definição do processo proposto nessa dissertação.

Este trabalho está baseado na definição do propósito da Gerência de *Assets*, descrito na norma ISO/IEC 1207, na definição do Processo de Gerência, descrito na norma IEE 1517 (1999) e nos serviços de suporte ao Processo de Gerência, definido por Apperly (2001).

Com base na definição do propósito do Processo de Gerência de *Assets*, ao longo deste capítulo, iremos especificar todos os itens necessários para a elaboração do processo proposto neste trabalho. São eles:

- Estrutura do Processo;
- Entradas e Saídas do Processo;

- Equipe de Gerência de *Assets;*
- Atores do Processo;
- Repositório de Dados;
- Método de Classificação;
- Estrutura de Armazenamento;
- Atividades do Processo; e
- Artefatos de Saída.

3.1 ESTRUTURA DO PROCESSO DE GERÊNCIA DE *ASSETS*

Como o Ministério da Defesa é uma instituição centralizadora e hierárquica, que possui Comandos Subordinados, e estes comandos, por sua vez, contém vários órgãos de desenvolvimento com diferentes áreas de atuação, este trabalho propõe, de acordo com a abordagem estabelecida em GURGEL (2004), uma estrutura organizacional para Processo de Gerência de *Assets* (PGA).

A **FIG. 2.17,** da seção anterior, está representada a organização estrutural do Processo de Desenvolvimento Baseado em Reuso, por Gurgel (2004), onde é apresentado o Processo de Gerência de *Assets* dentro desse contexto.

Da mesma forma, na **FIG. 2.18,** da seção anterior, está representada a organização estrutural do Processo de Qualidade de *Software* por Moore (2004), e a comunicação com o Processo de Gerência de *Assets.*

O Processo de Gerência de *Assets* fornece suporte ao Processo de Desenvolvimento de Software. A Equipe de Gerencia de *Assets*, é o grupo responsável pela "Gerência de Reuso", e o seu objetivo é assegurar o sucesso do reuso dentro do Contexto do Ministério da Defesa. O Processo de Gerência de *Assets* pertence ao Órgão Central de Gerência de Reuso e no escopo deste trabalho cabe a ele garantir a reusabilidade dos *assets* gerados na fase de desenvolvimento de um sistema por meio das atividades de armazenamento, recuperação, gerência e controle dos mesmos.

Com base nas atribuições do processo proposto, apresentamos na **FIG. 3.1** as interações do Processo de Gerência de *Assets* (PGA), foco deste trabalho, com os Processos de

Desenvolvimento e Qualidade de *Software*, especificando as entradas e saídas com seus respectivos papéis e os responsáveis pelo PGA.

O início do processo acontece após o artefato ser produzido no desenvolvimento de *software*, por um dos processos existente no ciclo de vida de desenvolvimento de sistemas, e enviado para o Grupo de Qualidade de *Software*. Este grupo certifica o *asset* candidato e envia para o Processo de Gerência de *Assets.* Da mesma forma, durante o desenvolvimento de um sistema, qualquer um dos processos responsáveis por produzir software pode consultar o Processo de Gerência de *Assets* acerca dos *assets* que possam ser reutilizados no desenvolvimento do novo produto (ver **FIG. 3.1**).

Desta forma podemos ressaltar que o processo definido neste trabalho, o Processo de Gerência de *Assets* (PGA), é responsável por gerenciar a vida dos *assets* desde a sua concepção até à sua retirada do sistema (ISO/IEC 12207:1995/FDAM 1) e oferecer recursos para que os usuários do sistema possam reutilizá-los

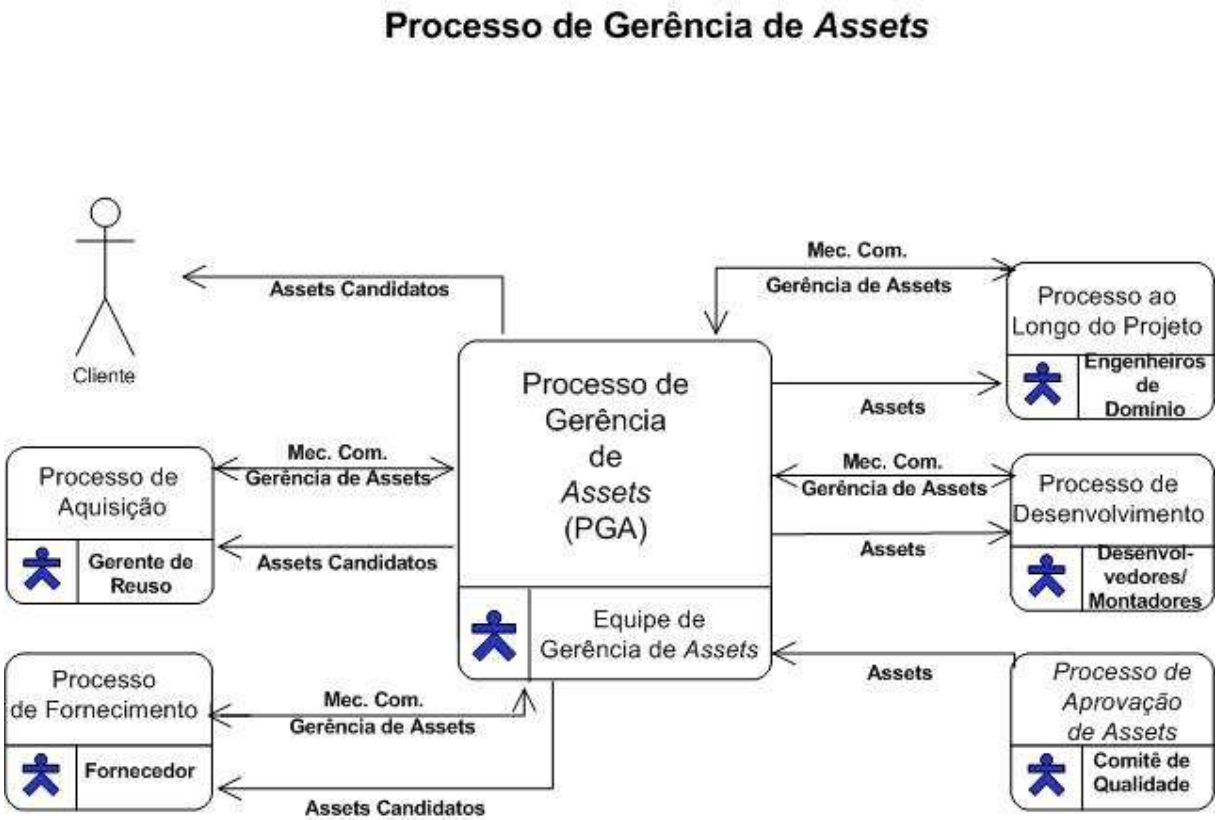


FIG. 3.1: Interações entre o processo proposto nesse trabalho, PGA (ALMEIDA, 2007), e os Processos de Desenvolvimento e Qualidade de *Software.*

3.2 ENTRADAS E SAÍDAS DO PROCESSO

Conforme a estrutura do processo apresentada na **FIG. 3.1** podemos definir as seguintes entradas e saídas necessária para a execução do processo:

- **Assets Candidatos (saída) –** lista de *assets* contendo os possíveis *assets* à serem reutilizados;
- **Mec. Com. Gerência de *Assets* (entrada/saída)** – provê um meio de comunicação entre a gerência de *assets* e os usuários do sistema como por exemplo o registro de problemas, informações advindas dos processos (Aquisição, Fornecimento, Ao Longo do Projeto e Desenvolvimento), bem como qualquer outra notificação entre o PGA e seus usuários; e
- ***Assets* (entrada/saída)** – artefato ou conjunto de artefatos, documentado, certificado e pronto para o reuso.

3.3 EQUIPE DE GERÊNCIA DE *ASSETS*

Para definirmos o processo de reuso, novas responsabilidades devem ser atribuídas de forma a assegurar que os procedimentos serão seguidos e monitorados corretamente. Nesta proposta novos papéis foram criados e especificados para serem distribuídos entre as pessoas dentro do Órgão Central de Gerência de Reuso (**FIG. 2.17**). É importante ressaltar que uma pessoa pode assumir mais de um papel dentro do processo de reuso (EZRAN, 2002).

No processo proposto criamos os seguintes papéis:

- **Gerente do Repositório –** Responsável por classificar, catalogar os *assets* e gerenciar tanto a adição (produção), quanto à utilização (consumo) da biblioteca de *assets*;
- **Gerente de Qualidade –** Responsável por garantir que os *assets* publicados no repositório satisfaçam determinados padrões de qualidade; e

- **Gerente de Configuração** - Responsável pelo controle de usuários, controle de versões, pela gerência das modificações feitas nos *assets*, pelo registro dos problemas detectados e pela elaboração do relatório das métricas do processo.

3.4 ATORES DO PROCESSO

Os usuários do Processo de Desenvolvimento de Software Baseado em Reuso (GURGEL, 2004) desempenham os diferentes papéis do Processo de Gerência de *Assets*. Esses usuários podem ser tanto produtores como consumidores de *assets* dentro do processo proposto nesse trabalho.

Os produtores de *assets* estabelecem as responsabilidades relativas à atividade de produção. A inserção de um *asset* totalmente novo pode ser proveniente do desenvolvimento interno de software, da reutilização de sistemas legados ou da encomenda de produtos para terceiros. Nesta proposta estas atividades são desempenhadas pela Equipe de Engenharia de Domínio e pela Equipe de Desenvolvedores.

Os consumidores são os participantes de um projeto interessados na reutilização de algum *asset* que atenda as suas necessidades. Executam as tarefas pertinentes à atividade de consumo, como pesquisa no repositório, reutilização de um *asset* e de suas especificações. Nesta proposta os consumidores são Clientes, Processo de Aquisição, Processo de Fornecimento, Equipe de Engenharia de Domínio e Equipe de Desenvolvedores.

No processo proposto temos os seguintes papéis:

- **Clientes** – Um dos órgãos de desenvolvimento subordinados ao Ministério da Defesa;
- **Processo de Aquisição** – Seu propósito é obter o produto ou serviço que atenda às necessidades expressas pelo cliente;
- **Processo de Fornecimento** - Desenvolvido pelo órgão interno ou externo à organização que apresenta uma proposta para prover um sistema, produto de software, ou *asset* para um adquirente, nesta situação, este órgão é chamado de fornecedor;
- **Equipe de Engenharia de Domínio** – Responsáveis por desenvolver e manter modelos de domínio, arquiteturas de domínio e componentes de domínio; e

- **Equipe de Desenvolvedores** – É executado por dois grupos: o de desenvolvimento e o de montagem. O grupo de desenvolvimento é o responsável por criar os *assets* de código a partir da especificação definida nos *assets* de componentes de domínio. O grupo de montagem é o responsável por montar o aplicativo a partir dos *assets* de código.

3.5 REPOSITÓRIO DE DADOS

O repositório de *assets* é definido como uma base de dados, cujo conteúdo foi projetado para permitir classificação, armazenamento, busca e recuperação da informação relacionada aos *assets*, tais como: *assets* de domínio, *assets* de arquitetura de domínio, *assets* de componente de domínio, e componentes.

Visando a implementação de um repositório de *assets* propomos neste trabalho um método de classificação e uma estrutura de armazenamento definidos a seguir.

3.5.1 MÉTODO DE CLASSIFICAÇÃO

Conforme definido no capítulo anterior a classificação é uma atividade necessária para auxiliar no processo de busca e recuperação dos *assets* em uma base de armazenamento. O esquema de classificação permite a definição de um método para a categorização do *asset*, de modo que, possibilite a busca e a localização de um *asset* de um modo mais fácil e rápido.

Desta forma, entre os métodos apresentados anteriormente, o método de classificação por facetas (PRIETO-DÍAZ e FREEMAN, 1987) foi o escolhido para a classificação dos *assets* no processo aqui proposto.

A escolha do método de classificação por facetas, levou em conta que o estudo de métodos de classificação não foi o foco principal da pesquisa acadêmica e resultou devido aos motivos descritos abaixo:

- Apresenta maior flexibilidade, permitindo a expansão dos elementos do esquema de classificação;

- Ela é intuitiva, adequada ao domínio de artefatos de software, de fácil manutenção e uso, e passível de automação (LUCRÉDIO, 2005);
- Possibilita a criação de relações complexas através da combinação de facetas e termos;
- Permite o tratamento de sinônimos com mais facilidade que os outros métodos apresentados;
- Permite estabelecer uma relação de similaridade entre os termos de uma faceta, através do grafo de proximidade conceitual; e
- Este método tem sido referenciado e tem sido base de outras propostas de classificação de informação encontradas na literatura sobre reutilização [(GIOVANI; MELNIKOFF, 1996), (DAMIANI; FUGINI; FUSASCHI, 1997), (MCCLURE, 1998), (CASANOVA; STRAETEN; JONCKES, 2003), (VITHARANA; ZAHEDI; JAIN, 2003) (LUCRÉDIO, 2005)], o que justifica a sua adoção.

No processo proposto, considerando os diferentes tipos de *assets* que podem ser catalogados foi elaborada a seguinte relação de facetas que devem auxiliar na classificação dos *assets* no repositório:

- **Domínio** - Especifica a área que é atendida pelos serviços oferecidos pelo *asset*;
- **Aplicação** - Especifica qual a finalidade dos serviços oferecidos dentro do domínio do *asset*;
- **Tipo de *Asset*** - Tipo de *asset* produzido pelo processo de desenvolvimento de software baseado em reuso no Ministério da Defesa;
- **Atividade** - Em que atividade (fase do desenvolvimento) o *asset* foi desenvolvido;
- **Tipo de componente** - Especifica o tipo de componente de software, ou seja, sub-rotina, classe, pacote, *framework* ou sistema;
- **Comando** - Comando subordinado ao Ministério da Defesa - Aeronáutica/Exército/Marinha;
- **Tipo de Sistema** - Sistema Científico (SC) ou Sistema Administrativo (SA); e
- **Descrição** - Conjunto de palavras que descrevam a finalidade do *asset*.

Foi definida nesta proposta que a elaboração do critério de busca no repositório será baseada nas oito facetas apresentadas acima. O usuário poderá formular a busca selecionando os termos de cada faceta, podendo não selecionar critérios para todas as facetas, desta forma, construindo uma pesquisa mais abrangente.

3.5.2 ESTRUTURA DE ARMAZENAMENTO

A estrutura de armazenamento utilizada neste processo foi baseada na *Reusable Asset Specification* (RAS, 2005). Como apresentada na sub-seção 2.1, a RAS define um formato de armazenamento baseado em meta-informações a respeito de uma *asset*. Neste, os *assets* podem ser armazenados como uma estrutura de pasta conforme apresentado na **FIG. 3.2**. Os arquivos XML são utilizados para guardar os dados referentes às pastas e os XSD os seus meta-dados.

FIG. 3.2: Estrutura de armazenamento baseada na *Reusable Asset Specification* (RAS, 2005).

As pastas *Profile*, Classificação, Solução, Utilização e *Assets* Relacionados conterão as informações provenientes da documentação que acompanha o *asset*.

Além da estrutura de armazenamento especificada na RAS (2005), nesta proposta, criamos uma outra pasta para armazenar as informações referentes às diferenças entre as versões de um *asset.*

Definimos que a documentação dos *assets* siga o padrão definido por SAMETINGER (1996) e deverá conter os seguintes itens:

Características Gerais:

- **Nome do *asset*:** nome, dando uma possível sugestão sobre a funcionalidade do *asset*;
- **Identificação:** declaração inicial concisa e clara sobre a funcionalidade do *asset* para seleção inicial, e inclusive com descrição geral de todas as operações externamente visíveis, se for um componente;

- **Especificação:** funcionalidade do *asset* que deve ser detalhada;
- **Classificação:** classificação do *asset* como uma lista de palavras chaves para habilitar recuperação futura, por exemplo, se *asset* for um componente, indicar o tipo (classes de C++, funções de C, aplicação de OpenDoc).

Características de Reutilização

- **Usabilidade:** descrição de como o *asset* pode ser utilizado corretamente;
- **Instalação:** descrição de como adaptar o *asset* em uma nova aplicação;
- **Adaptação:** descrição de como e para que necessidade específica pode ser adaptado o *asset*;
- **Implementação:** descrição de como o *asset* está implementado;
- **Suporte**: onde adquirir ajuda, se necessário for, por exemplo, ao adaptar o componente ou o que fazer se um problema ocorrer.

Características Administrativas

- **Histórico:** histórico de versões, datas de liberações, desenvolvedores responsáveis, principais diferenças entre as versões;
- **Restrições comerciais e legais:** restrições comerciais ou legais no uso do componente, por exemplo, aquisição, licença especial ou permissão requerida.

Características de Avaliação

- **Restrições técnicas:** restrições técnicas no uso do *asset*, por exemplo, capacidades, linguagem de programação, dependências de sistema operacional;
- **Qualidade:** indicar a situação do *asset*, quanto à qualidade, revisões, verificações e casos de testes aplicados, inclusive os resultados obtidos;
- **Performance:** recursos do sistema necessários para usar o *asset*, por exemplo, memória, processador, canais de comunicação;
- **Alternativas:** relação de *assets* semelhantes que poderiam ser usados no lugar deste;
- **Problemas:** relato de problemas conhecidos e custo requerido;
- **Interdependências:** o *asset* pode ser usado *stand-alone* ou deve ser usado junto a outros *assets*;
- **Custo recomendado:** custo conhecido para melhorar performance, tornar o *asset* mais robusto e / ou estender o escopo de reuso.

Características Complementares

- **Indexação:** provê um índice para *assets* complexos que requerem documentação extensa;
- **Referências:** referências para literatura ou outra documentação, por exemplo, documentação de sistemas.

3.6 ARTEFATOS GERADOS DURANTE A EXECUÇÃO DO PROCESSO

Artefato é um pedaço de informação que é produzida, modificada ou usada por um processo. Os artefatos são os produtos tangíveis do projeto: as coisas que o projeto produz ou usa enquanto trabalha para o produto final.

Neste trabalho elaboramos os artefatos de saída produzidos durante a execução de cada atividade do processo proposto.

A execução das atividades do processo irá gerar artefatos de saída como resultado de cada atividade. Além de especificarmos todas atividades do processo também especificamos todos os artefatos de saída de cada atividade.

Abaixo segue a relação de todos os artefartos:

- Lista de Usuários;
- Lista de *Assets* Candidatos;
- Notificações;
- Controle de Versões;
- Classificação do *Asset*;
- Catálogo dos *Asset;*
- Relatório de Métricas; e
- Registro de Problemas.

Os *templates* dos artefatos gerados serão apresentados na sub-seção 3.7, Atividades do Processo, juntamento com com os diagramas de cada atividade do processo.

3.7 ATIVIDADES DO PROCESSO

O Processo de Gerência de *Assets*, segundo definição da IEEE 1517 (1999), é o responsável por executar as atividades necessárias para gerenciar a certificação, classificação, armazenamento, busca, controle de versões e controle de alterações dos *assets*.

A proposta apresentada visa fornecer suporte ao processo de desenvolvimento de sistemas na MD que emprega a reutilização de *assets*, os quais são armazenados em um repositório e, por meio de mecanismos busca, recuperação e visualização da informação relativa ao *asset*, possibilita o compartilhamento, a divulgação e o entendimento destes *assets* no processo de desenvolvimento de sistemas.

Em linhas gerais, este trabalho fornece uma infra-estrutura de apoio à reutilização de *assets* através de um repositório. O processo proposto define as regras para as três atividades do processo de reutilização: consumo e produção em uma biblioteca de *assets*, e a manutenção (gerência) da mesma. Neste contexto, o objetivo do processo especificado neste trabalho é armazenar, catalogar e difundir os *assets* para construção de novos sistemas no contexto da MD, através de recursos para busca e recuperação de *assets*, entendimento da sua funcionalidade, armazenamento da documentação relativa ao *asset*, monitoramento do histórico das reutilizações efetuadas, e notificação de atualizações nos *assets* disponíveis no repositório.

O processo proposto oferece suporte aos serviços que estão em destaque na **TAB. 3.1**.

TAB. 3.1: Serviços que o Processo de Gerência de *Assets* fornece suporte (baseado em (APPERLY, 2001)).

Segmento	Serviços
Produção	Divulgar os componentes projetados informalmente ou não documentados
	Utilizar especificações de componentes com ponto de início para projeto
	Divulgar componentes com especificações
	Notificar consumidores de novos componentes ou problemas
Gerência	**Gerenciar a biblioteca do repositório**
	Gerenciar os usuários do repositório
	Gerenciar o catálogo
	Assegurar a qualidade dos componentes
	Gerenciar os componentes
	Tornar os componentes disponíveis
	Gerenciar as versões dos componentes
Consumo	**Buscar componentes que necessita**
	Reportar falhas identificadas
	Especificar componentes para desenvolvimento
	Utilizar e reutilizar especificações de componentes
	Utilizar e reutilizar componentes
	Dispor componentes
	Registrar interesse em componentes
	Receber novas notificações de componentes
	Revisar novos componentes
	Atualizar especificações de componentes e componentes

Dessa forma, as atividades definidas neste trabalho e executadas pela Equipe de Gerência de *Assets* e os artefatos gerados durante a execução deste processo estão apresentados no diagrama principal do Processo de Gerência de *Assets* (ver **FIG. 3.3**). Agrupamos as atividades do processo nos serviços de produção, gerência e consumo.

Iremos agora especificar cada uma dessas atividades definindo seus passos, papéis e artefatos de saída.

Na elaboração do processo proposto foram utilizados os artefatos dos processos de Engenharia de Domínio e Desenvolvimento do Controlador Tático gerados por Gurgel (2004) e Moore (2004).

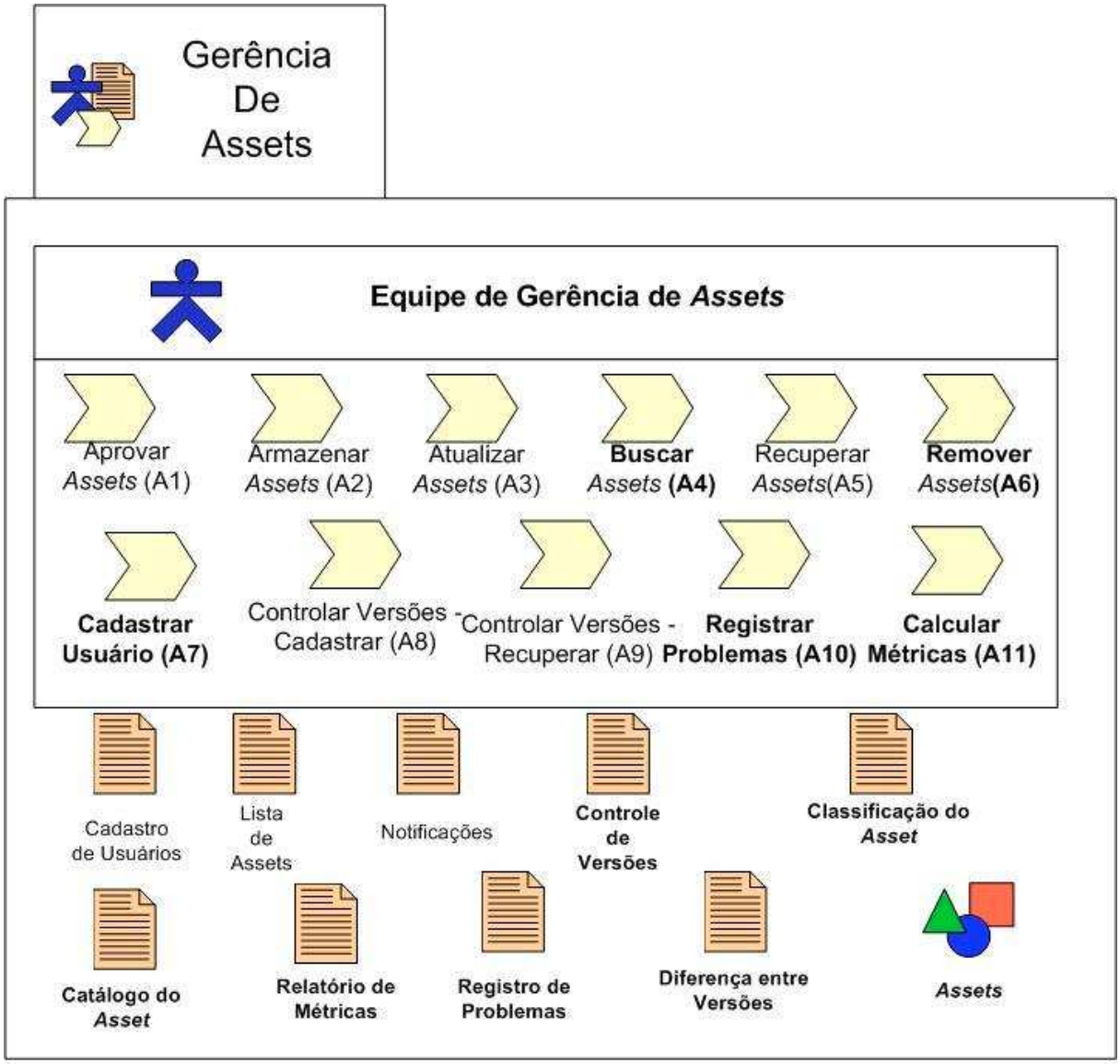


FIG. 3.3: Atividades executadas e artefatos gerados no Processo de Gerência de *Assets.*

3.7.1 PRODUÇÃO

As atividades que constituem o serviço de produção são compra/aquisição ou produção de *assets*. As atividades executadas no processo proposto estão descritas a seguir:

3.7.1.1 APROVAR *ASSETS* (A1)

No processo proposto o propósito dessa atividade é certificar os *assets* a serem reutilizados. A certificação do *asset* é uma fase muito importante no processo de desenvolvimento baseado em reuso. A adoção de um processo de certificação tem o intuito de assegurar a confiança nos *assets* que são utilizados em um sistema, como também no sistema como um todo. Uma vez que o *asset* foi desenvolvido ou adquirido a partir de um fornecedor externo, é necessário que ele seja entendido e qualificado para garantir que realiza as funções necessárias e possui as características de qualidade (por exemplo, desempenho, confiabilidade e usabilidade) (PRESSMAN, 2000). O processo responsável por assegurar essa certificação é o Processo de Garantia da Qualidade (MOORE, 2004). A **FIG. 3.4** apresenta o diagrama de atividades Aprovar *Assets*;

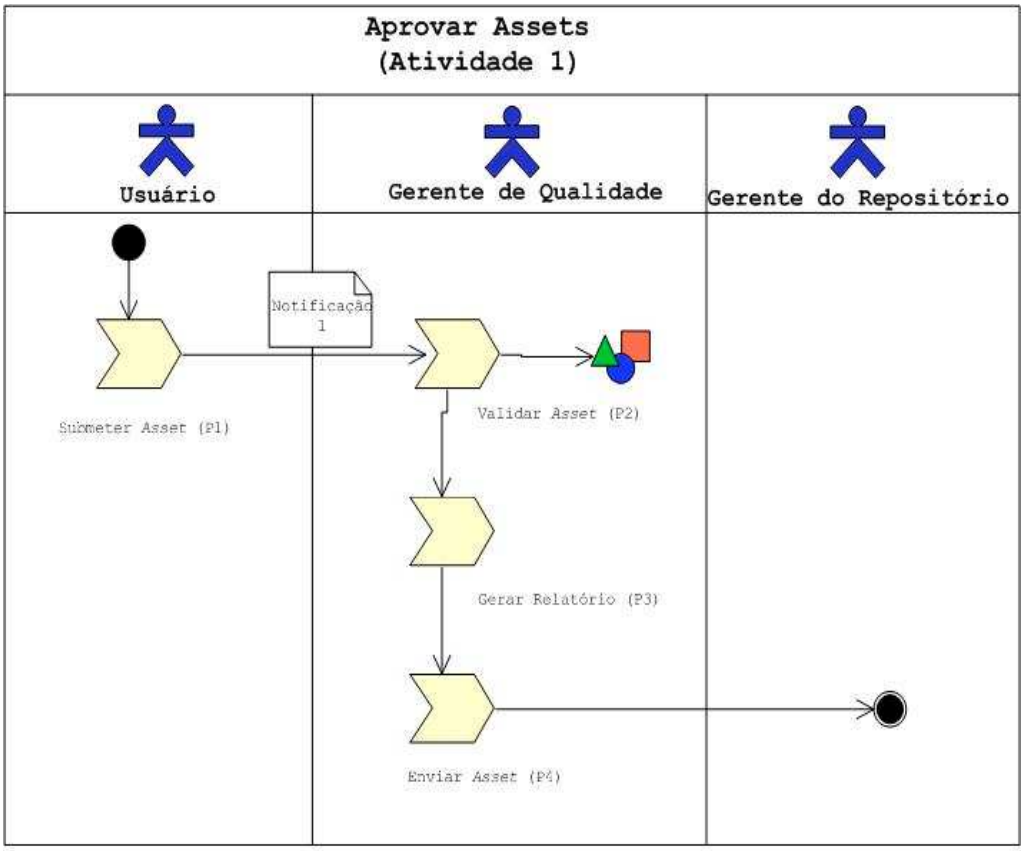


FIG. 3.4: Diagrama de atividades – Aprovar *Assets*

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade consiste na aprovação dos tipos de *assets* e da sua documentação para que eles possam ser posteriormente classificados e armazenados no repositório da biblioteca de *assets*. Para isso, ele deverá ser submetido a um processo de certificação, que consiste na execução de testes, com a finalidade de garantir a sua qualidade e confiança. A sua documentação também deverá propiciar o completo entendimento aos reutilizadores de forma concisa e clara de como reutilizar o *asset*.

RELACIONAMENTOS:	
papeis	Usuário (Engenheiro de Domínio, Desenvolvedores, Montadores) e Gerente de Qualidade
entrada	*Asset* Candidato
saída	*Asset*

TAREFAS:
- **Submeter *Asset* (P1)** O Engenheiro de Domínio ou Desenvolvedores ou Montadores, envia(m) uma notificação para o Gerente de Qualidade informando que há um *asset* candidato para ser certificado pelo Processo de Qualidade. - **Validar *Asset* (P2)** O Gerente de Qualidade valida via *checklist* características de reusabilidade, compatibilidade, padronização da documentação, rastreabilidade, corretude, facilidade no entendimento das funcionalidades e objetivos do *asset*. - **Gerar Relatórios (P3)** A partir da qualidade aprovada, o Gerente de Qualidade gera os relatório de verificação e de revisão. - **Enviar *Asset* (P4)** O Gerente de Qualidade envia o *asset* aprovado para o Gerente do Repositório.

TAB. 3.2: Checklist

Garantia da Qualdade de Software
Checklist : Necessidades do Cliente e Glossário — Versão: 1.5
VENC001/03 — Data da versão: 01/03/2004

Checklist: Necessidades do Cliente e Glossário

Artefato Verificado: CTNC001/03

A coluna P (Previsão) ccontém a resposta correta desejada e na coluna C (Confirmação) o revidor deve preencher com a resposta adequada: S (Sim) / N (Não) / NA (Não se aplica). Caso a resposta da coluna C diferencie da coluna P, exceto quando não se aplica (NA), a coluna I (Identificação) deve ser preenchida com número seqüencial para identificação do erro detectado. Todos os erros detectados identificados e comentados na parte de Não-Conformidades.

4. Aspecto: Rastreabilidade
Este aspecto deve analisar a rastreabilidade das funcionalidades e características desejadas.

		P	C	I
4.1	Cada necessidade é exclusivamente e corretamente identificada?	s		
4.2	Cada funcionalidade pode ser rastreada nos requisitos de *software* de mais alto nível, por exemplo, requisitos de sistema, casos de uso?	s		

Total de itens do aspecto verificados:
Total de não-conformidades:

5. Aspecto: Corretude
Este aspecto deve analisar a corretude das funcionalidades e características desejadas.

		P	C	I
5.1	Cada necessidade é verificável por teste de qualificação, inspeção ou revisão?	s		
5.2	Cada característica desejada é verificável por teste de qualificação, inspeção ou revisão?	s		
5.3	As funcionalidades evitam conflitos entre si e com todas as características desejadas?	s		
5.4	As características desejadas evitam conflitos entre si?	s		
5.5	As necessidades estão escritas em uma linguagem simples e concisa, sem ambigüidade, possibilitando o completo entendimento?	s		
5.6	Cada necessidade está dentro do escopo do projeto?	s		
5.7	Cada necessidade está livre de erros de conteúdo e gramaticais?	s		
5.8	Toda informação necessária para as funcionalidades e características desejadas estão devidamente especificadas? Em caso negativo, identificar no verso?	s		
5.9	Todas as necessidades podem ser implementadas dentro das restrições conhecidas?	s		
5.10	Todas as funcionalidades são de fato funcionalidades e não soluções de projeto ou de implementação?	s		
5.11	As mensagens de erros especificadas são únicas e significativas?	s		

Total de itens do aspecto verificados:
Total de não-conformidades:

Restrito — Instituto Militar de Engenharia — Página: 3/4

TAB. 3.3: Relatório da Revisão

Relatório da Revisão

Id. Projeto: ____ **Técnicas de Leitura:** ____
Data da revisão: ____ **Início (hr):** ____ **Término (hr):** ____
Revisores: **Duração da revisão:** ____ **hr**

Papel	**Nome**	**Rubrica**	**Nível de Conhecimento** (0-baixo/1-médio/2-alto)
Líder			
Expositor			
Secretário			
Autor(es)			
Revisor(es)			
		Média do conhencimento:	

Nome e tipo dos documentos revisados: Doc1: ____ Doc2: ____
Total de discrepâncias: ____
Resultado da Revisão

☐ **Aceito** ☐ Sem restrições ☐ Com restrições Data da próxima revisão: ___/___/___	☐ **Não Aceito** Data da próxima revisão: ___/___/___ ☐ Refazer para nova revisão ☐ Revisão não completada

Problemas

Listar os problemas detectados que não são não-conformidades (por exemplo, o artefato a ser verificado não foi produzido) e as devidas ações decorrentes para resolver o problema (o Gerente de Reúso justificou e autorizou por escrito a dispensa da elaboração do artefato que se encontra em

Problemas	**Ações Decorrentes**

O Questionário de Sugestões para Melhoria da Garantia da Qualidade deve ser preenchido e entregue junto a este Relatório. Sua CONTRIBUIÇÃO é fundamental para o contínuo progresso da melhoria da NOSSA QUALIDADE!

3.7.1.2 ARMAZENAR *ASSETS* (A2)

No processo proposto esta atividade define os passos e cuidados que devem ser tomados durante a publicação de um *asset* no repositório, de forma a permitir que usuários venham a consultá-lo, posteriormente, sem maiores problemas. Os tipos de *assets* que podem ser reutilizados são: documento de requisito, especificação de arquitetura ou qualquer outro artefato gerado na fase de desenvolvimento como os listado na **TAB. 3.2**. É dever do ator, que desenvolveu o *asset*, ter o cuidado de documentá-lo, a fim de viabilizar seu posterior consumo. A atividade de armazenar engloba a classificação e a catalogação do *asset* no repositório e a sua inclusão no Controle de Versões. Após inserção no repositório uma notificação é enviada ao Gerente de Confuguração. A **FIG. 3.5** apresenta o diagrama de atividade Armazenar *Assets*.

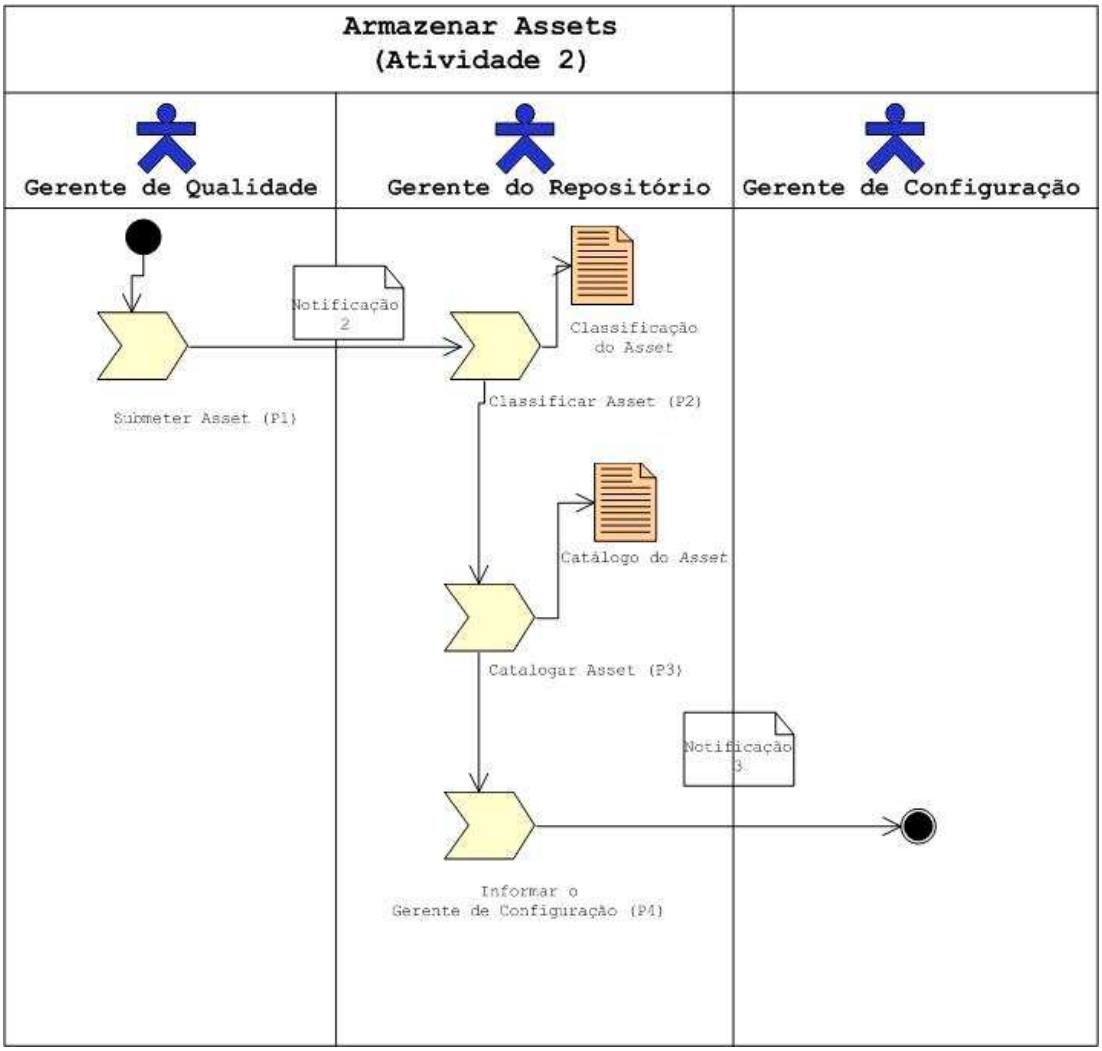


FIG. 3.5: Diagrama de atividades – Armazenar *Assets*

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é classificar, catalogar e armazenar os *assets* no repositório para futura busca e recuperação pelos usuários do sistema. A classificação baseia-se no padrão proposto por PRIETO DIAZ (1991), a documentação segue SAMETINGER (1996) e a catalogação o padrão proposto pela RAS (2005).

RELACIONAMENTOS:	
papeis	Gerente de Qualidade, Gerente do Repositório e Gerente de Configuração
entrada	*Asset*
saída	Classificação, Catálogo do *Asset* e Notificação

TAREFAS:
- **Submeter *Asset* (P1)** O Gerente de Qualidade, envia uma notificação para o Gerente do Repositório informando que há um *asset* para ser inserido no repositório. - **Classificar *Asset* (P2)** O Gerente do Repositório classifica o *asset*. A classificação segue o esquema proposto por Prieto Diaz (1991), chamada classificação em facetas. A designação de cada faceta, bem como as informações pertinentes a elas, está descriminada na documentação que acompanha o *asset*. O Gerente do Repositório de mãos dessa documentação extrai essas informações e gera o esquema de classificação do *asset* (ver *template*). Essa classificação é necessária para que posteriormente os usuários, possam buscar e selecionar os *assets* do repositório com mais eficiência. - **Catalogar *Asset* (P3)** O Gerente do Repositório cria uma estrutura de armazenamento, organiza e armazena as informações contidas na documentação do *asset*. Essa catalogação baseia-se na RAS 2005 (OMG), que define uma forma padrão de armazenamento das informações (ver *template*). A documentação descreve a funcionalidade do *asset*, as questões relativas ao ambiente, como usar e outros aspectos que auxiliem no seu entendimento. - **Informar o Gerente de Configuração (P4)** Após classificar e catalogar o *asset*, o Gerente do Repositório notifica o Gerente de Configuração sobre a inclusão do *asset* no repositório para seu cadastro no Controle de Versões.

TAB. 3.4: Tipos de *assets* produzidos pelo Processo de Desenvolvimento de *Software* baseado em Reuso na MD (MOORE, 2004).

Assets	**Artefatos**	**Atividade**	**Processo**
Domínio	▪ Modelo de Contexto ▪ Modelo Conceitual ▪ Modelo de Casos de Uso do Domínio e Cenários ▪ Modelo de Características ▪ Ontologias ▪ Taxonomias ▪ Linguagens de Domínio	Análise de Domínio	Engenharia de Domínio
Arquitetura de Domínio	▪ Modelo Tipo de Negócio ▪ Interfaces da Camada de Negócio e Tipos de Dados	Projeto de Domínio	Engenharia de Domínio

	▪ Interfaces da Camada de Aplicação ▪ Arquitetura de Domínio		
Componente de Domínio	▪ Especificação de Componentes de Domínio ▪ Interfaces de Componentes de Domínio	Projeto de Domínio	Engenharia de Domínio
Componentes	▪ Código de Componente ▪ Teste de Componentes	Construção do *Software*	Processo de Desenvolvimento

TEMPLATE CLASSIFICAÇÃO DO *ASSETS*

Estrutura (facetas) que contém as palavras-chaves para a realização de busca do *asset*;

Lista um conjunto de descritores para classificação dos *assets*, bem como uma descrição do contexto para o qual o *asset* é relevante.

Domínio:	*Especifica a área que é atendida pelos serviços oferecidos pelo asset*
Aplicação:	*Especifica qual a finalidade dos serviços oferecidos dentro do domínio do asset*
Tipo de Asset:	*Tipo de asset produzido pelo processo de desenvolvimento de software baseado em reuso no Ministério da Defesa*
Atividade:	*Em que atividade (fase do desenvolvimento) o asset foi desenvolvido*
Tipo de Componente:	*Especifica o tipo de componente de software, tais como: sub-rotina, classe, DLL e outros*
Comando:	*Comando subordinado ao Ministério da Defesa - Aeronáutica/Exército/Marinha*
Tipo de Sistema:	*Sistema Científico (SC) ou Sistema Administrativo (SA)*
Descrição:	*Conjunto de palavras que descrevam a finalidade do asset*

TEMPLATE CATÁLOGO DO *ASSET*

Estrutura onde são armazenadas as informações do *asset*. Baseado na RAS 2005 e na documentação de SAMETINGER (1996).

Profile:

Pasta onde são armazenadas as informações referentes ao perfil do asset.

- **Nome do *asset*:** *nome, dando uma possível sugestão sobre a funcionalidade do asset;*
- **Cod Ident:** *código de identificação associado ao asset;*

- **Autor:** *pessoa ou grupo de pessoas, que desenvolveram o asset;*
- **Descrição:** *declaração inicial concisa e clara sobre a funcionalidade do asset para seleção inicial, e inclusive com descrição geral de todas as operações externamente visíveis, se for um componente; e*
- **Versão:** *número da versão do asset.*
- **Artefatos:** *relação de artefatos contidos no asset acompanhado do seu número de identificação e de uma breve descrição;*

Solução:

Pasta onde são armazenados os artefatos que fazem parte do asset.

Utilização:

Pasta onde são armazenadas as regras para instalação, personalização, e utilização do asset;

- **Características de Reutilização:**
  - **Usabilidade:** *descrição de como o asset pode ser utilizado corretamente;*
  - **Instalação** *descrição de como adaptar o asset em uma nova aplicação;*
  - **Adaptação:** *descrição de como e para que necessidades específicas pode ser adaptado o asset;*
  - **Implementação:** *descrição de como o asset está implementado; e*
  - **Suporte:** *onde adquirir ajuda, se necessário for, por exemplo, ao adaptar o componente ou o que fazer se um problema ocorrer.*
- **Características Administrativas:**
  - **Histórico:** *histórico de versões, datas de liberações, desenvolvedores responsáveis, principais diferenças entre as versões; e*
  - **Restrições comerciais e legais:** *restrições comerciais ou legais no uso do componente, por exemplo, aquisição, licença especial ou permissão requerida.*
- **Características de Avaliação:**
  - **Restrições técnicas:** *restrições técnicas no uso do asset, por exemplo, capacidades, linguagem de programação, dependências de sistema operacional;*

- **Qualidade:** *indicar a situação do asset, quanto à qualidade, revisões, verificações e casos de testes aplicados, inclusive os resultados obtidos;*
- **Performance:** *recursos do sistema necessários para usar o asset, por exemplo, memória, processador, canais de comunicação;*
- **Alternativas:** *relação de assets semelhantes que poderiam ser usados no lugar deste;*
- **Problemas:** *relato de problemas conhecidos e custo requerido;*
- **Interdependências**: *o asset pode ser usado stand-alone ou deve ser usado junto a outros assets; e*
- **Custo recomendado:** *custo conhecido para melhorar performance, tornar o asset mais robusto e / ou estender o escopo de reuso.*

▪ **Características Complementares:**

- **Indexação:** *provê um índice para assets complexos que requerem documentação extensa;*
- **Referências:** *referências para literatura ou outra documentação, e por exemplo, documentação de sistemas.*

***Assets* Relacionados:**

Pasta onde é armazenada a relação de assets que possuam alguma ligação com o asset documentado.

3.7.1.3 ATUALIZAR *ASSETS* (A3)

No processo proposto a produção de um *asset* não envolve apenas a primeira publicação do mesmo no repositório. Um software evolui após ser colocado em produção, devido à descoberta de falhas, alterações nos requisitos (especificação), remodelagem etc. e neste caso armazenamos uma nova versão do *asset* no repositório. Manter um *asset* atualizado, portanto, é uma tarefa de extrema importância e está relacionada ao controle de suas versões. Sendo assim a atividade atualizar *assets* deve armazenar uma nova versão do *asset* no repositório e atualizar o controle de versões com as informações da nova versão. Após atualização do *asset*

no repositório, uma notificação é enviada aos usuários do sistema. A **FIG. 3.6** apresenta o diagrama de atividades Atualizar *Assets*;

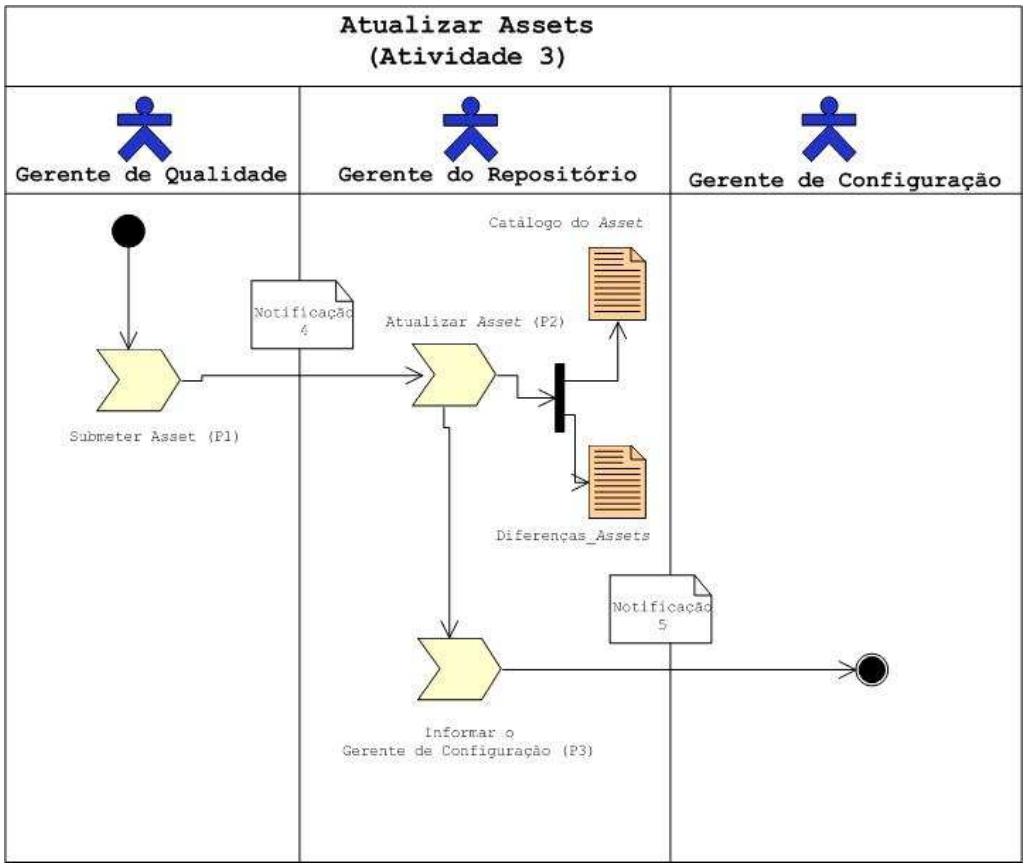


FIG. 3.6: Diagrama de atividades – Atualizar *Assets*

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é catalogar e armazenar as atualizações dos *assets* no repositório para futura busca e recuperação pelos usuários do sistema.

RELACIONAMENTOS:	
papeis	Gerente de Qualidade, Gerente do Repositório e Gerente de Configuração
entrada	*Asset*
saída	Diferenças_*Assets*

TAREFAS:
- **Submeter *Asset* (P1)** O Gerente de Qualidade, envia uma notificação para o Gerente do Repositório informando que há uma nova versão do *asset* para ser armazenado no repositório. - **Atualizar *Asset* (P2)** A partir da estrutura de armazenamento já criada, o Gerente do Repositório armazena as informações contidas na documentação do *asset* bem como as informações contidas no documento q registram as diferenças entre a versão atual e a anterior. - **Informar o Gerente de Configuração (P3)** Após armazenar o *asset*, o Gerente do Repositório notifica o Gerente de Configuração sobre a inclusão da nova versão do *asset.*

TEMPLATE DIFERENÇA_*ASSETS:*

Diferença_*Assets*: relação das diferenças entre as versões de um mesmo *asset*.

Nome do *Asset*	*Nome do asset*
Cod. Ident.	*Código de identificação associado ao asset*
Versão Anterior:	*Número da versão/versões que deram origem ao asset.*
Diferenças Profile:	*Descrição das diferenças nesse item da documentação*
Diferenças Solução:	*Descrição das diferenças nesse item da documentação*
Diferenças Utilização: **Características de Reutilização**	*Descrição das diferenças nesse item da documentação*
Diferenças Utilização: **Características Administrativas**	*Descrição das diferenças nesse item da documentação*
Diferenças Utilização: **Características de Avaliação**	*Descrição das diferenças nesse item da documentação*
Diferenças Utilização: **Características de Complementares**	*Descrição das diferenças nesse item da documentação*

3.7.1.4 NOTIFICAR CONSUMIDORES

No processo proposta definimos que essa atividade está embutida nas atividades de Armazenamento (A1), Atualização (A2), bem como na maioria das atividades do Processo de Gerência de Assets.

O consumo de *assets*, para se tornar mais eficiente, deve contar com um comportamento pró-ativo por parte do repositório. Assim, o usuário não só deve consultar um *asset*, na busca de determinada função (para reutilizá-lo dentro de um projeto), como também deve ser notificado pelo sistema, quando da inserção de algum novo *asset* (armazenamento), ou do lançamento de uma nova versão (atualização). Neste processo isto é feito através de envio de correio eletrônico a todos os usuários, quando da publicação de um *asset* totalmente novo ou de uma atualização.

TEMPLATE DAS NOTIFICAÇÕES

Contém a relação de todas as notificações geradas pelas atividades do processo.

NOTIFICAÇÃO 1:	*Tarefa APROVAR ASSETS*
DE:	*Usuário*
PARA:	*Gerente de Qualidade*
ASSUNTO:	*Certificar asset candidato*
NOME DO ASSET:	*Nome do asset a ser aprovado*
VERSÃO:	*Versão do asset candidato*
OBS:	-

NOTIFICAÇÃO 2:	*Tarefa ARMAZENAR ASSETS*
DE:	*Gerente de Qualidade*
PARA:	*Gerente do Repositório*
ASSUNTO:	*Incluir novo asset no repositório*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
OBS:	-

NOTIFICAÇÃO 3:	*Tarefa ARMAZENAR ASSETS*
DE:	*Gerente do Repositório*

PARA:	*Gerente de Configuração*
ASSUNTO:	*Cadastrar asset no Controle de Versões*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
OBS:	*Essa notificação é enviada para a tarefa CONTROLAR VERSÕES - CADASTRAR ASSETS (Notificação 10)*

NOTIFICAÇÃO 4:	*Tarefa ATUALIZAR ASSETS*
DE:	*Gerente de Qualidade*
PARA:	*Gerente do Repositório*
ASSUNTO:	*Incluir nova versão do asset no repositório*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
OBS:	-

NOTIFICAÇÃO 5:	*Tarefa ATUALIZAR ASSETS*
DE:	*Gerente do Repositório*
PARA:	*Gerente de Configuração*
ASSUNTO:	*Cadastrar nova versão do assset no Controle de Versões*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
OBS:	*Essa notificação é enviada para a tarefa CONTROLAR VERSÕES – CADASTRAR ASSETS (Notificação 10)*

NOTIFICAÇÃO 6:	*Tarefa RECUPERAR ASSETS*
DE:	*Gerente do Repositório*
PARA:	*Gerente de Configuração*
ASSUNTO:	*Atualizar Controle de Versões com as informações do asset que foi retirado do repositório para ser reutilizado*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
NOME DO USUÁRIO:	*Nome do usuário que reutilizará o asset*
ORGÃO:	*Nome do órgão que ele trabalha*
SISTEMA:	*Nome do sistema que o asset será reutilizado*
OBS:	*Essa notificação é enviada para a tarefa CONTROLAR VERSÕES – RECUPERAR ASSETS (Notificação 13)*

NOTIFICAÇÃO 7:	*Tarefa REMOVER ASSETS*
DE:	*Gerente de Configuração*

PARA:	*Gerente do Repositório*
ASSUNTO:	*Remover asset do repositório*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
OBS:	-

NOTIFICAÇÃO 8:	*Tarefa CADASTRAR USUÁRIO*
DE:	*Gerente de Configuração*
PARA:	*Usuário*
ASSUNTO:	*Pedido de cadastro recusado*
NOME DO USUÁRIO:	*Nome do usuário*
EMAIL:	*Email do usuário*
OBS:	-

NOTIFICAÇÃO 9:	*Tarefa CADASTRAR USUÁRIO*
DE:	*Gerente de Configuração*
PARA:	*Usuário*
ASSUNTO:	*Pedido de cadastro aceito*
NOME DO USUÁRIO:	*Nome do usuário*
EMAIL:	*Email do usuário*
OBS:	-

NOTIFICAÇÃO 10:	*Tarefa CONTROLAR VERSÕES – CADASTRAR ASSETS*
DE:	*Gerente do Repositório*
PARA:	*Gerente de Configuração*
ASSUNTO:	*Cadastrar novo asset ou nova versão do asset no Controle de Versões*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
OBS:	*Essa notificação foi enviada pela tarefa ARMAZENAR ASSETS (Notificação 3) ou pela tarefa ATUALIZAR ASSETS (Notificação 4)*

NOTIFICAÇÃO 11:	*Tarefa CONTROLAR VERSÕES - CADASTRAR ASSETS*
DE:	*Gerente de Configuração*
PARA:	*Usuários que trabalhem com esse tipo de asset*
ASSUNTO:	*Inclusão de asset no repositório*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
EMAIL:	*Email dos usuários*
OBS:	-

NOTIFICAÇÃO 12:	*Tarefa CONTROLAR VERSÕES - CADASTRAR ASSETS*
DE:	*Gerente de Configuração*
PARA:	*Usuários que trabalhem com esse tipo de asset*
ASSUNTO:	*Inclusão de nova versão do asset no repositório*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
EMAIL:	*Email dos usuários*
OBS:	-

NOTIFICAÇÃO 13:	*Tarefa CONTROLAR VERSÕES – RECUPERAR ASSETS*
DE:	*Gerente do Repositório*
PARA:	*Gerente de Configuração*
ASSUNTO:	*Atualizar Controle de Versões com as informações do asset que foi retirado do repositório para ser reutilizado*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
NOME DO USUÁRIO:	*Nome do usuário que reutilizará o asset*
ORGÃO:	*Nome do órgão que ele trabalha*
SISTEMA:	*Nome do sistema que o asset será reutilizado*
OBS:	*Essa notificação foi enviada pela tarefa RECUPERAR ASSETS (Notificação 6)*

NOTIFICAÇÃO 14:	*Tarefa REGISTRAR PROBLEMAS*
DE:	*Usuário*
PARA:	*Gerente de Configuração*
ASSUNTO:	*Registrar problema detectado pelo usuário*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
PROBLEMA:	*Relato de problema detectado pelo usuário e a solução caso tenha encontrado*
OBS:	-

NOTIFICAÇÃO 15:	*Tarefa REGISTRAR PROBLEMAS*
DE:	*Gerente de Configuração*
PARA:	*Usuários que utilizam o asset que foi detectado o problema*
ASSUNTO:	*Notificação de Problema*
NOME DO ASSET:	*Nome do asset*
VERSÃO:	*Versão do asset*
PROBLEMA:	*Relato de problema detectado pelo usuário e a solução caso ele tenha encontrado*
OBS:	-

3.7.2 GERÊNCIA

A atividade de gerência é realizada paralela e independentemente das atividades de produção e consumo. Engloba uma série de tarefas de cunho administrativo, como gerenciar repositórios e usuários. Estas tarefas administrativas são normalmente demandadas, durante a produção ou o consumo de *assets*. Desta maneira, quando um membro recém integrado a um projeto necessita, por exemplo, consumir um item do repositório, deve-se criar uma conta e uma senha de acesso para o mesmo, dentro da atividade de controle de usuários.

As atividades de Gerência no processo proposto estão descritas abaixo:

3.7.2.1 REMOVER *ASSET* (A6)

No processo proposto o propósito dessa atividade é retirar os assets que estejam obsoletos (término da sua vida útil), ou que tenham um índice de reuso muito baixo. A decisão da retirada de um asset é tomada a partir do relatório do cálculo de métricas do repositório. A FIG. 3.7 apresenta o diagrama de atividades Remover Assets;

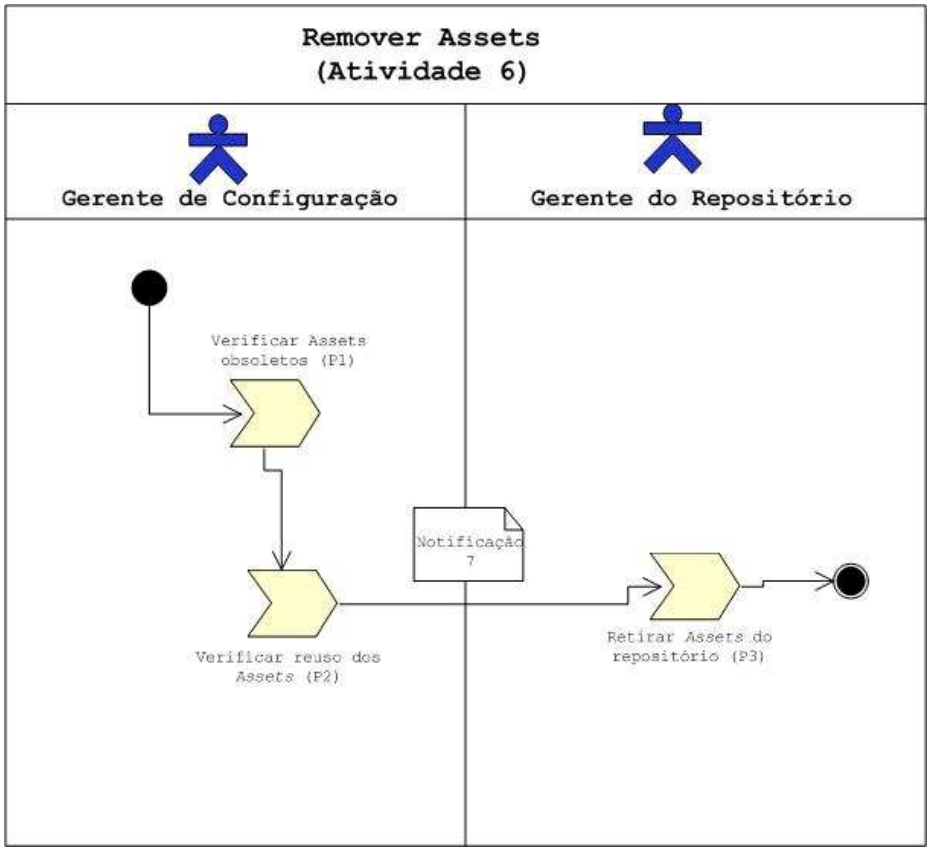


FIG. 3.7: Diagrama de atividades – Remover *Assets*

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é possibilitar que o Gerente do Repositório remova do repositório os *assets* que atendam a critérios de obsolescência definidos para o sistema e verificados pelo Gerente de Configuração.

RELACIONAMENTOS:	
papeis	Gerente do Repositório e Gerente de Configuração
entrada	Relatório de Métricas
saída	-

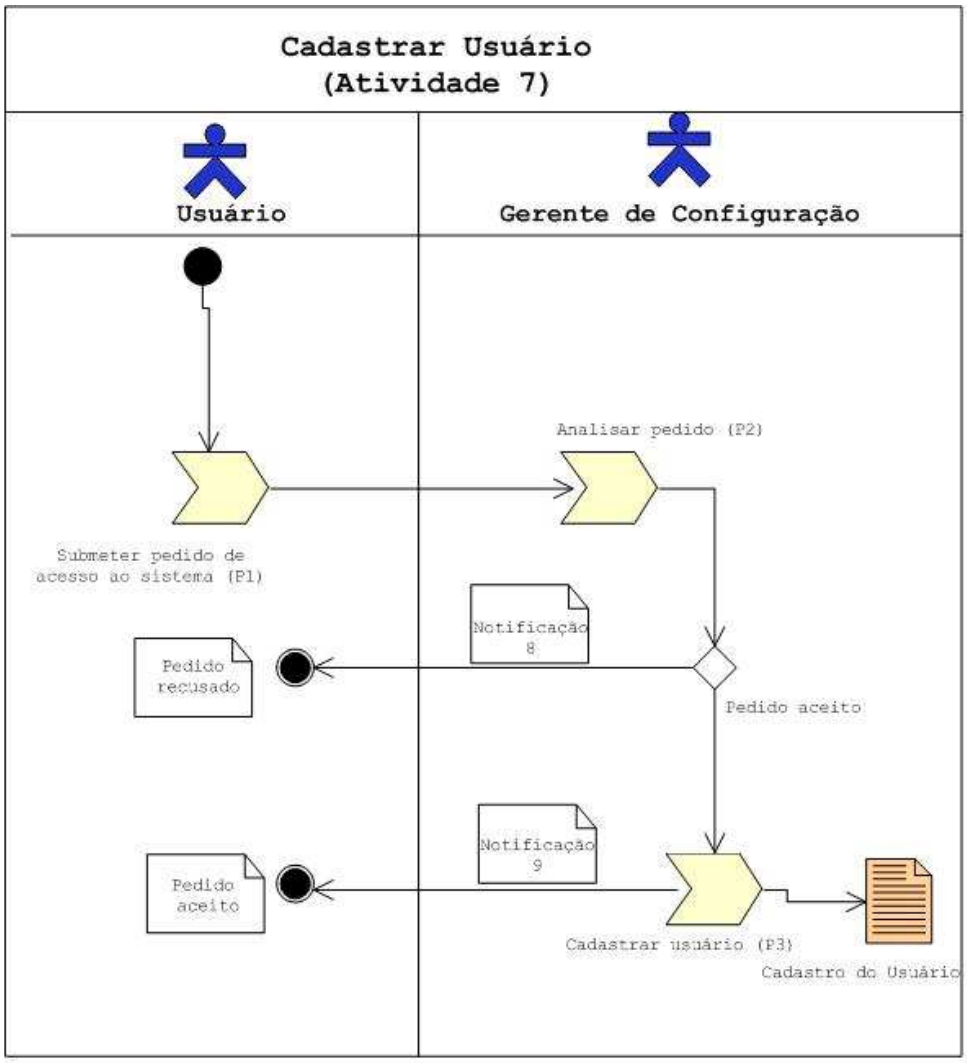
TAREFAS:
- **Verificar *Assets* Obsoletos (P1)** O Gerente de Configuração, através do levantamento das métricas, verifica os *assets* que não são reusados por um longo período de tempo ou que nunca foram reusados. Considerando esses números, ele pode decidir ou não manter certos *assets* no repositório. - **Verificar Reuso dos *Asset* (P2)** O Gerente de Configuração verifica se os *assets* ditos obsoletos estão sendo reutilizados no desenvolvimento de um sistema. Se ele estiver sendo reutilizado, não poderá ser retirado do repositório. Caso contrário, o Gerente de Configuração, deverá notificar o Gerente do Repositório e retirar os seus dados do sistema de Controle de Versões. - **Retirar *Asset* do Repositório (P3)** O Gerente de Configuração informa ao Gerente do Repositório quais são o *assets* que deverão ser excluídos do repositório. Após receber uma notificação, o Gerente do Repositório exclui do repositório todas as informações referentes aos *assets* (*asset* + esquema de classificação + catálogo).

3.7.2.2 CADASTRAR USUÁRIO (A7)

No processo proposto o propósito dessa atividade é prover uma política de controle de usuários. Para prover acesso controlado ao repositório de *assets* o Gerente de Configuração deve criar usuários, atribuir-lhes permissões, se for o caso, e removê-los quando necessário. Nesta proposta a gerência de usuários requer intervenção humana e a permissão atribuída a cada usuário depende do órgão subordinado ao Ministério da Defesa e da atividade que ele exerce. Nesse processo temos as seguintes permissões:

- **Sistema Científico** (SC) – os usuários têm acesso somente a *assets* construídos a partir de sistemas científicos;
- **Sistemas Administrativos** (SA) - os usuários têm acesso somente a *assets* construídos a partir de sistemas administrativos; e
- **Sem Restrição** (SR) os usuários têm acesso aos *assets* de ambos os sistemas.

A **FIG. 3.8** apresenta o diagrama de atividades Cadastrar Usuário:



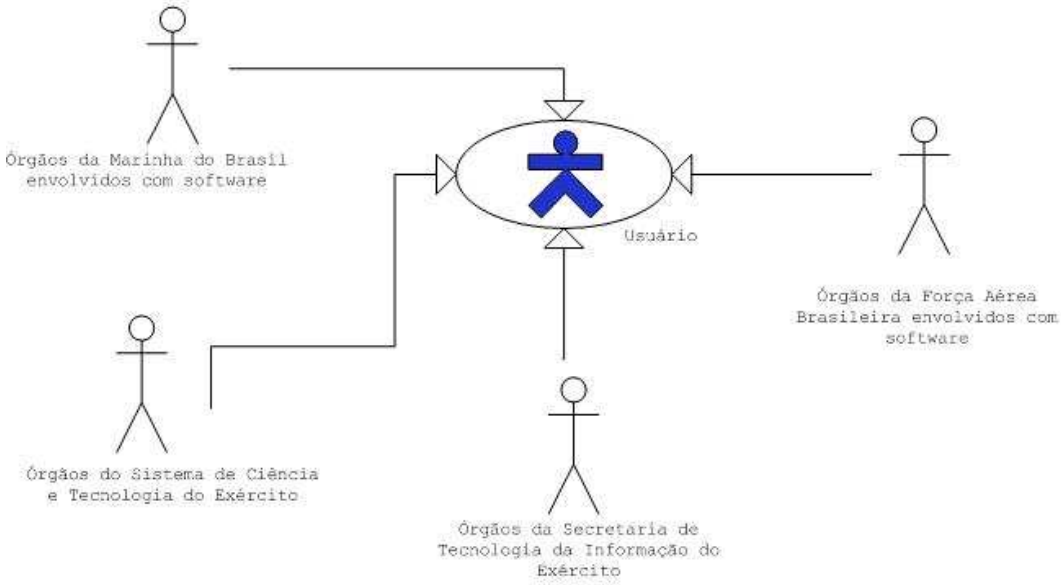


FIG. 3.8: Diagrama de atividades – Cadastrar Usuário

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é cadastrar os usuários e atribuir-lhes permissões para efetuar o controle de acesso ao sistema.

RELACIONAMENTOS:	
papeis	Usuário (Órgãos da Marinha do Brasil envolvidos com software, Órgãos do Sistema de Ciência e Tecnologia do Exército, Órgãos da Secretaria de Tecnologia da Informação do Exército e Órgãos da Força Aérea Brasileira envolvida com software – ver tabela) e Gerente do Configuração
entrada	-
saída	Cadastro do Usuário

TAREFAS:
- **Submeter Pedido de Acesso ao Sistema (P1)** O Usuário preenche seus dados cadastrais e estes são enviados para serem analisados pelo Gerente de Configuração. - **Analisar Pedido (P2)** Assim que o Usuário submete o seu pedido, o Gerente de Configuração faz a análise dos seus dados. Se o Usuário pertencer a um dos órgãos de desenvolvimento de software subordinados ao Ministério da Defesa a sua requisição é aceita. O Gerente de Configuração envia uma notificação para o usuário caso o seu pedido de inclusão não tenha sido aceito. - **Cadastrar Usuário (P3)** Caso o pedido de acesso feito pelo usuário tenha sido aceito, o Gerente de Configuração armazena as suas informações, atribui a sua permissão de acesso aos dados do repositório e notifica o usuário da sua inclusão no sistema. Caso ele trabalhe com desenvolvimento de sistemas científicos a sua permissão será SC e ele só acessará *assets* desse contexto. O mesmo será feito para sistemas administrativos e a sua permissão será SA e ele só acessará *assets* desse contexto. Caso ele trabalhe no desenvolvimento de ambos os sistemas o seu acesso será total (sistemas científicos e técnicos – sem restrição) e a sua permissão será SR (ver *template*).

Os usuários que trabalham nos órgãos de desenvolvimento de software, subordinados ao Ministério da Defesa, podem requisitar acesso ao repositório de dados onde estão

armazenados os *assets* para futuro reuso. As **TAB. 3.5, 3.6 e 3.7** apresentam as organizações participantes do processo de desenvolvimento de software baseado em reuso.

TAB. 3.5: Órgãos da Marinha do Brasil envolvidos com *software*

CASNAV Centro de Apoio a Sistemas Navais	Engenharia de software e pesquisa operacional com maior enfoque ao apoio à decisão.
DGMM Diretoria-Geral do Material da Marinha	Aquisição, aparelhamento e manutenção de meios
DSAM Diretoria de Sistemas de Armas da Marinha	Sistemas de Armas e de Comando e Controle da Marinha
DTM Diretoria de Telecomunicações da Marinha	Redes e segurança da informação.
IpqM Instituto de Pesquisa da Marinha	Eletrônica e computação com maior enfoque aos sistemas operativos.
CASOP Centro de Apoio a Sistemas Operativos	Eletrônica e computação com maior enfoque à manutenção.
DHN Diretoria de Hidrografia e Navegação	Computação gráfica.
DadM Diretoria de Administração da Marinha	Banco de dados, engenharia de sistemas de computação e sistemas de informação administrativos.
EMGEPRON Empresa de Gerenciamento de Projetos Navais	Diversas áreas com maior enfoque ao desenvolvimento de sistemas operativos para os meios Navais.

TAB. 3.6: Órgãos Subordinados ao Departamento de Ciência e Tecnologia - DCT

DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA DO EXÉRCITO	
CTEx Centro Tecnológico do Exército	Apoiar a pesquisa e o desenvolvimento na área de material no Exército.
IME Instituto Militar de Engenharia	Ensino e Pesquisa Científico-Tecnológica e de Capacitação de Recursos Humanos
CAEx Centro de Avaliação do Exército	Encarregado das avaliações operacionais de materiais de emprego militar.
CDS Centro de Desenvolvimento de Sistemas	Sistemas pertinentes às áreas de comunicações estratégica e tática, guerra eletrônica, informática e de informações organizacionais de interesse do Exercito.
CITEx Centro de Telemática do Exército	Sistemas Corporativos.
DSG Diretoria de Serviço Geográfico	Cartas topográficas militares.
CIGE Centro Integrado de Guerra Eletrônica	Responsável pela formação de recursos humanos nos sistemas de Guerra Eletrônica

DF Diretoria de Fabricação	Órgão de Apoio técnico-normativo do Departamento Logístico (D Log) incumbido de superintender as atividades relativas à fabricação, recuperação, adaptação, transformação, modernização e nacionalização do material de emprego militar.

TAB. 3.7: Órgãos da Força Aérea Brasileira envolvidos com *software*

SUB-DEPARTAMENTO DE TECNOLOGIA DA INFORMAÇÃO	
CCA BR Centro de Computação de Aeronáutica de Brasília	Sistemas de apoio à decisão e redes na área Norte e Central.
CCA SJ Centro de Computação de Aeronáutica de São José dos Campos	Sistemas operativos e redes na área de São Paulo e Sul.
CCA RJ Centro de Computação de Aeronáutica do Rio de Janeiro	Sistemas administrativos e redes na área do Rio de Janeiro, Espírito Santo, Minas Gerais e Nordeste.
SIGPES Sistema de Gerências de Pessoal	Sistemas de gerência de pessoal.
SILOMS Sistema de Logística de Material e Suprimentos	Sistema de logística de material e suprimentos.
ITA Instituto Tecnológico da Aeronáutica	Ensino e Pesquisa Científico-Tecnológica e de Capacitação de Recursos Humanos.

TEMPLATE CADASTRO DO USUÁRIO

Nome:	*Nome do usuário*
Login:	*Login do usuário*
Senha:	*Senha do usuário*
Email:	*Email do usuário*
Orgão:	*Orgão subordinado ao Comando que o usuário trabalha*
Comando:	*Comando q o usuário trabalha - Aeronáutica/Exército/Marinha*
Função:	*Função do usuário*
Endereço:	*Endereço de trabalho*
Telefone:	*Telefone do trabalho*
Cidade:	*Cidade do local de trabalho*
Estado:	*Estado do local de trabalho*
Acesso	*SC (Sistema Científico)/ SA (Sistema Administrativo)/ SR (sem restrição - ambos os sistemas)*

3.7.2.3 CONTROLAR VERSÃO (A8 E A9)

No processo proposto o propósito dessa atividade é manter o controle das diferentes versões que são geradas com as alterações realizadas sobre os assets durante o seu ciclo de vida. Essa atividade também é responsável por controlar qual usuário está utilizando determinado asset e a sua referida versão. Armazenar e gerenciar diferentes versões de todos os assets de um projeto envolve uma grande quantidade de informação e por isso há a necessidade de um controle efetivo por parte da equipe de Gerência de Assets. A FIG. 3.9 apresenta o diagrama de atividades Controlar Versões – Cadastrar e a FIG. 3.10 apresenta o diagrama de atividades Controlar Versões - Recuperar;

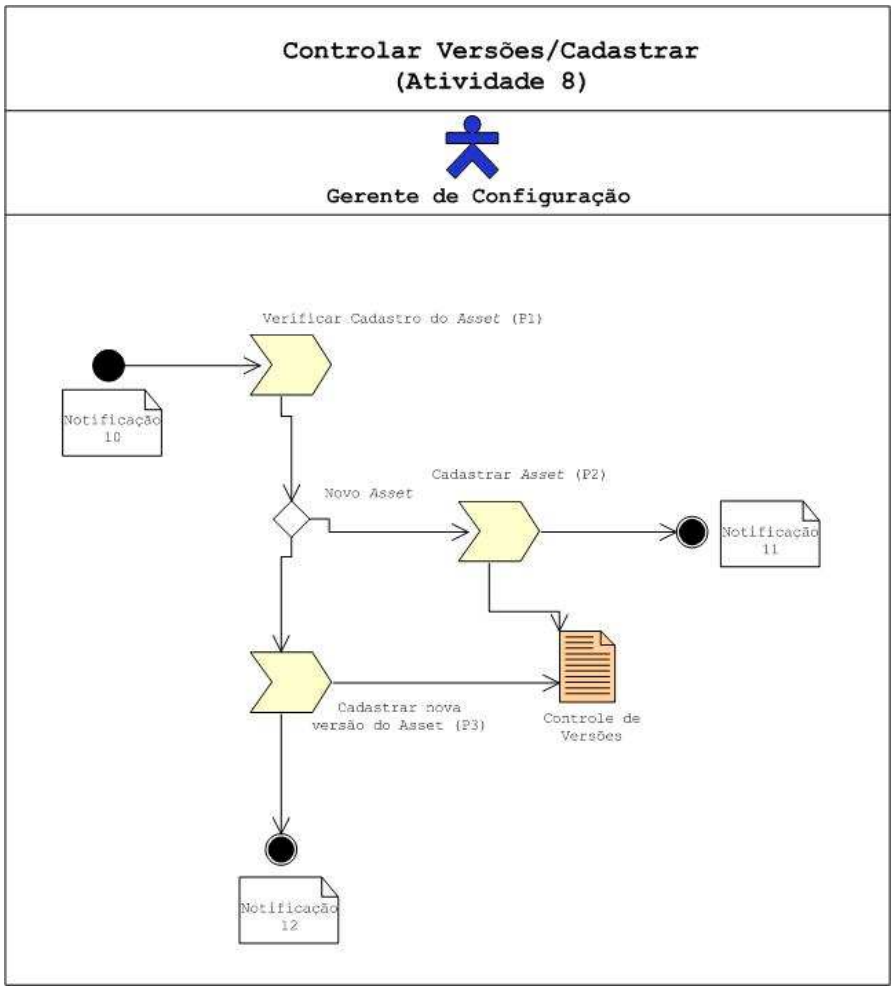


FIG. 3.9: Diagrama de atividades – Controlar Versão - Cadastrar

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é cadastrar um novo *asset* ou uma nova versão no Controle de Versões, para q se possa controlar as diferentes versões e os diferentes usuários q as utilizam. Após o cadastro, o Gerente de Configuração envia uma notificação para os usuários do sistema informando a inclusão de novo *asset*/versão no repositório.

RELACIONAMENTOS:	
papeis	Gerente de Configuração
entrada	Notificação
saída	Controle de Versões e Notificação

TAREFAS:

- **Verificar Cadastro do *Asset* (P1)**

A partir da notificação recebida, o Gerente de Configuração verifica se é um novo asset ou uma nova versão para ser inserido no Controle de Versões.

- **Cadastrar *Asset* (P2)**

O Gerente de Configuração efetua o cadastro do novo *asset* no sistema. Para isso, ele deverá armazenar as informações necessárias do *asset* no Controle de Versões (vide *template*). Após o cadastro, será enviada uma notificação para os usuários (que trabalhem com o tipo de *asset* – SA/SC), sobre o novo *asset* no repositório.

- **Cadastrar Nova Versão do *Asset* (P3)**

O Gerente de Configuração efetua o cadastro de uma nova versão do *asset* no sistema. Caso já exista uma versão, ele deverá apenas atualizar essas informações acrescentando a nova versão do *asset* (ver *template*). Após o cadastro, será enviada uma notificação para os usuários (que trabalhem com uma versão anterior), sobre nova versão do *asset* no repositório.

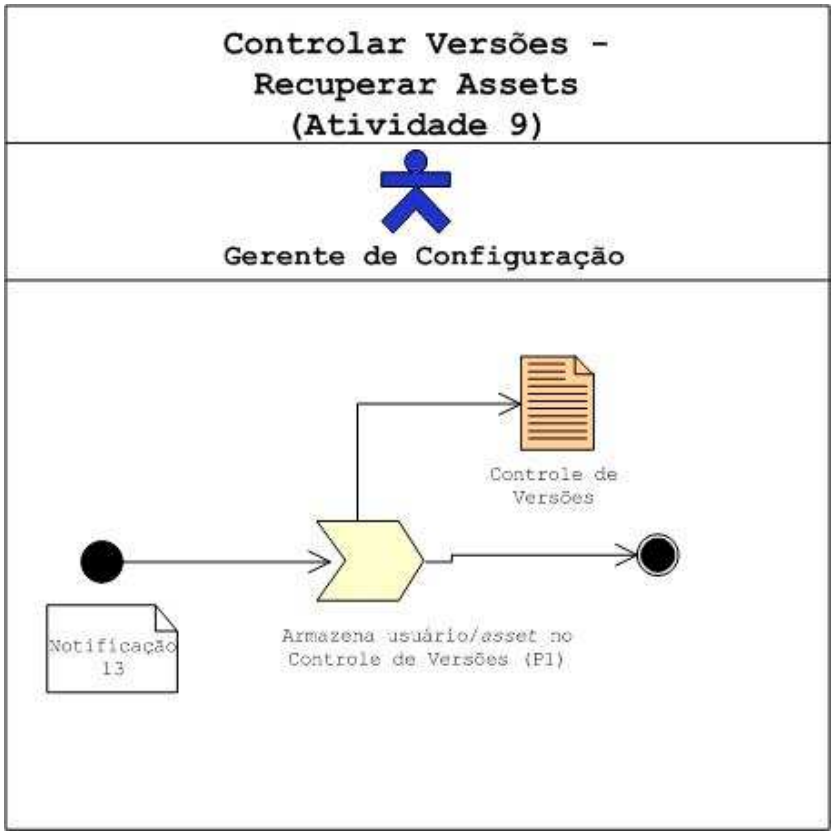


FIG. 3.10: Diagrama de atividades – Controlar Versão – Recuperar

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é armazenar no Controle de Versões os dados do usuário e da versão do *asset* que será reutilizada. Os motivos desse controle são: - evitar q seja retirado do repositório um *asset* q esteja sendo reutilizado; - enviar notificações sobre novas versões do *asset*; e - enviar notificações para os usuários sobre os *assets* que estão sendo reusados.

RELACIONAMENTOS:	
papeis	Gerente de Configuração
entrada	Notificação
saída	Controle de Versões

TAREFAS:
- **Armazena Usuário/*Asset* no Controle de Versões (P1)** O Gerente de Configuração armazena no Controle de Versões os dados do usuário e do *asset* que foi recuperado para ser reutilizado na construção de um novo sistema (ver *template*).

TEMPLATE CONTROLE DE VERSÕES

Contém o controle de todas as versões dos *assets* armazenados no repositório;

Nome do Asset:	*Nome do asset*
Cod Ident:	*Código de identificação associado ao asset*

Versão/Versões Origem:	*Número da versão (versões) que originou (originaram) esta versão*
Versão:	*Número da versão*
Orgão:	*Organização que desenvolveu o asset*

Nome do Usuário:	*Nome do usuário q está utilizando o asset*
Email:	*Email do usuário*
Orgão_Usuario:	*Organização a que ele pertence*
Sistema:	*Nome do sistema que o asset está sendo reutilizado*

Nome do Usuário:	*Nome do usuário q está utilizando o asset*
Email:	*Email do usuário*
Orgão_Usuario:	*Organização a que ele pertence*
Sistema:	*Nome do sistema que o asset está sendo reutilizado*

Versão/Versões Origem:	*Número da versão (versões) que originou (originaram) esta versão*
Versão:	*Número da versão*

Nome do Usuário:	*Nome do usuário q está utilizando o asset*
Orgão_Usuario:	*Organização a que ele pertence*
Sistema:	*Nome do sistema que o asset está sendo reutilizado*

3.7.2.4 REGISTRAR PROBLEMAS (A10)

No processo proposto o propósito dessa atividade é fazer o registro dos problemas detectados nos *assets* que estão sendo reutilizados. Após o registro, uma notificação é enviada aos usuários que também estejam reutilizando o *asset.* A **FIG. 3.11** apresenta o diagrama de atividades Registrar Problemas;

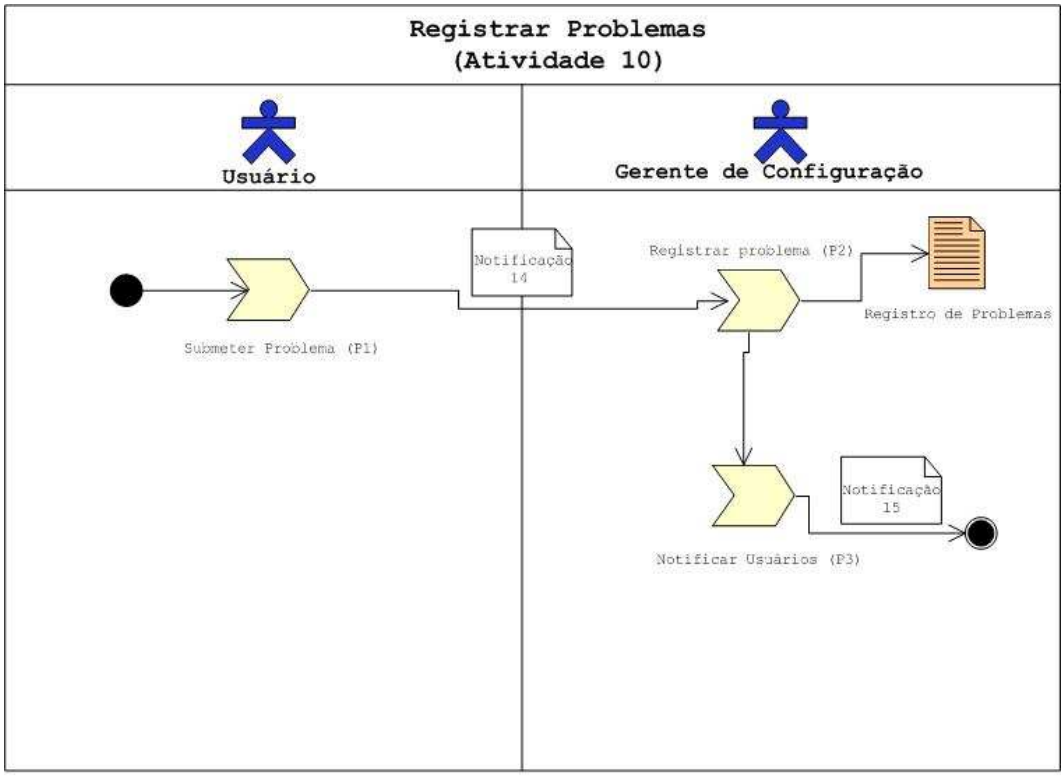


FIG. 3.11: Diagrama de atividades – Registrar Problemas

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é fazer o registro de problemas detectados pelos usuários do sistema, e notificar aqueles que por ventura estiver utilizando o *asset*.

RELACIONAMENTOS:	
papeis	Usuário (Engenheiro de Domínio, Desenvolvedores, Montadores) e Gerente de Configuração
entrada	Notificação
saída	Registro de Problemas e Notificação

TAREFAS:
- **Submeter Notificação (P1)** O Usuário registra do problema detectado e envia uma notificação para o Gerente de Configuração. - **Registrar Problema (P2)** O Gerente de Configuração cadastra a notificação do usuário no sistema (ver *template* Registro de Problemas). - **Notificar Usuários (P3)** Após registrar o defeito, o Gerente de Configuração verifica os usuários que estejam utilizando esse *asset* no Controle de Versões e envia uma notificação informando sobre o problema detectado.

TEMPLATE REGISTRO DE PROBLEMAS

Relação de todos os problemas detectados pelos usuários do sistema.

Asset:	*Nome do asset*
Cod Ident:	*Código de identificação associado ao asset*
Versão:	*Número da versão*

Notificado:	*Nome do usuário que notificou o problema*
Email:	*Email do usuário*
Órgão:	*Nome da organização que o usuário trabalha*
Sistema:	*Nome do sistema que o asset está sendo reutilizado*
Problema:	*Descrição do problema relatado pelo usuário*

3.7.2.5 CALCULAR MÉTRICAS (A11)

No processo proposto o propósito dessa atividade é, trimestralmente, elaborar um relatório contendo as métricas do sistema. A partir do cálculo de métricas do repositório é que podemos avaliar o grau de sucesso do reuso dentro do Contexto do Ministério da Defesa e de seus Comandos Subordinados. A **FIG. 3.12** apresenta o diagrama de atividades Controle de Métricas.

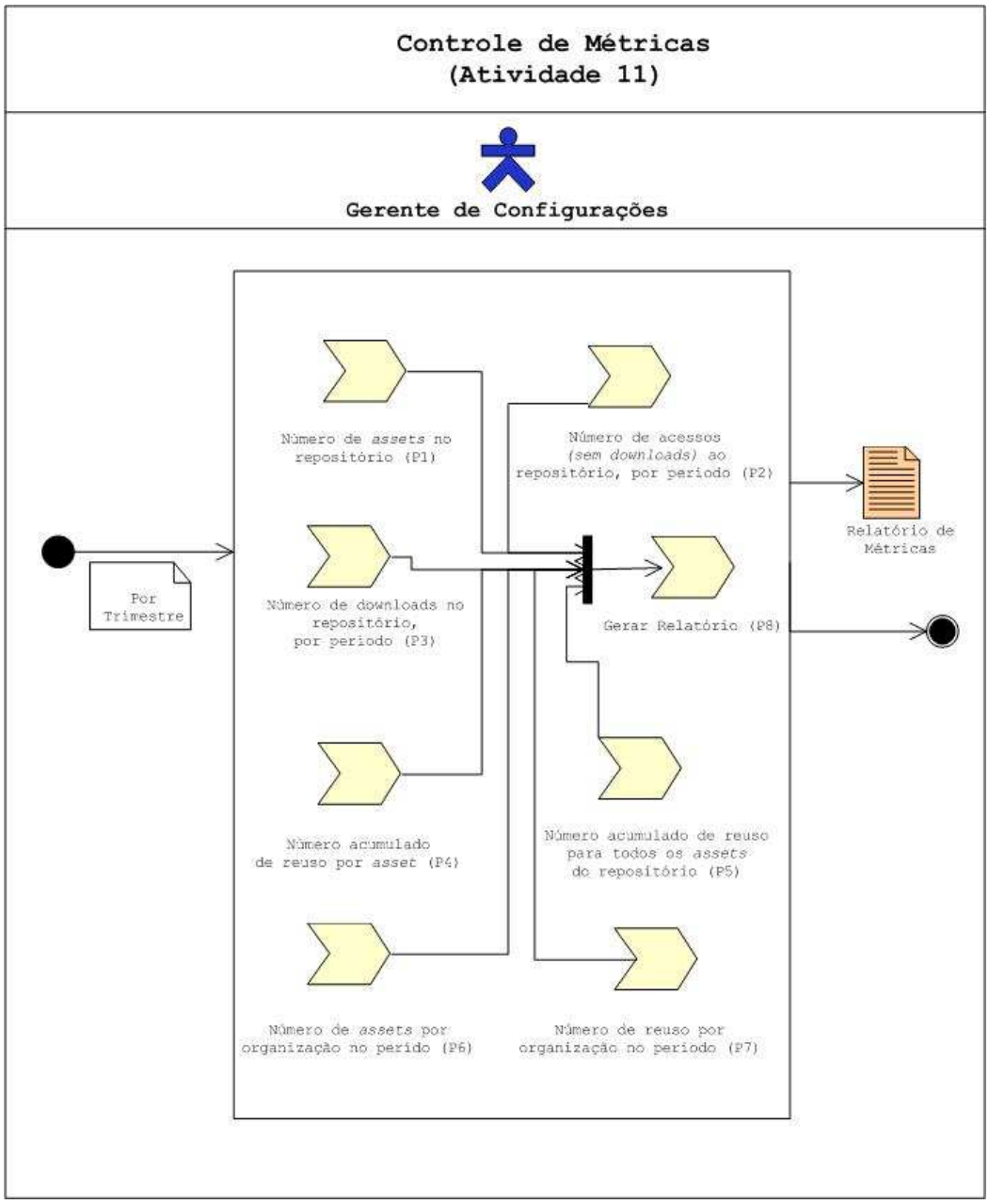


FIG. 3.12: Diagrama de atividades – Controle de Métricas

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é coletar dados e fazer medições (levantamento de métricas) de alguns atributos do repositório e gerar um relatório de métricas. Esse levantamento é feito a cada três meses e a partir dos resultados dessas medidas poderemos verificar o sucesso do programa de reuso dentro da instituição.

RELACIONAMENTOS:	
Papeis	Gerente de Configuração
Entrada	-
Saída	Relatório de Métricas

TAREFAS:

- **Número de *assets* no repositório (P1)**

O Gerente de Configuração verifica o número total de *assets* no repositório. Esse número é usado para prever o potencial de reuso existente. Um número crescente indica que a produção de *assets* para reuso está em pleno andamento, mas um valor muito alto não é isoladamente um indicador de sucesso. Se essa métrica for utilizada como o principal indicador de sucesso do programa de reuso, isso irá incentivar apenas a produção de *assets* e não o seu reuso efetivo.

- **Número de acessos (sem *downloads*) ao repositório, por período (P2)**

O Gerente de Configuração verifica o número de vezes que os usuários do sistema acessaram o repositório para estudar a lista de *assets* candidatos. Um número alto dessa métrica indica que os *assets* não estão sendo encontrados ou que está faltando algum *asset* importante no repositório.

- **Número de downloads do repositório, por período (P3)**

O Gerente de Configuração verifica o número de downloads de *assets* feito pelos usuários do sistema, no repositório. Um número baixo dessa métrica indica que os *assets* não estão sendo encontrados ou que está faltando algum *asset* importante no repositório.

- **Número acumulado de reuso por *asset* (P4)**

Gerente de Configuração verifica o número acumulado de reuso por asset. Um bom indicador dessa métrica significa sucesso do programa de reuso sob o ponto de vista técnico. Um valor baixo desse número pode fazer o gerenciador de reuso decidir manter ou não certos *assets* no repositório, tendo em vista o passado de reuso do *asset*, sua expectativa de reuso no futuro, o custo de manter um *asset* no repositório e a política de se possuir grandes repositórios ou não.

- **Número acumulado de reuso para todos os *assets* do repositório (P5)**

O Gerente de Configuração verifica o número acumulado de reuso para todos os *asset* do repositório. Como na

métrica anterior, um bom indicador dessa métrica significa sucesso do programa de reuso dentro da instituição.

- **Número de *assets* por organização no período (P6)**

O Gerente de Configuração verifica o número de *assets* armazenado no repositório por organização. A importância dessa métrica é verificar qual organização desenvolveu mais *assets* no período e premiá-la como incentivo ao desenvolvimento para reuso.

- **Número de reuso por organização no período (P7)**

O Gerente de Configuração verifica o número de *assets* reutilizados por organização. A importância dessa métrica é verificar qual organização reutilizou mais *assets* no período e premiá-la como incentivo ao desenvolvimento com reuso.

- **Gerar Relatório (P8)**

O Gerente de Configuração elabora um relatório contendo o resultado de todas as métricas (ver *template*).

TEMPLATE RELATÓRIO DE MÉTRICAS

Relatório trimestral contendo as métricas do sistema;

Número de *Assets* no Repositório:	*Quantidade de assets no repositório*
Número de Acessos (sem download) ao Repositório, no trimestre: (dd/mm/aa até dd/mm/aa):	*Quantidade de acesso ao repositório (sem download – no trimestre)*
Número de Downloads do Repositório, no trimestre: (dd/mm/aa até dd/mm/aa)	*Quantidade de downloads do repositório, por trimestre (período)*
Número Acumulado de Reuso para todos os *Assets* do Repositório	*Quantidade total de reuso de assets do repositório*

NÚMERO ACUMULADO DE REUSO POR *ASSET*:

Nome:	*Nome do asset*
Reutilizado:	*Quantidade total que o asset foi reutilizado*

Nome:	*Nome do asset*
Reutilizado:	*Quantidade total que o asset foi reutilizado*

NÚMERO DE *ASSET* POR ORGANIZAÇÃO NO PERÍODO:

Nome da organização:	*Nome da organização que desenvolveu os assets*
Número de assets:	*Quantidade de assets desenvolvido*

Nome da organização:	*Nome da organização que desenvolveu os assets*
Número de assets:	*Quantidade de assets desenvolvido*

NÚMERO DE REUSO POR ORGANIZAÇÃO NO PERÍODO:

Nome da organização:	*Nome da organização que reutilizou os assets*
Número de reuso:	*Quantidade de assets reutilizados*

Nome da organização:	*Nome da organização que reutilizou os assets*
Número de reuso:	*Quantidade de assets reutilizados*

3.7.3 CONSUMO

As atividades que constituem o serviço de consumo são busca e uso de *assets* e suas especificações. As atividades executadas no processo proposto estão descritas abaixo:

3.7.3.1 BUSCAR ASSETS (A4)

No processo proposto o propósito dessa atividade é prover meios necessários para busca de *assets* candidatos à reutilização. O usuário pode fazer a sua busca através da estrutura de catálogos (busca seqüencial); busca simplificada, através de palavra chave; ou por mecanismo de pesquisa, por diferentes critérios, ao repositório. A **FIG. 3.13** apresenta o diagrama de atividades Buscar *Assets*.

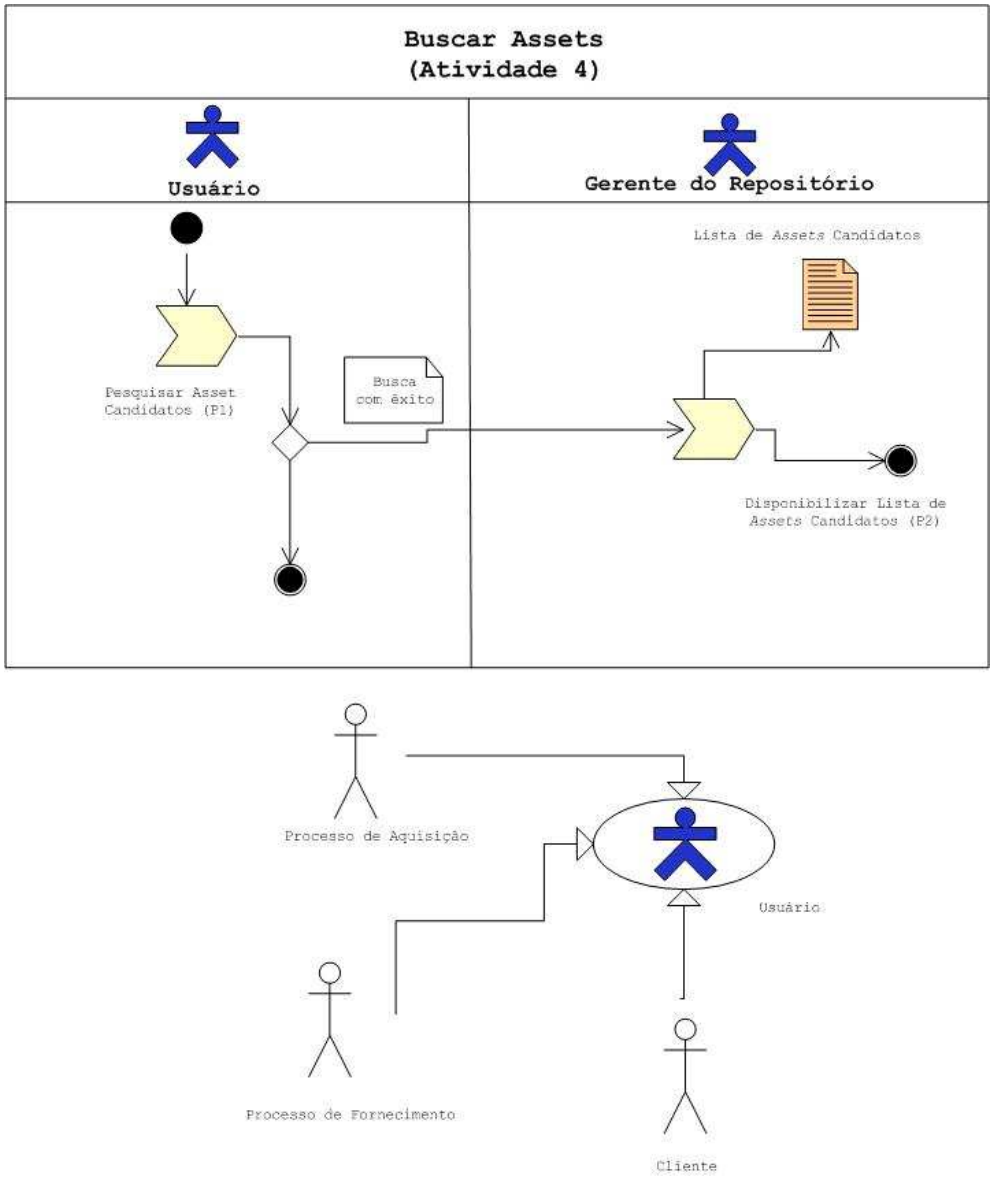


FIG. 3.13: Diagrama de atividades – Buscar *Assets*

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é possibilitar que o usuário realize buscas por *assets* no repositório, através de recursos para estabelecer critérios e parâmetros sobre as características do *asset* que está sendo procurado. Isso é possível através da seleção dos termos em cada faceta, do esquema de classificação de *assets* do repositório. Caso existam *assets* que atendam aos critérios da consulta feita pelo usuário, o sistema disponibilizará uma lista de *assets* candidatos a serem reutilizados no desenvolvimento do novo sistema.

RELACIONAMENTOS:	
papeis	Usuário (Cliente, Processo de Aquisição, Processo de Fornecimento) e Gerente do Repositório
entrada	-
saída	Lista de *Assets* Candidatos

TAREFAS:
- **Pesquisar *Asset* Candidatos (P1)** O usuário efetua a pesquisa através de uma formulação de consulta, que apresenta a relação de facetas do esquema de classificação de *assets* do repositório. O usuário, através da seleção dos termos das facetas, elabora a sua consulta. Após a seleção dos Termos, o sistema executa a solicitação da busca e pesquisa no repositório dos *assets* que atendam os parâmetros solicitados na pesquisa. - **Disponibilizar Lista de Assets Candidatos (P2)** Caso exista(m) *asset(s)* que atenda(m) ao critério de busca selecionado pelo usuário, é gerada uma Lista de *Assets* Candidatos pelo Gerente do Repositório (ver *template*). Essa lista é usada pelo usuário para avaliação dos possíveis *assets* a serem reutilizados no desenvolvimento de um novo sistema.

TEMPLATE LISTA DE *ASSETS* CANDIDATOS

Contém a relação de *assets* candidatos ao reuso;

Nome do Asset:	*Nome do asset*
Cod. Ident:	*Código de identificação do asset*
Artefatos:	*(*) Artefatos contidos no asset*
Autor:	*Nome do autor(es)*
Data:	*Data da sua elaboração*
Identificação:	*Descrição sucinta do asset*
Versão:	*Número da versão*
Acesso:	*SC (Sistema Científico) ou SA (Sistema Administrativo)*

3.7.3.2 RECUPERAR *ASSETS* (A5)

No processo proposto o propósito dessa atividade é prover meios para o usuário recuperar o(s) *asset(s)*, previamente selecionado(s), na atividade anterior (Buscar *Assets* - A4). A **FIG. 3.14** apresenta o diagrama de atividades Recuperar *Assets.*

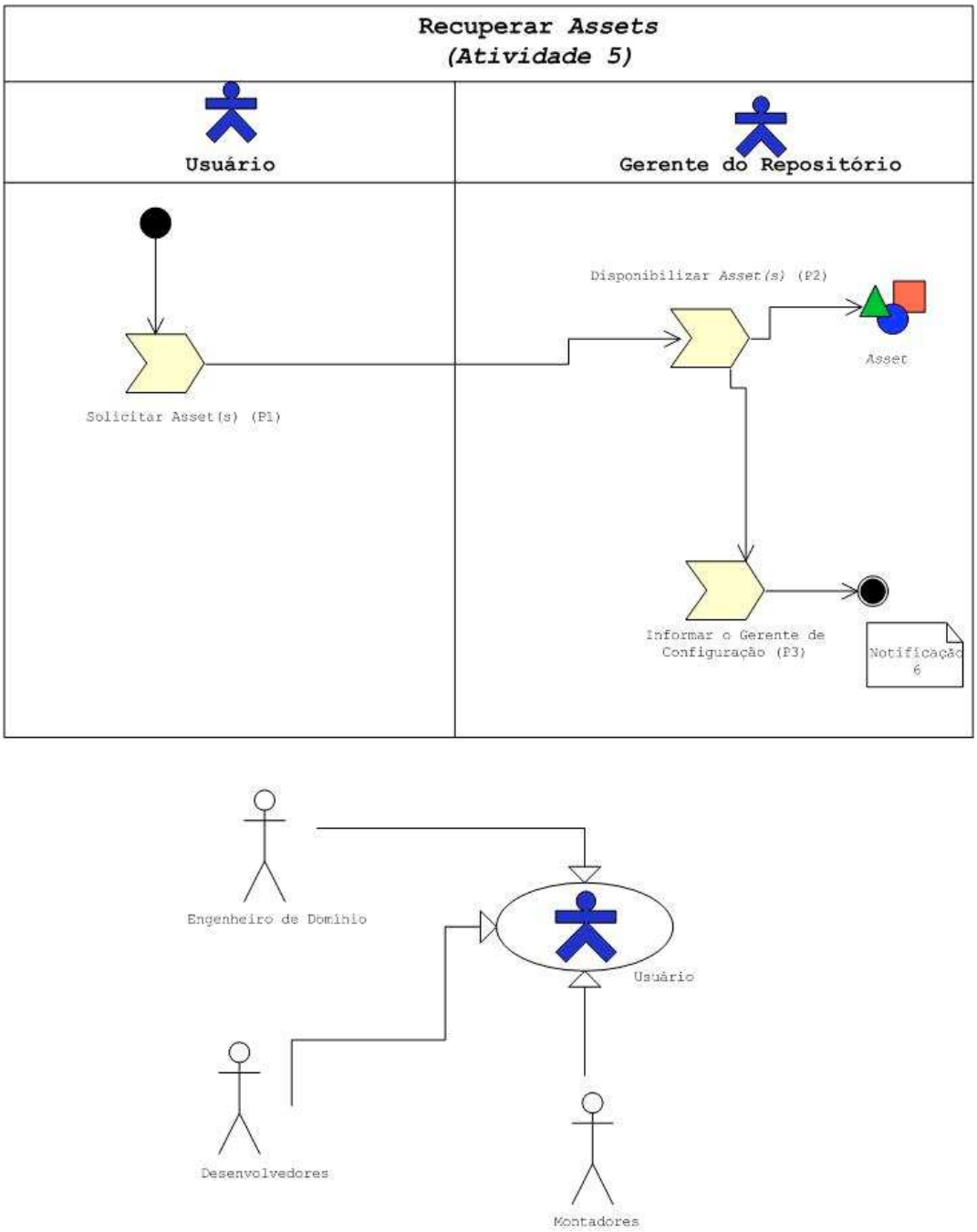


FIG. 3.14: Diagrama de atividades – Recuperar *Assets*

DESCRIÇÃO DA ATIVIDADE:
O objetivo dessa atividade é possibilitar que o usuário extraia do repositório o *asset (assets)* que será reutilizado no desenvolvimento de um novo sistema.

RELACIONAMENTOS:	
papeis	Usuário (Engenheiro de Domínio, Desenvolvedores, Montadores) e Gerente do Repositório
entrada	-
saída	*Asset(s)* e Notificação

TAREFAS:
- **Solicitar *Asset(s)* (P1)** O Usuário solicita o *asset(s)* que foi previamente selecionado através da Lista de *Assets* Candidatos. - **Disponibilizar *Asset(s)* (P2)** O Gerente do Repositório acessa o repositório, disponibiliza o *asset(s)* e envia uma notificação para o Usuário. - **Informar o Gerente de Configuração (P3)** O Gerente do Repositório informa ao Gerente de Configuração o nome do usuário e o *asset(s)* solicitado para reuso. Essa informação deverá ser inserida no Controle de Versões do sistema.

4 ESTUDO DE CASO: ANÁLISE DO PROCESSO PROPOSTO

Nesta dissertação é utilizada a estratégia de um estudo de caso por ser um método que consiste na observação de um projeto, que pode prover subsídios para verificar os prós e contras da implantação de um processo específico em um determinado contexto (WOHLIN et al., 1999).

Jacobson et al. descrevem casos reais de organizações como AT&T, HP e outras que sentiram a necessidade de uma estratégia para implementação de um processo de reuso (JACOBSON et al., 1997). Tais organizações adotaram uma metodologia incremental: desenvolver a partir de um processo gradativo, ou seja, a partir de um projeto piloto. Esta estratégia possibilita aumento em experiências e uma migração cultural passo a passo, minimizando riscos e reduzindo conflitos sociais entre as equipes de desenvolvimento.

Desta forma, uma experiência inicial de implantação da proposta desta dissertação foi realizada com o intuito de analisar a viabilidade do processo de Gerência de *Assets* para o processo proposto em (GURGEL, 2004).

Baseado no guia para estudo de caso de Kitchenham et al., o estudo de caso apresentado neste capítulo segue as etapas de Planejamento, Monitoração e Avaliação (KITCHENHAM et al., 1995).

O conjunto completo dos artefatos gerados neste estudo de caso está apresentado no Apêndice 1 desta dissertação.

4.1 PLANEJAMENTO DO ESTUDO DE CASO

4.1.1 DEFINIÇÃO DO CONTEXTO DO ESTUDO DE CASO

Para se alcançar melhores resultados com um experimento, o ideal é que ele seja um caso real. Por isso, na elaboração deste estudo de caso, além dos artefatos gerados por Gurgel

(2004) e Moore (2004), também foram utilizados os artefatos advindos do desenvolvimento de dois sistemas implantados na Marinha do Brasil, descritos abaixo:

- **Sistema de Simulação do Treinador de Ataque:** órgão desenvolvedor – IPQM;
  - **Propósito:** O Treinador de Ataque é um sistema de simulação, para uso no CAAML, com o propósito de adestrar e avaliar as equipes de guerra anti-submarino compostas pelo Avaliador, Oficial de Guerra A/S, Oficial de Manobra e Operador Sonar, em diversos níveis de dificuldade, nos procedimentos básicos de: táticas anti-submarino, busca anti-submarino, obtenção de contato, classificação de contato, medidas anti-submarino, medidas anti-torpédicas, ataque urgente e deliberado a contato submarino, procedimento fonia, procedimento telefônico, ataques coordenados e destaque de Unidade de Busca e Ataque (UBA). O Treinador de Ataque foi montado de modo a simular dois navios escoltas, limitando seu ambiente da Organização de Combate às Estações de Controle ligadas à guerra anti-submarino, excetuando-se aquelas ligadas ao controle do armamento. Para facilidade da condução dos adestramentos, este simulador possui uma estrutura que possibilita o Instrutor: gerar um exercício (conforme desejado ou existente em banco de dados), realizar um briefing antes de sua execução, interagir em sua execução e apresentar um debriefing ao seu término. O sistema levou quatro anos para ser desenvolvido e contou com uma equipe de três Analistas de Sistemas do IPQM. Ele entrou em operação no final de 2000. O sistema foi projetado utilizando uma estrutura de pacotes visando o seu reuso no desenvolvimento do SSTT2 (Sistema de Simulação Tática e Treinamento);
  - **Tecnologias Utilizadas:** A arquitetuta de hardware utilizada foi composta de 49 Pentium's II 300 Mhz, 49 Monitores de Vídeo de 21", um Projetor Panorâmico, um Sistema de Comunicação e um Chaveador de Imagens. A arquitetura de software do sistema foi baseada no Paradigma da Orientação a Objetos. Para isto, usaram-se ferramentas e linguagens que suportam este paradigma de desenvolvimento;
    - **Linguagem de Modelagem:** O Sistema foi modelado utilizando-se a linguagem UML (Unified Modeling Language), tanto na fase de análise quanto na de projeto;

- **Ferramenta Case**: foi utilizada a ferramenta de desenvolvimento ROSE da Rational Systems. Em algumas partes, durante a fase de projeto, foi usada a técnica de Engenharia Reversa para agilizar o desenvolvimento incremental;
- **Ferramenta de Documentação:** Para a fase de geração da versão final de documentação utilizou-se o aplicativo SODA, que funciona como um *Plug-in* do Rose;
- **Linguagens de Implementação:** Para implementar a maioria dos objetos foi utilizada a linguagem C++ e o compilador Visual C++ da Microsoft. Na definição das interfaces entre os objetos distribuídos foi utilizada a linguagem IDL. Na implementação de parte da funcionalidade da IHM (Interface Homem-Máquina), foi utilizada a linguagem LUA. Na IHM do sistema, visando garantir a portabilidade, foi utilizado um pacote de aplicativos, desenvolvido pelo Grupo de Tecnologia em Computação Gráfica da PUC-Rio (TecGraf);
- **Comunicação entre Objetos Distribuídos:** O padrão de comunicação entre os objetos distribuídos adotado no projeto foi o CORBA. O ORB (Object Request Broker) usado como ferramenta foi o COOL-ORB da Chorus Systems;
- **Sistema Operacional:** O sistema operacional utilizado foi o Windows NT da Microsoft. Para garantir a portabilidade, a implementação não fez uso da biblioteca de classes MFC (Microsoft Foundation Class);
- **Banco de Dados:** Os dados necessários para carregamento do exercício, são armazenados em um banco de dados relacional e atualizados através de um aplicativo escrito em ACCESS, durante a fase de preparação do exercício. O carregamento dos dados é feito no início da simulação, através de um *driver* ODBC, usado pela aplicação, a partir de um pacote de interfaceamento (DBGraf) desenvolvido pelo TecGraf;
- **Controle de Versões:** Para controlar o desenvolvimento foi utilizado o aplicativo de controle de versões Visual Source Safe da Microsoft; e
- **Controle e Planejamento de Recursos:** Para a parte de planejamento e controle dos recursos humanos e materiais, foi utilizado o Microsoft Project.

- **Sistema de Apoio à Decisão Logística:** órgão desenvolvedor - CASNAV;
  - **Propósito:** O Sistema de Apoio à Decisão Logística é um sistema computacional de apoio às atividades logísticas com ênfase na avaliação dos meios operativos. Ele foi desenvolvido com o propósito de executar as seguintes atividades:
    - Possibilitar o acesso ao histórico de falhas de equipamento dos meios navais;
    - Possibilitar a extração de dados a partir das mensagens que hoje trafegam no sistema;
    - Possibilitar a disponibilização das informações acerca das restrições dos meios; e
    - Possibilitar o acompanhamento da disponibilidade dos meios a partir das informações das suas restrições.

    O sistema computacional utilizado não poderá aumentar a carga de trabalho dos meios navais no encaminhamento das informações necessárias. O sistema também deverá:
    - Ser portável entre os sistemas operacionais Windows e Linux; e
    - Utilizar, sempre que possível softwares livres e de código aberto com vistas a minimizar os custos de manutenção e maximizar segurança dos sistemas gerados.

    A última versão, 3.0, levou um ano para ser produzida e contou com uma equipe de cinco Analistas de Sistemas. Ele entrou em operação em abril de 2007.
  - **Tecnologias Utilizadas:** Sistema para a web, desenvolvido em PHP com servidor web Apache HTTPD 2.0.59. Servidor de produção: Linux SuSe Enterprise Edition 10.0

O estudo de caso será a aplicação do processo de gerência de *assets*, seguindo os seus fluxos de atividades a fim de avaliar a viabilidade do processo proposto.

4.1.2 HIPÓTESE DO ESTUDO DE CASO

A hipótese formulada é que o processo proposto atenda as atividades especificadas na IEEE 1517 (1999), que são: gerenciar a certificação, classificação, armazenamento, busca, controle de versões e controle de alterações dos *assets*,

A viabilidade do processo será comprovada pela sua execução. A avaliação quanto à melhoria do processo de reuso será baseada nos resultados obtidos durante a execução do processo.

Para a sua condução, houve a necessidade da criação de algumas situações e complementação de informações com o objetivo de que fossem abordadas as diversas atividades do processo proposto.

4.1.3 CONDUÇÃO DO ESTUDO DE CASO

O estudo de caso seguiu o processo de Gerência de *Assets* no Desenvolvimento de *Software* Baseado em Reuso proposto no capítulo anterior, tendo como subsídio os artefatos dos sistemas descritos na sub-seção 4.1.1. Cada atividade do processo foi conduzida conforme a etapa Monitoração de (KITCHENHAM et al., 1995) e será apresentada na sub-seção 4.2.

4.1.4 PARTICIPANTES DO ESTUDO CASO

O estudo de caso foi executado por uma pessoa e teve a duração total de seis meses. Houve a colaboração do pessoal envolvido no desenvolvimento dos artefatos utilizados neste estudo de caso por meio de entrevistas para o seu entendimento.

4.1.5 OBJETIVO DO ESTUDO DE CASO

O objetivo deste Estudo de Caso é analisar a viabilidade do processo proposto, no que tange a gerência de *assets*, permitindo o reuso dos artefatos criados durante a etapa de desenvolvimento de um sistema.

4.2 MONITORAÇÃO DO ESTUDO DE CASO

4.2.1 CENÁRIOS DO ESTUDO DE CASO

TAB. 4.1: Cenários de Estudo de Caso

CENÁRIO	DESCRIÇÃO
1	Aprovar *Asset* de Análise de Domínio do Controlador Tático
2	Armazenar *Asset* de Análise de Domínio do Controlador Tático
3	Armazenar *Asset* de Arquitetura de Domínio do Controlador Tático
4	Armazenar *Asset* de Componente de Domínio do Controlador Tático
5	Armazenar *Asset* de Análise de Domínio do Treinador de Ataque
6	Armazenar *Asset* de Arquitetura de Domínio do Treinador de Ataque
7	Armazenar *Asset* de Componente de Domínio do Treinador de Ataque
8	Armazenar *Asset* de Componente do Treinador de Ataque
9	Armazenar *Asset* Análise de Domínio do Sistema de Apoio à Decisão Logística
10	Armazenar *Asset* de Arquitetura de Domínio do Sistema de Apoio à Decisão Logística
11	Armazenar *Asset* de Componente de Domínio do Sistema de Apoio à Decisão Logística
12	Armazenar *Asset* de Componente do Sistema de Apoio à Decisão Logística
13	Cadastrar Usuário do CASOP
14	Cadastrar Usuário do IME
15	Cadastrar Usuário do ITA
16	Buscar *Assets* no domínio de sistemas táticos
17	Buscar *Assets* no domínio de sistemas de apoio à decisão
18	Buscar *Assets* no domínio de simuladores
19	Recuperar Asset do domínio de sistemas táticos
20	Recuperar Asset do domínio de sistemas de apoio à decisão
21	Recuperar Asset do domínio de simulador
22	Registrar Problema do Componente do Sistema de Apoio à Decisão Logística
23	Atualizar *Asset* de Análise de Domínio do Controlador Tático
24	Elaborar Relatório de Métricas

4.2.2 CENÁRIO DE APROVAÇÃO DE *ASSET*

A Equipe de Desenvolvedores submete o *asset* de análise de domínio do Controlador Tático para certificação. O Gerente de Qualidade valida via checklist características de reusabilidade, compatibilidade, padronização da documentação, rastreabilidade, corretude, facilidade no entendimento das funcionalidades e objetivos do *asset*. Após ter a sua qualidade aprovada o Gerente de Qualidade envia o *asset* para ser armazenado no repositório pelo Gerente do Repositório.

TAB. 4.2: Atividade Aprovar *Asset*

Passo 1	A equipe de desenvolvimento submete *asset* à certificação
Passo 2	O Gerente de Qualidade aplica testes no *asset* e valida a sua qualidade para o reuso
Passo 3	Após o *asset* ter sido, aprovado o Gerente de Qualidade envia o *asset* para ser armazenado pelo Gerente do Repositório.

4.2.3 CENÁRIOS DE ARMAZENAGEM DE *ASSETS*

A equipe de desenvolvimento dos sistemas Controlador Tático, Treinador de Ataque e Sistema de Apoio à Decisão Logística desejam armazenar os artefatos produzidos para futuro reuso em outros projetos. Para isso eles enviam uma notificação para o Gerente de Qualidade certificar os *assets* candidatos à reutilização. O Gerente de Qualidade aprova os seguintes *assets* candidatos:

- Análise de Domínio do Controlador Tático;
- Arquitetura de Domínio do Controlador Tático;
- Componente de Domínio do Controlador Tático;
- Análise de Domínio do Treinador de Ataque;
- Arquitetura de Domínio do Treinador de Ataque;
- Componente de Domínio do Treinador de Ataque;
- Componente do Treinador de Ataque;
- Análise de Domínio do Sistema de Apoio à Decisão Logística;
- Arquitetura de Domínio do Sistema de Apoio à Decisão Logística;

- Componente de Domínio do Sistema de Apoio à Decisão Logística; e
- Componente do Sistema de Apoio à Decisão Logística.

Após a certificação o Gerente de Qualidade envia uma notificação para o Gerente do Repositório armazenar os *assets* no sistema. Após os *assets* serem armazenados no repositório, o Gerente do Repositório envia uma notificação para o Gerente de Configuração incluir os *assets* no Controle de Versões.

TAB. 4.3: Atividade Armazenar *Asset*

Passo 1	O Gerente de Qualidade envia notificação para o Gerente do Repositório
Passo 2	O Gerente do Repositório classifica o *asset*
Passo 3	O Gerente do Repositório cataloga o *asset*
Passo 4	O Gerente do Repositório notifica o Gerente de Configuração para inclusão no Controle de Versões.

TAB. 4.4: Atividade Controlar Versões - Cadastrar

Passo 1	O Gerente de Configuração recebe notificação e verifica se é um novo *asset* para ser cadastrado
Passo 2	O Gerente de Configuração cadastra o *asset* no Controle de Versões

4.2.4 CENÁRIOS DE CADASTRO DE USUÁRIOS

Os usuários relacionados abaixo desejam reutilizar *assets* em seus projetos e solicitam cadastro no sistema de Gerência de *Assets*:

- Tecnologista Sênior III Luís Pontes, servidor do CASOP, Marinha;
- Major Alexandre Magno militar do IME, Exército; e
- Primeiro Tenente Renato Alves, militar do ITA, Aeronáutica.

TAB. 4.5: Atividade Cadastrar Usuário

Passo 1	O Usuário submete pedido de acesso ao sistema
Passo 2	O Gerente do Repositório verifica se o usuário pertence a um dos orgãos do Ministério da Defesa (ver TAB. 7, 8, 9 e 10)
Passo 3	O Gerente do Repositório cadastra o usuário e envia uma notificação informando se o seu pedido foi aceito ou não.

4.2.5 CENÁRIOS DE PROCURA DE *ASSETS*

Devido ao desenvolvimento de novos sistemas os usuários do sistema executam buscas no repositório a fim de encontrar *assets* a serem reutilizados em seus projetos A busca a ser executada obedece aos critérios definidos pelos seguintes termos das facetas:

- Major Alexandre Magno:
  - Domínio: sistemas táticos; e
  - Atividade: análise de domínio.
- Tenente Renato Alves:
  - Domínio: Sistema de Apoio à Decisão Logística;
  - Tipo de *Asset:* componentes; e
  - Tipo de Componente: classe.
- Tecnologista Sênior III Luís Pontes:
  - Domínio*:* simuladores; e
  - Atividade: projeto de domínio.

TAB. 4.6: Atividade Buscar Asset

Passo 1	Os usuários pesquisam possíveis *assets* candidatos através da seleção dos termos de busca que obedecem ao esquema de classificação em facetas. Para efetuar uma busca não há a necessidade de preencher todos os termos das facetas
Passo 2	O Gerente do Repositório gera uma lista de *assets* candidatos que atendam aos critérios de pesquisa feitos pelo usuário

4.2.6 CENÁRIOS DE RECUPERAÇÃO DE *ASSETS*

Os usuários que anteriormente efetuaram busca no sistema, solicitam os *assets* ao Gerente do Repositório para ser reutilizado. Após os *assets* serem enviados aos usuários, o Gerente do Repositório envia uma notificação para o Gerente de Configuração cadastrar as informações no Controle de Versões. O Gerente de Configuração recebe notificação e inclui as informações dos usuários que recuperaram os *assets* no Controle de Versões.

- Major Alexandre Magno: *asset* - Análise de Domínio do Controlador Tático;

- Tenente Renato Alves: *asset* - Componentes do Sistema de Apoio à Decisão Logística; e
- Tecnologista Sênior III Luís Pontes: *assets* - Arquitetura de Domínio do Treinador de Ataque e Componente de Domínio do Treinador de Ataque.

TAB. 4.7: Atividade Recuperar *Asset*

Passo 1	Os usuários solicitam os *assets* para o Gerente do Repositório
Passo 2	O Gerente do Repositório acessa o repositório, disponibiliza os *assets* e envia uma notificação para os Usuários.
Passo 3	O Gerente do Repositório informa ao Gerente de Configuração o nome dos usuários e os *assets* solicitados para reuso. Essas informações deverão ser inseridas no Controle de Versões.

TAB. 4.8: Atividade: Controlar Versões - Recuperar

Passo 1	Armazenar Usuário/ *Asset* no Controle de Versões

4.2.7 CENÁRIO DE REGISTRO DE PROBLEMA

O Tenente Renato Alves faz o registro de um problema detectado no *asset* Componentes do Sistema de Apoio à Decisão Logística. O Gerente de Configuração efetua o registro e envia uma notificação aos possíveis usuários que também estejam reutilizando o mesmo *asset*.

TAB. 4.9: Atividade Registrar Problemas

Passo 1	O Tenente Renato Alves registra o problema detectado e envia uma notificação para o Gerente de Configuração.
Passo 2	O Gerente de Configuração cadastra a notificação do Ten. Renato Alves no sistema
Passo 3	Gerente de Configuração verifica, no Controle de Versões, usuários que estejam reutilizando esse *asset* e envia uma notificação informando sobre o problema detectado. Neste cenário, não há nenhum outro usuário reutilizando o *asset*

4.2.8 CENÁRIO DE ATUALIZAÇÃO DE *ASSET*

O Major Alexandre Magno gera uma nova versão do *asset* de Análise de Domínio do Controlador Tático que ele reutilizou no desenvolvimento do Sistema Tático da Divisão de Tanques. Ele envia o *asset* para o Gerente Qualidade certificá-lo. Após a certificação, o Gerente de Qualidade envia uma notificação para o Gerente do Repositório armazená-lo no sistema. Após o *asset* ser armazenado no repositório, o Gerente do Repositório envia uma notificação para o Gerente de Configuração atualizar as informações do *asset* no Controle de Versões.

TAB. 4.10: Atividade Aprovar Asset

Passo 1	O Engenheiro de Domínio ou Desenvolvedores ou Montadores, envia(m) uma notificação para o Gerente de Qualidade informando que há um *asset* candidato para ser certificado pelo Processo de Qualidade
Passo 2	O Gerente de Qualidade valida via *checklist* características de reusabilidade, compatibilidade, padronização da documentação, rastreabilidade, corretude, facilidade no entendimento das funcionalidades e objetivos do *asset*
Passo 3	A partir da qualidade aprovada, o Gerente de Qualidade gera o relatório de revisão
Passo 4	O Gerente de Qualidade envia o *asset* aprovado e uma notificação para o Gerente do Repositório

TAB 4.11: Atividade: Atualizar Asset

Passo 1	O Gerente de Qualidade envia notificação para o Gerente do Repositório
Passo 2	O Gerente do Repositório cataloga a nova versão do *asset* de Análise de Domínio do Controlador Tático
Passo 3	O Gerente do Repositório notifica o Gerente de Configuração para atualização das informações no Controle de Versões.

TAB. 4.12: Atividade Controlar Versões - Cadastrar

Passo 1	O Gerente de Configuração verifica se já existe cadastro do *asset* no controle de versões. Como foi encontrado registro desse asset, o Gerente de Configuração atualiza suas informações no sistema
Passo 2	O Gerente de Configuração atualiza as informações do *asset* no Controle de Versões

4.2.9 CENÁRIO DE ELABORAÇÃO DE RELATÓRIO DE MÉTRICAS

O Gerente de Configuração, ao final de um trimestre, gera um relatório de métricas com as seguintes informações:

- Número de *assets* no repositório;
- Número de acessos (sem *download*) ao repositório, por período;
- Número de *downloads* no repositório, por período;
- Número acumulado de reuso, por *asset*;
- Número acumulado de reuso para todos os *assets* do repositório;
- Número de *assets* desenvolvidos, por organização, no período; e
- Número de reuso, por organização, por período.

4.3 AVALIAÇÃO DO ESTUDO DE CASO

Através dos resultados obtidos na aplicação do processo nos *assets* do Controlador Tático, Treinador de Ataque e Sistema de Apoio à Decisão Logística, tanto a viabilidade do processo e a hipótese formulada são válidas, embora a estrutura organizacional definida para a proposta do Processo de Gerência de *Assets* não tenha sido implantada, pois envolveria vários Órgãos Militares das 3 Forças sem um meio adequado de comunicação. Esta estrutura deve ser implantada justamente na fase 3 (Ações Integradas) do Programa de Reuso definido por GURGEL (2004) - sugestão para trabalho futuro, onde deverá ser avaliado o fluxo de informação entre o Processo de Gerência de *Assets*, o Processo de Garantia da Qualidade e o Processo de Desenvolvimento de *Software* e os problemas culturais oriundos da nova estrutura nas organizações militares fornecedoras de software.

É importante ressaltar que foi utilizado um conjunto razoavelmente pequeno de *assets*, mas que foram suficientes para execução do estudo de caso e para avaliação da contribuição do processo quanto à melhora da sistemática do reuso.

Em relação ao emprego do processo no apoio ao processo de desenvolvimento de software (GURGEL, 2004), podem-se citar algumas observações:

- A importância da utilização de uma base comum de *assets* (repositório), visando-se retrabalho desnecessário, no caso da existência de algum tipo de solução de software aplicável ao problema em questão;
- A atividade de gerência cumpriu o seu papel de apoio às atividades de produção e consumo;
- O Controle de Métricas mostrou-se eficiente em coletar dados para avaliação do sucesso do programa de reuso segundo os objetivos estabelecidos por Ezran (2002);
- Houve uma abordagem prática de certificação de *assets* através da interação com o Processo de Qualidade (Moore, 2004);
- E finalmente, houve um nível de sistematização do processo de reutilização de *software*, no que diz respeito ao consumo e à produção de *assets* em um repositório.

Podemos citar algumas dificuldades encontradas durante a aplicação do experimento:

- Os artefatos utilizados no estudo de caso não estavam acompanhados da sua documentação. Houve a necessidade de gerar todos os documentos para a execução do processo. Foi gasto um grande esforço nesta tarefa, visto que não se possuía o conhecimento técnico dos artefatos que foram utilizados neste experimento;
- Durante a execução do estudo de caso precisou-se redefinir o processo de gerência de *assets* para se adequar à realidade do trabalho proposto. Algumas atividades como armazenamento, atualização, controle de versões e cálculo de métricas tiveram que ser alteradas durante a aplicação do processo;
- Como no item anterior, alguns artefatos do estudo de caso, como documento de classificação, catálogo, controle de versões e relatório de métricas tiveram que ser alterados ao longo da execução do processo para também refletirem a realidade do processo proposto;
- O estudo de caso consistiu na implementação de um grande número de cenários para testar todas as atividades do processo. Isto resultou na criação de um grande volume de artefatos de saída. A realização dessas atividades sem o apoio de uma ferramenta mostrou que a execução do processo torna-se difícil sem a utilização de soluções automáticas. Essa solução estaria baseada na utilização de ferramentas CASE .

Conclusão: Segundo os resultados alcançados, podemos concluir que a hipótese formulada é válida.

Quanto à viabilidade do processo, esta foi comprovada pela execução do estudo de caso. O processo, dentro do contexto no qual ele foi desenvolvido, realizou satisfatoriamente todas as atividades especificadas, mostrando que é passível ser aplicado dentro desta estrutura.

O processo proposto atendeu as atividades de gerência de *assets*, definidas na IEEE 1517 (1999), aplicado ao Programa de Reuso definido por GURGEL (2004) dentro do contexto do Ministério da Defesa e de seus Comandos Subordinados.

5 CONCLUSÕES

O Processo de Gerência de *Assets* (PGA), desenvolvido neste trabalho, atingiu seu objetivo de apoiar o Processo de Desenvolvimento de *Software* baseado em Reuso de GURGEL (2004), visando capacitar as organizações militares de desenvolvimento de *software* na produção de *assets* e promover o seu compartilhamento para reutilização no processo de desenvolvimento de sistemas (consumo) por meio de um repositório de artefatos reutilizáveis (*assets*).

Fundamentado na Norma ISO/IEC 12207 e de sua extensão ao reuso IEEE 1517 (1999), o Processo de Gerência de *Assets* proposto definiu um padrão de Gerência de *Assets* específico para gerenciar a sua vida útil, desde a sua concepção até a sua retirada do sistema. Para isso, foram especificadas as atividades do processo, os papeis e os artefatos gerados durante a sua execução. O ambiente de apoio ao processo de reutilização de software baseou-se nos serviços de consumo, gerência e produção definido por Apperly (2001).

O processo proposto atuou no reuso sistemático de *software*, disponibilizando os recursos de armazenamento, busca, divulgação, compartilhamento e entendimento de *assets* para a construção de sistemas de software, auxiliando no sucesso do desenvolvimento de software baseado em reuso.

Uma consideração importante foi observada durante a elaboração deste trabalho:

- O trabalho de pesquisa realizado comprovou a necessidade de ferramentas que forneçam suporte às atividades de produção, consumo e gerência de *assets*. As tarefas repetitivas necessitam do auxílio de ferramentas CASE para serem executadas.

Conclui-se que o reuso é uma estratégia de médio-longo prazo no qual a empresa constrói uma biblioteca de *assets* usados freqüentemente de modo a permitir que novos programas sejam construídos mais rapidamente a partir dos *assets* existentes nessa biblioteca. Existem muitos riscos envolvidos nesse processo, mas quando suportado por uma gerência de *assets*, o reuso pode trazer grandes benefícios em termos de redução de prazos e custos, maior que outras práticas de desenvolvimento de software. Além disso, o reuso pode ser aplicado teoricamente por qualquer organização para qualquer tipo de software ou projeto de software.

5.1 BENEFÍCIOS

O principal benefício deste trabalho é o desenvolvimento de um processo de gerência de *assets* que se integre plenamente ao processo de reuso sistemático proposto por GURGEL (2004) para o Ministério da Defesa e seus Comandos Subordinados. Com base neste processo de gerência de *assets*, pode-se observar que a combinação da reutilização de software com as técnicas de identificação, seleção e entendimento do *asset*, é uma alternativa para alcançar a produtividade na elaboração de um produto de software com qualidade, visando custos mais baixos no processo de desenvolvimento.

Visto isso podemos citar os seguintes benefícios com a implantação do processo proposto nesse trabalho:

- Utilização de um processo de certificação de *assets* responsável por testar a sua qualidade para reuso;
- Utilização de um sistema de classificação (facetas) responsável por auxiliar a busca e recuperação dos *assets;*
- Definição de um modelo de armazenamento (catalogação) baseado na RAS (2005);
- Elaboração de um relatório de métricas visando fornecer informações sobre o grau de reuso dentro da organização;
- Registro dos problemas detectados nos *assets* reutilizados e compartilhamento dessas informações para futuros reutilizadores;
- Padronização das atividades de gerência, consumo e produção de *assets* dentro da organização;
- Definição dos papéis a serem executados no processo; e
- Definição de todos os artefatos necessários para a condução do processo.

A utilização desse processo também trouxe benefícios na sistemática do reuso como os citados a seguir:

- Simplificação do desenvolvimento de software;
- Redução de custos, prazos de entrega e esforço para se desenvolver e manter o software;
- Evita repetição e redundância no desenvolvimento;
- Aumento de produtividade;

- Desenvolvimento de softwares de maior qualidade, e portanto, de maior confiabilidade;
- Redução de erros;
- Evita Repetições (redundâncias entre sistemas); e
- Conhecimento sobre sistemas e como construí-los podem ser compartilhados.

Para maior compreensão desses benefícios agrupamos-os em três tipos de melhorias: **produtividade**, **qualidade** e **desempenho do negócio**.

Em termos de **produtividade**, podemos destacar o fato de que o reuso significa menos esforço da equipe porque significa menos escrita, ou seja, não há "reinvenção da roda". Em consequência dessa diminuição de esforço, pode-se perceber que também há redução de custos e tempo necessários para se desenvolver um sistema.

A melhoria da **qualidade** pode até ser considerada como uma consequência do que foi dito no parágrafo anterior, pois se os esforços num projeto são substancialmente reduzidos, possivelmente existirão menos defeitos. Outro aspecto que conduz ao aumento de qualidade é quando se considera que uma vez que o *asset* tenha qualidade comprovada em um projeto, ele poderá ser carregado para outros. O aumento de qualidade gera menores custos de manutenção e serviços.

Quanto à melhoria de **desempenho do negócio** podemos incluir os já citados aspectos de redução de custos no desenvolvimento de novos sistemas. Esse benefício é muito importante no âmbito federal, neste caso, no contexto do Ministério da Defesa onde muitas vezes ocorrem restrições orçamentárias que impactam na redução do desenvolvimento de novas tecnologias.

Outro benefício do processo reside no detalhamento de como executar as atividades, tarefas incluídas no processo, responsabilidades a agentes e artefatos gerados durante a sua execução. O processo de reuso da IEEE 1517 (1999) apenas descreve um *framework* de alto nível, mas não os detalhes de como executar esse processo.

Cabe salientar a possibilidade de utilização da proposta em outras organizações que não o Ministério da Defesa, desde que o desenvolvimento baseado em reuso, no que tange aos processos de engenharia de domínio e de desenvolvimento, já tenham sido implantados na instituição.

5.2 TRABALHOS FUTUROS

Os futuros trabalhos que podem ser realizados para melhoria do Processo de Gerência de *Assets* são:

- Desenvolver uma ferramenta de auxílio ao processo que implemente os serviços de suporte ao processo de gerência de *assets* definidos nessa proposta;
- Desenvolver adaptadores de *assets* para serem utilizados como solução na elaboração de um novo *asset (asset* + adaptador = novo *asset*);
- Quanto ao esquema de classificação e recuperação dos *assets* no repositório, é necessário desenvolver pesquisas para avaliar a proposta apresentada neste trabalho e outras propostas da literatura para definir o esquema da ferramenta;
- Implementar o repositório de dados baseado na RAS (2005), quanto ao esquema de catalogação e dados;
- A partir da implementação da estrutura e dados do repositório (RAS 2005), definir uma documentação padrão para os *assets;*
- Criar uma nova atividade que analise os resultados do cálculo de métricas. A partir desses resultados a nova atividade deverá auxiliar a equipe de Gerência de *Assets* a verificar os *assets* armazenados no repositório que estão obsoletos e o sucesso da sistemática de reuso dentro da organização;
- Ampliar a estratégia proposta e além da gerência de *assets* também considerar os aspectos humanos para o sucesso do processo de reuso;
- Realizar novos estudos de casos para verificar o grau de facilidade de aplicação da estratégia proposta, o fluxo de informação, as resistências e os problemas culturais considerando um ambiente com a estrutura organizacional proposta; e
- Realizar experimentos mais formais e detalhados do processo proposto.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ABNT9000 - ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS: **ABNT/NBR ISO 9000** – Sistemas de gestão da qualidade, ABNT/CB-25 – Comitê Brasileiro da Qualidade, 2001.

ABNT9001 - ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS: **ABNT/NBR ISO 9001** - Sistemas de gestão da qualidade – Requisitos, 2001.

ABNT12207 - ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS: **ABNT/NBR ISO/IEC 12207** – Tecnologia da Informação – Processo de ciclo de vida de software, Out. 1998.

ALMEIDA, K. M., STAA, A. V. Avaliação da qualidade em um modelo de ciclo de vida orientado a reuso. **III Simpósio de Desenvolvimento e Manutenção de Software**, Sessão Técnica VI, 10 p., ago 2003.

APPERLY, H. Configuration Management and Component Libraries. In: HEINEMAN, G. T.; COUNCIL, W. T. **Component-Based Software Engineering.** 1a ed. Addison-Wesley, 2001. p. 513-526.

ARANGO, G., Domain Engineering for Software Reuse, **Technical Report UCI-ICS**, 88-27, Universidade da Califórnia, 1988.

BASILI, V.R.; CALDIERA, G.; CANTONE, G. A Reference Architecture for the Component Factory. **ACM Transactions on Software Engineering and Methodology**, v.1, n.1, p. 53-80, January 1992.

BASILI, V.R.; BRIAND, C.; MELO, W.L. How Reuse Influences Produtivity in Object-Oriented Systems. **Communications of the ACM**, v.39, n.10, p.104-126, Octuber 1996.

BOEHM, B. W., PAPACCIO, F. N. Understanding and controlling software costs. **IEEE Transactions on Software Engineering**, v.14, n. 10, Oct. 1988.

BOOCH, G; RUMBAUGH, J.; JACOBSON, I. **The Unified Modeling Language:User Guide.** Addison-Wesley, Reading, MA, Fev 1998.

CASANOVA, M.; STRAETEN, R. V.; JONCKERS, V. Supporting Evolution in Component-Based Development using Component Libraries. Proceedings of the Seventh European Conference on Software Maintenance and Reengineering (CSMR'03), **IEEE Computer Society**, 2003.

CHEESMAN, J., DANIELS, J. **UML Components: a simple process for specifying component-based software**. Boston: Addison Wesley, 2000. 176 p

CROSBY, P. B. **Quality is still free: making quality certain in uncertain times**. Mc Graw-Hill, 1996. 264 p.

DAMIANI, E.; FUGINI, M.G.; FUSASCHI, E. A Descriptor-Based Aproach to OO Code Reuse. **Computer**, p. 73-80, October 1997.

EZRAN, Michel; MORISIO, Maurizio; TULLY, Colin. **Practical Software Reuse**. Springer Verlag 2002.

FARIA, Henrique Rocha de. **Um Modelo de Processo de Apoio ao Desenvolvimento de Software Baseado em Componentes, Orientado a Qualidade**, **e Centrado em um Repositório**. Dissertação (Mestrado em Engenharia de Software) . Escola Politécnica da Universidade de São Paulo, São Paulo, 2005.

FICHMAN, R. G. Incentive compatibility and systematic software reuse. **Journal of Systems and Software**, New York, v. 57, iss. 1, April 27, 2001. 45 p.

FIORINI, S. T., STAA, A., BAPTISTA, R. **Engenharia de software com CMM**. Rio de Janeiro: Brasport, 346 p., 1998

FRAKES, W.B.; GANDEL, P.B. Representing Reusable Software. **Information and Software Technology**, v. 35, n.10, p.653-664, 1990.

FRAKES, W.B.; ISODA, S. Success Factors of Systematic Reuse. **IEEE Software**, v.11, n.5, pp.15-19, September 1994.

FRAKES, W. B., NEJMEH, B.A. Software Reuse through Information Retrieval. In: **Proceedings of the Twentieth Annual Hawaii International Conference on Systems Sciences, Kona**, p. 530-535, Jan 1987.

FRAKES, W., POLE, T. An empirical study of representation methods for reusable software components. **IEEE Transactions on Software Engineering**, v. 20, n. 8, p. 617-630, Aug. 1994.

FRAKES, W., FOX, C. **Sixteen questions about software reuse. Communication ACM**, v. 38, n. 6, p. 75-87, 1993.

FRAKESa, W. B., FOX, C. J. Quality improvement using software reuse failure modes model. **IEEE Transactions on Software Engineering**, v. 22, n. 4, p. 274-279, April 1996.

FRAKESb, W. B., TERRY, C. Software reuse metrics and models. **ACM Computing Surveys**, v. 28, n. 2, p. 415-435, 1996.

FREEMAN, P., Reusable Software Engineering: A statement of long-range research objectives, **Technical Report TR-159**, Universidade da Califórnia. October 1980.

FREEMAN, P.,Reusable Software Enginnering: concepts and Research Directions. **Proocedings of the Workshop on Reusability in Programming**, pp. 129-137. ITT, September 1983.

FREEMAN, P., **A conceptual Analysis of the Draco Approach to Constructing Software Systems, Trans. On Software Enginnering**, SE-13(7): 830-844, July 1984.

GAFFNEY, J. E. Jr., DUREK, T. A. Software reuse-key to enhanced productivity: some quantitative models. **Information Software Technology**, German, v.31, n. 5, p. 258-267, 1989.

GAMMA, E.; HELM, R.; JOHNSON, R. VLISSIDES, J. **Design Patterns: Elements of Reusable Object-Oriented Software**. Addison-Wesley, 1995. 416 p.

GIOVANI, P.; MELNIKOFF, S. S.S**.** Reutilização em Software Orientado a Objetos: Um Estudo Empírico para Analisar a Dificuldade de Localização e Entendimento de Classes. **Boletim Técnico da Escola politécnica da USP**, Departamento de Engenharia de Computação e Sistemas Digitais, BT/PCS/9612, 1996.

GRISS, M. CBSE Success Factors: Integrating Architecture, Process, and Organization. In: HEINEMAN, G. T.; COUNCIL, W.T. **Component-Based Software Engineering: Putting the Pieces Together**. 1a.ed, Addison-Wesley, 2001.

GUO, J.; LUQI A Survey of Software Reuse Repositories. In: **Proceedings, The Seventh IEEE International Conference and Workshop, Engineering of Computer Based Systems**, p. 92-100, 2000.

GURGEL, M. S. **Estudo do Desenvolvimento de Software Baseado em Reuso no Contexto do Ministério da Defesa e de seus Comandos Subordinados**. Dissertação de Mestrado, Instituto Militar de Engenharia - IME, 2004.

HAZAN, C. **Metodologia para o uso de indicadores na gerência de projetos de desenvolvimento de software**. Tese de Mestrado, Instituto Militar de Engenharia - IME, 1999.

HENNINGER, S. An Evolutionary Approach to Constructing Effective Software Reuse Repositories. **ACM Transactions on Software Engineering and Methodologies**, v.16, n.2, p. 111-140, April 1997.

HOPKINS, J. Component Primer: Laying the Foundation**. Communications of the ACM**, v. 43, n.10, p. 27-30, October 2000.

IEEE1517 - INSTITUTE OF ELECTRICAL AND ELETRONICS ENGINEERS: IEEE Standard 1517-1999, Software life cycle processes – reuse processes, **IEEE Standard for Information Technology**, 1999.

ISO15504 - INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: **ISO/IEC TR 15504**, Information technology – software process assessment, 1998. Disponível: http://www.sqi.gu.edu.au/spice . Acesso: fev 2006.

ISO/IEC 12207:1995 - **Information Technology – Software Life Cycle Processes.** Geneva, 1995**.**

JACOBSON, I.; BOOCH, G; RUMBAUGH, J. **The Unified Software Development Process**. New Jersey: Addison-Wesley, 1998. 463p.

JACOBSON, I.; GRISS, M.; JONSSON, P. **Software Reuse: Architecture, Process and Organization for Business Success**. 1a. ed, ACM Press, 1997.

KAN, S. H. **Metrics and models in software quality engineering**. Addison Wesley, 1995. 361 p.

KITCHENHAM, B., PICKARD, L. PFLEEGER, S. L. Case studies for method and tool evaluation. **IEEE Software**, Los Angeles, v. 12, n. 4, p. 52-62, Jul. 1995.

KLABUNDE, Charles C. **Um Estudo Sobre Componentes de Software**. Curso de Pós-Graduação em Ciência da Computação – UFRGS – 1999.

KRUEGER, C., Software Reuse**, ACM Computing Surveys**, v. 24, n. 2, pp. 131-183, 1992.

KRUCHTEN, P. Modeling Component Systems with the Unified Modeling Language**, International Workshop on Component-Based Software Engineering** (Position Paper), 1998.

LIM, W.C. **Managing Software Reuse**, 1. ed., New Jersey: Prentice Hall PTR, 1998.

LUCRÉDIO, DANIEL. **Extensão da Ferramenta MVCASE com Serviços Remotos de Armazenamento e Busca de Artefatos de Software.** Dissertação de Mestrado, São Carlos : UFSCar, 2005.

MCCLURE, C. **Software Reuse Techniques**: Adding Reuse to the Ssytem Development Process. 1ª ed., New Jersey: Prentice Hall PTR, 1997.

McILROY, M., **Mass produced Software Components, from 1969 NATO Conference in Software Engineering.** Sofware Enginneering Concepts and Techniques**,** pp. 88-98. Petrocelli/Cahrter, Bruxelas, 1976.

MCCONNELL, STEVE. **Rapid Development: Taming Wild Software Schedules**. Microsoft Press 1996

MINISTÉRIO DA DEFESA**.** Disponível: <http://www.defesa.gov.br/>. Acesso em mar 2007.

MOORE, Kleyna Almeida. **Garantia da Qualidade de Software no Desenvolvimento Baseado em Reuso: Uma Abordagem no Contexto do Ministério da Defesa e de seus Comandos Subordinados**. Dissertação de Mestrado, Instituto Militar de Engenharia - IME, 2004.

MORISIO, M.; TULLY, C.; EZRAN, M. Diversity in Reuse Process. **IEEE Software**, Los Alamitos, v. 17, n. 4, p. 56-63, jul./ago. 2000.

NEIGHBORS, J., **Software Construction Using Components**. Tese (Doutorado em Ciências) - Universidade da Califórnia, Irvine, Estados Unidos, 1981.

OMG, **Unified Modeling Language (UML 2.1.1) Specification**. Disponível em: < http://www.omg.org/technology/documents/formal/uml.htm>. Acesso em: jan 2006.

OMG, **Meta-Object Facility (MOF 2.0) Specification**,. Disponível em: <http://www.omg.org/cgi-bin/doc?ad/2003-04-08 >. Acesso em: jan 2006.

OMG, **Reusable Asset Specification (RAS 2.2) , 2005**. Disponível em: < http://www.omg.org/technology/documents/formal/ras.htm >. Acesso em: jan 2006.

OMG, **Software Process Engineering Metamodel (SPEM 1.1), 2005**. Disponível em: <http://www.omg.org/technology/documents/formal/spem.htm >. Acesso em: jan 2006.

PALADINI, E. P. **Qualidade total na prática: implantação e avaliação de sistemas de qualidade total.** 2ª Edição, São Paulo: Atlas, 1994.

PAULK, M. C., CURTIS, B., CHRISSIS, M. B., WEBER, C. V. Capability maturity model for software. Version. 1.1**,** Technical Report CMU/SEI-93-TR-24, ESC-TR-93-177, **Software Engineering Institute**, Carnegie Mellon University, Feb. 1993. Disponível: http://www.sei.cmu.edu/pub/documents/93.reports/pdf/tr24.93.pdf . Acesso: fev 2006.

POYNTON, B., LANERGAN, R., **Reusable Code: The Application Development Technique of the Future**, Proocedings of the IBM SHARE/GUIDE Software symposium, IBM, Monterey, California. out. 1979.

POULIN, J. S. **Measuring software reuse: principles, practices and economic models**, Massachusetts: Addison-Wesley, Jan. 1996. 224 p.

PRESSMAN, R. **Software Engineering: A Practioner's Approach**. 5a. ed., United Kingdom:McGraw-Hill, 2000.

PRIETO-DIAZ, R., A software Classification Scheme for Reusability. **IEEE Software,** 4(1): 6-16, January 1987.

PRIETO-DÍAZ, R. Status Report: Software Reusability. **IEEE Software**, v.10, n. 3,p. 61-66, May 1993.

PRIETO-DÍAZ, R.; FREEMAN, P. Classifying Software for Reusability. **IEEE Software**, v. 4, n. 1, p. 6-16, January 1987.

ROCHA, A. R. C., MALDONADO, J. C., WERBER, K. C. **Qualidade de software: teoria e prática.** 1ª Edição, São Paulo: Prentice-Hall, 2001. 303 p.

ROSSI, A. C. **Representação do Componente de Software na FARCSoft: Ferramenta de Apoio à Reutilização de Componentes de Software**. Dissertação (Mestrado em Engenharia de Software) . Escola Politécnica da Universidade de São Paulo, 2004.

SAMETINGER, J. **Reuse documentation and documentation reuse. In: Proceedings of TOOLS**, Europe 96, Paris, February 1996. 12 pp.

SANCHES, Maurício Grunh. **Um Estudo sobre os Riscos Inerentes à Implantação do Reuso de Componentes no Processo de Desenvolvimento de Software**. Dissertação (Mestrado Profissional). Universidade Estadual de Campinas, Instituto de Computação, 2005.

SEI - SOFTWARE ENGINEERING INSTITUTE. **CMMI: Capability maturity model integration**, v. 1.1 – CMU/SEI-2002-TR-012, Software Engineering Institute, Pittsburgh, 2002. p. 465-480.

SOMMERVILLE, I. **Software engineering.** 6th Edition, Massachusetts: Addison Wesley, Aug. 2000. 693 p.

SILVA, Mônica F. da; WERNER, Cláudia M. L. **Packaging Reusable Components Using Patterns and Hypermedia**. 4th International Conference on Software Reuse (ICSR, 1996).

SILVA, Ricardo Pereira e. **Suporte ao desenvolvimento e uso de frameworks e componentes**. Tese de Doutorado, PPGC da UFRGS, 2000.

SZYPERSKI, C. **Component Software: Beyond Object-Oriented Programming**. Addison-Wesley / ACM Press, 1998. 411 p.

THE ARMY STRATEGIC SOFTWARE REUSE PLAN. Disponível:http://www.belvoir.army.mil/odisc4/assrptoc.htm [Capturado em 12 ago. 1992]. Não mais disponível.

TRACZ, Will. **Confessions of a Used Program Salesman**. Addison-Wesley 1995

VITHARANA, P.; ZAHEDI, F. M.; JAIN, H. Knowledge-Based Repository Scheme for Storing and Retrieving Business Components: A Theoretical Design and Empirical Analysis. **IEEE Transactions on Software Engineering**, v. 29, n.7, July 2003.

WIEGERS, K. E. Improving quality through software inspections. **Software Development**, April 1995. p. 55-64.

7 APÊNDICES

7.1 APÊNDICE 1: ARTEFATOS GERADOS NO ESTUDO DE CASO

7.1.1 CERTIFICAÇÃO

TAB 4.2 - Atividade Aprovar *ASSET* – passo 2 – Artefato Necessidades do Cliente e Glossário

***Checklist*: Necessidades do Cliente e Glossário**

Artefato Verificado: CTNC001/03

As questões devem ser respondidas após um breve contato com o documento. A coluna P (Previsão) contém a resposta correta desejada e na coluna C (Confirmação) o revisor deve fornecer a resposta adequada: S (Sim) / N (Não) / NA (Não se aplica). Caso a resposta da Coluna C diferencie da coluna P, exceto quando não se aplica (NA), a coluna I (Identificação) deve ser preenchida com número seqüencial para identificação do erro detectado. Todos os erros detectados devem ser identificados e comentados na parte de Não-Conformidades.

1. Aspecto: Apresentação do Documento

Este aspecto deve analisar as questões gerais de apresentação do documento.	P	C	I
1.1 O documento está de acordo com o *template* padrão?	s	n	1
1.2 O documento teve ortografia e gramática checada?	s	s	
1.3 O documento está livre de erros de *layout*?	s	s	
1.4 O glossário dispõe de todas as definições e explicações que o revisor necessita para o completo entendimento das necessidades e características desejadas?	s	s	
1.5 Os números das linhas do texto do documento estão impressos para facilitar a referência de localização específica durante a verificação?	s	s	

Total de itens do aspecto verificados:	5
Total de não-conformidades:	1

2. Aspecto: Qualidade

Este aspecto descreve a qualidade que as necessidades, isto é, as funcionalidades e as características desejadas, devem apresentar no documento.	P	C	I
2.1 As necessidades apresentam nível de detalhamento adequado?	s	s	
2.2 Todas as características de desempenho estão corretamente especificadas?	s	s	
2.3 Toda as considerações de segurança estão corretamente especificadas?	s	s	
2.4 Outras características desejadas de qualidade pertinentes estão explicitamente documentadas e quantificadas de acordo com o objetivo especificado?	s	s	

Total de itens do aspecto verificados:	4
Total de não-conformidades:	0

3. Aspecto: Completude

Este aspecto descreve o que o documento deve apresentar com relação à consistência das necessidades e a completude destes documentos.	P	C	I
3.1 As funcionalidades e características desejadas provêem uma base adequada para análise de requisitos da engenharia de domínio?	s	s	
3.2 O documento Necessidades do Cliente inclui todas as funcionalidades do sistema?	s	s	
3.3 O documento Necessidades do Cliente inclui todas as características desejadas que o sistema precisa ter?	s	s	
3.4 Todas as referências internas das necessidades estão corretas?	s	s	
3.5 As necessidades fornecem uma base adequada para o projeto?	s	s	
3.6 A prioridade de implementação das necessidades está definida?	s	s	
3.7 Todo *hardware*, *software* e interfaces de comunicação estão definidos?	s	n	2
3.8 Os algoritmos intrínsecos para as necessidades funcionais estão definidos?	s	s	
3.9 A necessidade inclui todo o conhecimento do cliente ou necessidades de sistema?	s	s	
3.10 O comportamento esperado está documentado para todas as condições de erro?	s	s	
3.11 Todas as funções críticas de tempo estão identificadas e seus critérios de tempo especificados?	s	s	

Total de itens do aspecto verificados:	11
Total de não-conformidades:	1

Não - Conformidades Detectadas
Artefato Verificado: CTNC001/03

Identificar e comentar brevemente os defeitos detectados. O número de identificação (Id) deve ser o mesmo fornecido na coluna I (Identificação) do *checklist*.

Id	Comentários
1	Os itens 5.8, 5.9, 5.11, 6.5.5 e 6.5.6 estão em desacordo em Necessidades do Cliente.
2	Falta definição dos tipos de monitores a serem usados.

Identificação da Verificação

Projeto: Controlador Tálico — Id. do artefato: CTNC001/03
Verificador: — Data da Verificação:

Nível de conhecimento do domínio: [x] Baixo [ ] Médio [ ] Alto

Dados da Verificação

Total de Tempo
- Preparação: 0 hr
- Realização da verificação: 1 hr
- Dispendido: 1 hr

Quantidade
- Páginas verificadas: 13

Análise da Verificação

a1)	Total de itens do checklist:	33
a2)	Total de itens verificados:	32
	Qtd de itens atendidos:	30
	Qtd de itens NÃO atendidos:	2
a3)	Total de itens NÃO aplicáveis:	1
	Total de Defeito c Detectados:	2

Resultado da Verificação

[ ] Aceito
- [ ] Sem restrições
- [ ] Com restrições
- Data da próxima verificação: / /

[x] Não Aceito
- Data da próxima verificação: 10 / 01 / 2004
- [x] Refazer para nova verificação
- [ ] Verificação não completada

Problemas

Listar os problemas detectados que não são não-conformidades (por ex., o artefato a ser verificado não foi produzido) e as devidas ações decorrentes para resolver o problema (Gerente de Recursos Humanos e autorizou por escrito a dispensa da elaboração do artefato que se encontra em anexo).

Problemas	Ações Decorrentes

O Questionário de Sugestões para Melhoria da Garantia da Qualidade deve ser preenchido e entregue junto a este Relatório. Sua CONTRIBUIÇÃO é fundamental para o contínuo progresso da melhoria da NOSSA QUALIDADE!

Data: 09 / 12 / 2004 — Assinatura do Verificador:

Tabela de Discrepância para Leitura Vertical

Projeto: CT001/03 **Tipo Leitura Vertical (TLV):** TLV1
Data da revisão: 10/3/2003 **Início da revisão:** 9 **hr**

Nome e tipo dos documentos inspecionados:
- Doc1: CU010/03 Diagrama Casos de Uso e Cenários
- Doc2: DC010/03 Diagrama de Classes

ND - Número seqüencial da discrepância

TC - Tipo de Conceito

ATO-ator	COM-comportamento	HER-herança	PAP-papel
ATR-atributo	CON-condição	MEN-mensagem	REL-relacionamento
CAR-cardinalidade	DAD-dado	OBJ-objeto/classe	RES-restrição

OUT-outros tipos (identificar em comentários)

TD - Tipo de Discrepância
1. Omissão de funcionalidade ou conceito.
2. Projeto incorreto em relação às necessidades.
3. Informação de projeto não está mencionada nas necessidades.
4. Implementação das necessidades está ambígua ou não complemente especificada.
5. Outro tipo de problema de projeto (explique).

Sev - Severidade
- A Não é sério, mas precisa verificar este documento.
- B Esta discrepância invalida esta parte do documento. Verifique ambos documentos.
- C É sério. Não é possível continuar a leitura deste documento. Ele deve ser reorganizado.

Todas as discrepâncias com severidade C e do tipo 5 devem ser explicadas na parte de Detalhamento, assim como outras discrepâncias consideradas importantes.

ND	TC	Nome	TD	Sev	Comentários
1	ATR	Extrair Dados / lextração	2	B	Método não confere com a interface.
2	ATR	Extrair Dados	2	B	Vento não definido.
3	ATR	Atualizar dados da plataforma	1	A	Faltam dados (atributos) na interface.

Caso seja necessário utilize a tabela adicional e / ou detalhamento adicional.

Témino da revisão: 10 **hr.** (Só preencher se NÃO for utilizar a tabela adicional)

Tabela de Discrepância para Leitura Horizontal

Projeto: CT001/03 **Tipo Leitura Horizontal (TLH):** TLH1

Data da revisão: 10/3/2003 **Início da revisão:** 9 **hr**

Nome e tipo dos documentos inspecionados:

Doc1: DC010/03 Diagrama de Classe

Doc2: DS010/03 Diagrama de Seqüência

ND - Número seqüencial da discrepância

TC - Tipo de Conceito

ATO-ator COM-comportamento HER-herança PAP-papel

ATR-atributo CON-condição MEN-mensagem REL-relacionamento

CAR-cardinalidade DAD-dado OBJ-objeto/classe RES-restrição

OUT-outros tipos (identificar em comentários)

TD - Tipo de Discrepância

1 Presente no Doc1 e ausente no Doc2.

2 Presente no Doc2 e ausente no Doc1.

3 Presente em ambos documentos, mas inconsistente ou ambíguo.

4 Presente em ambos documentos, mas usando notação ou representação incorreta.

5 Presente em ambos documentos, mas representa informação estranha.

6 Ausente em ambos documentos.

Sev - Severidade

A Não é sério, mas precisa verificar este documento.

B Esta discrepância invalida esta parte do documento. Verifique ambos documentos.

C É sério. Não é possível continuar a leitura deste documento. Ele deve ser reorganizado.

Todas as discrepâncias com severidade C e do tipo 5 ou 6 devem ser explicadas na parte de Detalhamento, assim como outras discrepâncias consideradas importantes.

ND	TC	Nome	TD	Sev	Comentários
1	HER	DOC 2	2	A	Não se encontra a herança.
2	OBJ	DOC 2	3	A	Classe operador = usuário? Nome não OK.
3	HER	DOC 2	2	A	Herança de operador.
4	OUT	DOC 1	2	C	Modelo de Contexto sem descrição textual.

Caso seja necessário utilize a tabela adicional e / ou detalhamento adicional.

Témino da revisão: 10 **hr.** (Só preencher se NÃO for utilizar a tabela adicional)

Detalhamento de Discrepância para Leitura Horizontal

Projeto: CT001/03 **TLH: TLH1**

Data-revisão: 10/3/2003

Detalhar as discrepâncias sérias e dos tipos 5 e 6 conforme a estrutura a seguir:

ND	Descrição Detalhada
4	Sem descrição textual do Modelo Contexto dificulta o entendimento.

7.1.2 CLASSIFICAÇÃO

TAB 4.3 - Atividade Armazenar *ASSET* – passo 2

Nome do *Asset*:	Análise de Domínio do Controlador Tático
Domínio:	Sistema Tático
Aplicação:	Controle e Defesa do Mar
Tipo de *Asset*:	Domínio
Atividade:	Análise de Domínio
Tipo de Componente:	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse *Asset* apresenta os artefatos gerados na Análise de Domínio de um Controlador Tático. A definição do domínio baseou-se na documentação utilizada no Terminal Tático Inteligente (TTI). A finalidade desse *asset* é a representação do conhecimento da análise de Domínio do Controlador Tático

==========

Nome do *Asset*:	Arquitetura de Domínio do Controlador Tático
Domínio:	Sistema Tático
Aplicação:	Controle e Defesa do Mar
Tipo de Assets:	Arquitetura de Domínio
Atividade:	Projeto de Domínio
Tipo de Componente:	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse *Asset* apresenta os artefatos gerados na especificação da Arquitetura de Domínio. Foi selecionado o estilo arquitetural baseado em camadas pelas facilidades de reuso, a utilização do padrão arquitetural MVC e o padrão J2EE: controle (*front controller*) e despachante de visões (*dispatcher view*).

Nome do *Asset*:	Componente de Domínio do Controlador Tático
Domínio:	Sistema Tático
Aplicação:	Controle e Defesa do Mar
Tipo de Assets:	Componentes de Domínio
Atividade:	Projeto de Domínio
Tipo de Componente:	-
Comando	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse *asset* apresenta os Componentes de Domínio do Controlador Tático. A partir da Arquitetura de Domínio, refinaram-se os componentes da camada de aplicação e definiram-se as suas arquiteturas internas e pontos em comum. A forma de composição usada foi a funcional. A finalidade desse *asset* é apresentar os componentes da camada de aplicação e a definição de suas arquiteturas internas e pontos em comum

Nome do *Asset*:	Análise de Domínio do Treinador de Ataque
Domínio:	Simulador
Aplicação:	Treinamento de guerra A/S
Tipo de *Asset*:	Domínio
Atividade:	Análise de Domínio
Tipo de Componente	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse *Asset* apresenta os artefatos gerados na atividade de Análise de domínio do Treinador

	de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)

Nome do *Asset*:	Arquitetura de Domínio do Treinador de Ataque
Domínio:	Simulador
Aplicação:	Treinamento de guerra A/S
Tipo de *Asset*:	Arquitetura de Domínio
Atividade:	Projeto de Domínio
Tipo de Componente	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse Asset de arquitetura de domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)

Nome do *Asset*:	Componente de Domínio do Treinador de Ataque
Domínio:	Simulador
Aplicação:	Treinamento de guerra A/S
Tipo de *Asset*:	Componente de Domínio
Atividade:	Projeto de Domínio
Tipo de Componente	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse Asset de Componente de Domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)

Nome do *Asset*:	Componente do Treinador de Ataque
Domínio:	Simulador
Aplicação:	Treinamento de guerra A/S
Tipo de *Asset*:	Componente de Domínio
Atividade:	Construção de Software
Tipo de Componente	Classe
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse Asset de componente apresenta os artefatos gerados durante a atividade de construção de software do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o

	treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)

Nome do *Asset*:	Análise de Domínio do Sistema de Apoio à Decisão Logística
Domínio:	Sistema de Apoio à Decisão
Aplicação:	Logística de Meios Navais
Tipo de *Asset*:	Domínio
Atividade:	Análise de Domínio
Tipo de Componente	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse *Asset* apresenta os artefatos gerados na atividade de Análise de Domínio do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições

Nome do *Asset*:	Arquitetura de Domínio do Sistema de Apoio à Decisão Logística
Domínio:	Sistema de Apoio à Decisão
Aplicação:	Logística de Meios Navais
Tipo de *Asset*:	Arquitetura de Domínio
Atividade:	Projeto de Domínio
Tipo de Componente	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse *Asset* apresenta os artefatos gerados na atividade de Projeto de Domínio do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições

Nome do *Asset*:	Componente de Domínio do Sistema de Apoio à Decisão Logística
Domínio:	Sistema de Apoio à Decisão
Aplicação:	Logística de Meios Navais
Tipo de *Asset*:	Componente de Domínio
Atividade:	Projeto de Domínio

Tipo de Componente	Classe
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Esse Asset de componente de domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições

Nome do *Asset*:	Componente do Sistema de Apoio à Decisão Logística
Domínio:	Sistema de Apoio à Decisão
Aplicação:	Logística de Meios Navais
Tipo de *Asset*:	Componente
Atividade:	Construção de Software
Tipo de Componente	-
Comando:	Marinha
Tipo de Sistema:	Sistema Científico
Descrição:	Descrição - Esse Asset de componente apresenta os artefatos gerados durante a atividade de construção de software do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições

7.1.3 CATÁLOGO

TAB 4.3 - Atividade Armazenar *ASSET* – passo 3

TAB 4.11 - Atividade Atualizar *ASSET* – passo 2

Catálogo dos *Assets*:

Profile - AnD01:

- **Nome do *asset*:** Análise de Domínio do Controlador Tático
- **Cod Ident:** AnD01
- **Autor:** CASOP
- **Data:** Novembro/2003
- **Descrição:** Esse *Asset* apresenta os artefatos gerados na Análise de Domínio de um Controlador Tático. A definição do domínio baseou-se na documentação utilizada no Terminal Tático Inteligente (TTI)
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução - AnD01:

- Necessidades do Cliente - AnD01-01;
- Glossário - AnD01-02;
- Modelo de Contexto do Domínio do Controlador Tático - AnD01-03;
- Modelo Conceitual do Controlador Tático - – AnD01-04;
- Diagrama de Casos de Uso de Domínio do Controle Tático - AnD01-05;
- Cenário Inserir Dados de Ambiente - AnD01—06;
- Cenário Abrir Acompanhamento – AnD01-07;
- Cenário Atualizar Dados da Plataforma – AnD01-08;
- Cenário Extrair Dados – AnD01-09;
- Cenário Definir Configuração – AnD01-10;
- Cenário Cancelar Acompanhamento – AnD01-11;
- Cenário Alarme de Colisão – AnD01-12; - AnD01-12; e
- Modelo de Características Parcial do Controle Tático - AnD01-13.

Utilização - AnD01:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* pode ser utilizado na etapa de representação do conhecimento de um sistema tático;
    - **Instalação:** esse asset não requer instalação;

- **Adaptação:** livre;
- **Implementação:** foi utilizada a ferramenta gráfica Microsoft Visio Professional 2002;
- **Suporte:** qualquer dúvida entrar em contato com CASOP, Marinha do Brasil.

▪ Características Administrativas:

- **Histórico:** versão: 1.0 – responsável: CASOP;
- **Restrições comerciais e legais:** esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;

▪ Características de Avaliação:

- **Restrições técnicas:** uso do sistema operacional Windows XP;
- **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

  Total de itens do *checklist*: 33;

  Total de itens verificados: 32;

  Quantidade de itens atendidos: 30;

  Quantidade de itens não atendidos: 2;

  Total de itens não aplicáveis: 1;

  Total de defeitos detectados: 0.
- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** *asset* de domínio do TTI (Terminal Tático Inteligente);
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;
- **Custo recomendado: -** não há.

▪ Características Complementares:

- **Indexação:** não há;
- **Referências:** documentações do Sistema TTI, Terminal Tático Inteligente (especificação operativa, manual do usuário e outros); e o livro UML Guia do Usuário (Booch – Rumbaugh – Jacobson).

Assets Relacionados - AnD01:

Arquitetura de Domínio do Controlador Tático – ArD01;

Componente de Domínio de Controlador Tático – CoD01.

Diferenças_Assets – AnD01:

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

==========

Profile ArD01:

- **Nome do *asset*:** Arquitetura de Domínio do Controlador Tático – ArD01
- **Autor:** CASOP
- **Data:** Novembro/2003
- **Descrição:** Esse *Asset* apresenta os artefatos gerados na especificação da Arquitetura de Domínio. Foi selecionado o estilo arquitetural baseado em camadas pelas facilidades de reuso, a utilização do padrão arquitetural MVC e o padrão J2EE: controle (*front controller*) e despachante de visões (*dispatcher view*).
- **Versão:** 1.0
- **Acesso:** C (Sistema Científico)

Solução – ArD01:

- Modelo de Tipo do Controle Tático - ArD01-01;
- Interface de Camada de Negócios e Tipos de Dados do Controle Tático - ArD01-02;
- Interface da Camada de Aplicação do Controle Tático - ArD01-03
- Camada de Apresentação da Arquitetura de Domínio do Controle Tático - ArD01-04;
- Arquitetura de Domínio Inicial do Controle Tático - ArD01-05;

- Diagrama de Seqüência: Camada Apresentação do Controle Tático - ArD01-06;
- Diagrama de Seqüência: Camada Aplicação – Abrir Acompanhamento - ArD01-07;
- Diagrama de Seqüência: Camada Aplicação – Configuração - ArD01-08;
- Diagrama de Seqüência: Camada Aplicação – Cancela Acompanhamento - ArD01-09;
- Diagrama de Seqüência: Camada Aplicação – Atualiza Acompanhamento - ArD01-10;
- Diagrama de Seqüência: Camada Aplicação – Extração Plataforma Primária - ArD01-11;
- Diagrama de Seqüência: Camada Aplicação – Extração Posição Plataforma Primária - ArD01-12; e
- Diagrama de Seqüência: Camada Aplicação – Extração Acompanhamentos - ArD01-13.

Utilização – ArD01:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* poderá ser utilizado na etapa de especificação da arquitetura de domínio do controlador tático;
    - **Instalação:** esse *asset* não requer instalação;
    - **Adaptação:** livre;
    - **Implementação:** foi utilizada a ferramenta gráfica Microsoft Visio Professional 2002;
    - **Suporte**: qualquer dúvida entrar em contato com CASOP, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: CASOP – Marinha do Brasil;
    - **Restrições comerciais e legais:** esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;
- Características de Avaliação:
    - **Restrições técnicas:** uso do sistema operacional Windows XP;

- **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

  Total de itens do *checklist*: 56;

  Total de itens verificados: 56;

  Quantidade de itens atendidos: 50;

  Quantidade de itens não atendidos: 5;

  Total de itens não aplicáveis: 3;

  Total de defeitos detectados: 0.
- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** *asset* de arquitetura de domínio do TTI (Terminal Tático Inteligente);
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* deve ser utilizado com o A*sset* de Componentes de Domínio do Controlador Tático – CoD01;
- **Custo recomendado**: - não há

Assets Relacionados – ArD01:

Análise de Domínio do Controlador Tático – AnD01; e

Componente de Domínio de Controlador Tático – CoD01.

Diferenças_Assets – ArD01 :

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características de Complementares	-

Profile – CoD01:

- **Nome do *asset*:** Componentes de Domínio do Controlador Tático
- **Cod. Ident:** CoD01;
- **Autor:** CASOP
- **Data:** Novembro/2003
- **Descrição:** Esse *asset* apresenta os Componentes de Domínio do Controlador Tático. A partir da Arquitetura de Domínio, refinou-se os componentes da camada de aplicação e definiu-se as suas arquiteturas internas e pontos em comum. A forma de composição usada foi a funcional. A finalidade desse asset é apresentar os componentes da camada de aplicação e a definição de suas arquiteturas internas e pontos em comum;
- **Versão:** 1.0
- **Acesso:** C (Sistema Científico)

Solução - CoD01:

- Camada de Aplicação do Controle Tático Refinada - CoD01-01;
- Componente: InsDadosAmb com suas dependências - CoD01-02;
- Componente: CalculaCorrente com suas dependências CoD01-03;
- Componente: CalculaVentoReal com suas dependências - CoD01-04;
- Componente: SomaVetor – CoD01-05;
- Componente: Extração com suas dependências - CoD01-06;
- Componente: Calcula Posição Estimada com suas dependências - CoD01-07;
- Componente: Alarme de Colisão - CoD01-08;
- Componente: Conversão - CoD01-09;
- Componente: Atualiza Plataforma com suas dependências - CoD01-10;
- Componente: Configuração com suas dependências - CoD01-11;
- Componente: Calcula PMA com suas dependências - CoD01-12;
- Componente: Calcula Posição Estimada com suas dependências - CoD01-13;
- Componente: Calcula Rumo e Velocidade com suas dependências - CoD01-14; e
- Componente: Calcula Rumo para Passagem Segura - CoD01-15.

Utilização - CoD01:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* pode ser utilizado na etapa de especificação dos componentes de domínio do controlador tático;
    - **Instalação:** esse *asset* não requer instalação;
    - **Adaptação:** livre;
    - **Implementação: :** foi utilizada a ferramenta gráfica Microsoft Visio Professional 2002;
    - **Suporte**: qualquer dúvida entrar em contato com CASOP, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: CASOP - Marinha do Brasil;
    - **Restrições comerciais e legais:** esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;
- Características de Avaliação:
    - **Restrições técnicas:** uso do sistema operacional Windows XP;
    - **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

      Total de itens do *checklist*: 38;

      Total de itens verificados: 35;

      Quantidade de itens atendidos: 33;

      Quantidade de itens não atendidos: 2;

      Total de itens não aplicáveis: 0;
    - Total de defeitos detectados: 0.
    - **Performance:** micro-computador pentium IV (configuração mínima);
    - **Alternativas:** *asset* de arquitetura de domínio do TTI (Terminal Tático Inteligente);
    - **Problemas:** não há registro de problemas;
    - **Interdependências:** o *asset* deverá ser utilizado com o *asset* da Arquitetura de Domínio do Controlador Tático – ArD01;
    - **Custo recomendado: -** não;

Assets Relacionados - CoD01:

Análise de Domínio de Controlador Tático - AnD01; e

Arquitetura de Domínio do Controlador Tático ArD01.

Diferenças_Assets - CoD01 :

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características de Complementares	-

Profile - AnD02:

- **Nome do *asset*:** Análise de Domínio do Treinador de Ataque
- **Cod Ident:** AnD02
- **Autor:** IPQM
- **Data:** 2002
- **Descrição:** Esse Asset apresenta os artefatos gerados na atividade de Análise de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução - AnD02:

- Especificação operativa – AnD02-01;
- Glossário - AnD02-02;
- Diagrama de casos de uso - AnD02-03;
- Modelo conceitual dos simuladores - AnD02-04;
- Modelo conceitual dos sensores - AnD02-05;
- Modelo conceitual dos consoles - AnD02-06;

- Modelo conceitual dos SistemasInformaçoesTaticas - AnD02-07;
- Modelo conceitual dos SistemaDeArmas - AnD02-08;
- Modelo conceitual das Armas - AnD02-09.

Utilização - AnD02:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* deve ser utilizado na etapa de representação do conhecimento de um sistema tático;
    - **Instalação:** esse *asse*t pode ser adaptado para a construção de um novo sistema utilizando as ferramentas descritas no "item instalação";
    - **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de análise de domínio no desenvolvimento de um Treinador de Ataque;
    - **Implementação:** foi utilizada a ferramenta Microsoft Word e Rational Rose;
    - **Suporte:** qualquer dúvida entrar em contato com IPQM, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: IPQM;
    - **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados
- Características de Avaliação:
    - **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
    - **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

        Total de itens do *checklist*: 33;

        Total de itens verificados: 32;

        Quantidade de itens atendidos: 30;

        Quantidade de itens não atendidos: 2;

        Total de itens não aplicáveis: 1;

        Total de defeitos detectados: 0.

- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;
- **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso.

▪ Características Complementares:

- **Indexação:** não há índice para *assets* complexos
- **Referências:** documentações do Treinador de Ataque QFA - 3, QFA 4 AN/UQS T1A;

Assets Relacionados - AnD02:

Arquitetura de Domínio do Treinador de Ataque - ArD02;

Componente de Domínio de Treinador de Ataque - CoD02; e

Componente do Treinador de Ataque – Co02.

Diferenças_Assets - AnD02 :

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

Profile - ArD02:

▪ **Nome do *asset*:** Arquitetura de Domínio do Treinador de Ataque

▪ **Cod Ident:** ArD02

- **Autor:** IPQM
- **Data:** 2002
- **Descrição:** Esse Asset de arquitetura de domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução - ArD02:

- Diagrama de pacote da camada de negócio e tipo de dados do AplicIhmInstrutor – ArD02-01;
- Diagrama de pacote da camada de negócio e tipo de dados do AplicIhmVeiculo – ArD02-02;
- Diagrama de pacote da camada de negócio e tipo de dados do AplicVeiculo – ArD02-03;
- Diagrama de pacote da camada de negócio e tipo de dados do AplicCCM – ArD02-04;
- Diagrama de pacote da camada de negócio e tipo de dados do Arma – ArD02-05;
- Diagrama de distribuição Instrutor Master - ArD02-06;
- Diagrama de distribuição Instrutor Slave 01 - ArD02-07;
- Diagrama de distribuição Instrutor Slave 02 - ArD02-08;
- Diagrama de distribuição Instrutor Slave 03 - ArD02-09;
- Diagrama de distribuição Aluno Manobra - ArD02-10;
- Diagrama de distribuição Aluno CIC 01 - ArD02-11;
- Diagrama de distribuição Aluno CIC 02 - ArD02-12;
- Diagrama de distribuição Aluno CIC 03 - ArD02-13.

Utilização - ArD02:

- Características de Reutilização:
  - **Usabilidade:** esse *asset* deve ser utilizado na etapa de especificação da arquitetura de domínio de um Treinador de Ataque;
  - **Instalação:** esse *asse*t pode ser adaptado para a construção de um novo sistema utilizando as ferramentas descritas no “item instalação”;

- **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de projeto de domínio no desenvolvimento de um Treinador de Ataque;
- **Implementação:** foi utilizada a ferramenta Microsoft Word e Rational Rose;
- **Suporte:** qualquer dúvida entrar em contato com IPQM, Marinha do Brasil.

▪ Características Administrativas:

- **Histórico:** versão: 1.0 – responsável: IPQM;
- **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados

▪ Características de Avaliação:

- **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
- **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

  Total de itens do *checklist*: 27;
  Total de itens verificados: 27;
  Quantidade de itens atendidos: 25;
  Quantidade de itens não atendidos: 2;
  Total de itens não aplicáveis: 0;
  Total de defeitos detectados: 0.
- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;
- **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso.

- Características Complementares:
    - **Indexação:** não há índice para *assets* complexos
    - **Referências:** documentações do Treinador de Ataque QFA - 3, QFA 4 AN/UQS T1A;

Assets Relacionados - ArD02:

Análise de Domínio do Treinador de Ataque - AnD02;

Componente de Domínio de Treinador de Ataque - CoD02; e

Componente do Treinador de Ataque – Co02.

Diferenças_Assets - ArD02 :

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

Profile – CoD02:

- **Nome do *asset*:** Componente de Domínio do Treinador de Ataque
- **Cod Ident:** CoD02
- **Autor:** IPQM
- **Data:** 2002
- **Descrição:** Esse Asset de Componente de Domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução - CoD02:

- Diagrama de pacote dos componentes - CoD02-01;
- Diagrama do componente armas - CoD02-02;
- Diagrama do componente foguete - CoD02-03;
- Diagrama do componente torpedo AS - CoD02-04;
- Diagrama do componente bomba_prof - CoD02-05;
- Diagrama do componente lançador - CoD02-06;
- Diagrama do componente Sistemas_armas - CoD02-07;
- Diagrama do componente projetil - CoD02-08;
- Diagrama do componente consoles - CoD02-09;
- Diagrama do componente veículo - CoD02-10;
- Diagrama do componente simulador - CoD02-11;
- Diagrama do componente ElemNav - CoD02-12;
- Diagrama do componente CadastroAlvo - CoD02-13;
- Diagrama do componente Alvo - CoD02-14;
- Diagrama do componente CadastroConsoleInst - CoD02-15;
- Diagrama do componente ControleDeDanos - CoD02-16;
- Diagrama do componente Servidor LC - CoD02-17;
- Diagrama do componente Reprodutor - CoD02-18;
- Diagrama do componente Gravador - CoD02-19;

Utilização - CoD02:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* deve ser utilizado na etapa de especificação do componente de domínio de um Treinador de Ataque;
    - **Instalação:** esse *asse*t pode ser adaptado para a construção de um novo sistema utilizando as ferramentas descritas no "item instalação";
    - **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de projeto de domínio no desenvolvimento de um Treinador de Ataque;
    - **Implementação:** foi utilizada a ferramenta Rational Rose;
    - **Suporte:** qualquer dúvida entrar em contato com IPQM, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: IPQM;

- **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados

- Características de Avaliação:
  - **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
  - **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

    Total de itens do *checklist*: 24;

    Total de itens verificados: 20;

    Quantidade de itens atendidos: 17;

    Quantidade de itens não atendidos: 3;

    Total de itens não aplicáveis: 4;

    Total de defeitos detectados: 0.
  - **Performance:** micro-computador pentium IV (configuração mínima);
  - **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
  - **Problemas:** não há registro de problemas;
  - **Interdependências:** o *asset* pode ser usado stand-alone;
  - **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso.

- Características Complementares:
  - **Indexação:** não há índice para *assets* complexos
  - **Referências:** documentações do Treinador de Ataque QFA - 3, QFA 4 AN/UQS T1A;

Assets Relacionados - CoD02:

Análise de Domínio do Treinador de Ataque - AnD02;

Componente de Domínio de Treinador de Ataque - CoD02; e

Componente do Treinador de Ataque – Co02.

Diferenças_Assets - CoD02 :

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

Profile – Co02:

- **Nome do *asset*:** Componente do Treinador de Ataque
- **Cod Ident:** Co02
- **Autor:** IPQM
- **Data:** 2002
- **Descrição:** Esse Asset de componente apresenta os artefatos gerados durante a atividade de construção de software do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução – Co02:

- Classe Agent – Co02-01;
- Classe BombaProf – Co02-02;
- Classe Foguete – Co02-03;
- Classe Lançador – Co02-04;
- Classe Projetil – Co02-05;
- Classe SistArmas – Co02-06;
- Classe Torpedo – Co02-07;
- Classe Torpedo_AS – Co02-08;
- Classe Alvo – Co02-09;
- Classe Cadastro_Alvo - Co02-10;

- Classe Cadastr_Console - Co02-11;
- Classe ControleDeDanos - Co02-12;
- Classe ElemNav - Co02-13;
- Classe ItAlvo - Co02-14;
- Classe Gravador – Co02-15;
- Classe Reprodutor – Co02-16;
- Classe ServidorLC - Co02-17 ;
- Classe Simulador – Co02-18;
- Classe ConsoleInstrutor – Co02-19;
- Classe AreasSens – Co02-20;
- Classe AreaVO – Co02-21;
- Classe BipVO – Co02-22;
- Classe Cenario Tatico – Co02-23;
- Classe Console – Co02-24;
- Classe Consultas - Co02-25;
- Classe LiCostaVO – Co02-26;
- Classe SolTatVO – Co02-27;
- Classe SimuladorSonar – Co02-28;
- Classe EquipamentosNavegação – Co02-29;
- Classe HardSensor – Co02-30;
- Classe PapelExercício – Co02-31;
- Classe Plataforma – Co02-32;
- Classe Propulsão – Co02-33;
- Classe SensoresTáticos – Co02-34;
- Classe SistemaNavegação – Co02-35;
- Classe Acompanhamento – Co02-36;
- Classe Area – Co02-37;
- Classe SolTat – Co02-38;
- Classe ZonaRadiação – Co02-39.

Utilização – Co02:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* deve ser utilizado na etapa de construção de software de um treinador de ataque;

- **Instalação:** esse *asse*t pode ser adaptado para a construção de um novo sistema utilizando as ferramentas descritas no "item instalação";
- **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de construção de componentes para o desenvolvimento de um Treinador de Ataque;
- **Implementação:** foi utilizada a linguagem de programação C^{++};
- **Suporte:** qualquer dúvida entrar em contato com IPQM, Marinha do Brasil.

▪ Características Administrativas:

- **Histórico:** versão: 1.0 – responsável: IPQM;
- **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados

▪ Características de Avaliação:

- **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
- **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

  Total de itens do *checklist*: 40;

  Total de itens verificados: 36;

  Quantidade de itens atendidos: 33;

  Quantidade de itens não atendidos: 3;

  Total de itens não aplicáveis: 4;

  Total de defeitos detectados: 0.
- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;

- **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso.

- Características Complementares:
  - **Indexação:** não há índice para *assets* complexos
  - **Referências:** documentações do Treinador de Ataque QFA - 3, QFA 4 AN/UQS T1A;

Assets Relacionados – Co02:

Análise de Domínio do Treinador de Ataque - AnD02;
Análise de Domínio de Treinador de Ataque - AnD02; e
Componente Domínio do Treinador de Ataque – CoD02.

Diferenças_Assets – Co02 :

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

Profile – AnD03:

- **Nome do *asset*:** Análise de Domínio do Sistema de Apoio à Decisão Logística;
- **Cod Ident:** AnD03
- **Autor:** CASNAV
- **Data:** 2005
- **Descrição:** Esse *Asset* apresenta os artefatos gerados na atividade de Análise de Domínio do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios

navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições

- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução – AnD03:

- Especificação de Requisitos – AnD03-01;
- Diagrama Principal de Casos de Uso - AnD03-02;
- Diagrama de Casos de Uso de Mensagem - AnD03-03;
- Diagrama de Casos de Uso de Meios Navais - AnD03-04;
- Diagrama de Casos de Uso de CONDEF - AnD03-05;
- Diagrama de Casos de Uso de Restrição de Meio Naval - AnD03-06;
- Diagrama de Casos de Uso de Administração do Sistema - AnD03-07;
- Cenário Consultar Mensagem- AnD03-08;
- Cenário Cadastrar Mensagem – AnD03-09;
- Cenário Corrigir Mensagem – AnD03-10;
- Cenário Excluir Mensagem – AnD03-11;
- Cenário Consultar Avaria do Meio Naval – AnD03-12;
- Cenário Evoluir Restrição de Meio Naval – AnD03-13;
- Cenário Cadastrar Restrição de Meio Naval – AnD03-14;
- Cenário Corrigir Restrição de Meio Naval – AnD03-15;
- Cenário Excluir Restrição de Meio Naval – AnD03-16;
- Cenário Atualizar CONDEF – AnD03-17;
- Cenário Corrigir CONDEF - AnD03-18;
- Cenário Excluir CONDEF – AnD03-19;
- Cenário Consultar Meio Naval – AnD03-20;
- Cenário Cadastrar Meio Naval – AnD03-21;
- Cenário Corrigir Meio Naval – AnD03-22;
- Cenário Excluir Meio Naval – AnD03-23;
- Cenário Obter Relatório de Meio Naval – AnD03-24;
- Cenário Consultar Serie – AnD03-25;
- Cenário Cadastrar Serie – AnD03-26;

- Cenário Corrigir Serie – AnD03-27;
- Cenário Excluir Serie – AnD03-28;
- Cenário Consultar Tipo de Restrição de Meio Naval – AnD03-30;
- Cenário Cadastrar Tipo de Restrição de Meio Naval – AnD03-31;
- Cenário Corrigir Tipo de Restrição de Meio Naval – AnD03-32;
- Cenário Excluir Tipo de Restrição de Meio Naval – AnD03-33;
- Cenário Consultar Usuário – AnD03-34;
- Cenário Cadastrar Usuário – AnD03-35;
- Cenário Corrigir Usuário – AnD03-36;
- Cenário Excluir Usuário – AnD03-37;
- Cenário Efetuar Login – AnD03-38;
- Modelo Conceitual – AnD03-39; e
- Modelo do Banco de Dados – AnD03-40.

Utilização – AnD03:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* deve ser utilizado na etapa de análise de domínio de um sistema de apoio à decisão logística;
    - **Instalação:** esse *asse*t pode ser adaptado para a construção de um novo sistema utilizando as ferramentas descritas no "item instalação";
    - **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de análise de domínio no desenvolvimento de um novo sistema de apoio à decisão logística;
    - **Implementação:** foi utilizada a ferramenta Microsoft Word e Enterprise Architect;
    - **Suporte:** qualquer dúvida entrar em contato com CASNAV, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: CASNAV;
    - **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;
- Características de Avaliação:

- **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
- **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

  Total de itens do *checklist*: 25;

  Total de itens verificados: 24;

  Quantidade de itens atendidos: 22;

  Quantidade de itens não atendidos: 2;

  Total de itens não aplicáveis: 1;

  Total de defeitos detectados: 0.
- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;
- **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso

▪ Características Complementares:

- **Indexação:** não há índice para *assets* complexos;
- **Referências:** documentos sobre Logística publicados pelo EMA.

Assets Relacionados – AnD03:

Arquitetura de Domínio do Sistema de Apoio à Decisão Logística - ArD03;

Componente de Domínio do Sistema de Apoio à Decisão Logística - CoD03; e

Componente do Sistema de Apoio à Decisão Logística - Co03.

Diferenças_Assets – AnD03:

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização:	-

Características de Avaliação	
Diferenças Utilização: **Características Complementares**	-

Profile – ArD03:

- **Nome do *asset*:** Arquitetura de Domínio do Sistema de Apoio à Decisão Logística;
- **Cod Ident:** ArD03
- **Autor:** CASNAV
- **Data:** 2005
- **Descrição:** Esse *Asset* apresenta os artefatos gerados na atividade de Projeto de Domínio do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução – ArD03:

- Diagrama de Seqüência Consultar Mensagem – ArD03-01;
- Diagrama de Seqüência Cadastrar Mensagem – ArD03-02;
- Diagrama de Seqüência Consultar Avaria de Meio Naval – ArD03-03;
- Diagrama de Seqüência Cadastrar Restrição de Meio Naval – ArD03-04;
- Diagrama de Seqüência Corrigir Restrição de Meio Naval – ArD03-05;
- Diagrama de Seqüência Excluir Restrição de Meio Naval - ArD03-06;
- Diagrama de Seqüência Atualizar CONDEF - ArD03-07;
- Diagrama de Seqüência Corrigir CONDEF - ArD03-08;
- Diagrama de Seqüência Excluir CONDEF- ArD03-09;
- Diagrama de Seqüência Consultar Meio Naval- ArD03-10;
- Diagrama de Seqüência Cadastrar Meio Naval - ArD03-11;
- Diagrama de Seqüência Corrigir Meio Naval - ArD03-12;
- Diagrama de Seqüência Excluir Meio Naval- ArD03-13;

- Diagrama de Seqüência Consultar Série – ArD03-14;
- Diagrama de Seqüência Cadastrar Série – ArD03-15;
- Diagrama de Seqüência Corrigir Série – ArD03-16;
- Diagrama de Seqüência Excluir Série – ArD03-17.

Utilização – ArD03:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* deve ser utilizado na etapa de especificação da arquitetura de domínio de um sistema de apoio à decisão logística;
    - **Instalação:** esse *asse*t pode ser adaptado para um novo sistema utilizando as ferramentas descritas no "item instalação";
    - **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de projeto de domínio no desenvolvimento de um novo sistema de apoio à decisão logística;
    - **Implementação:** foi utilizada a ferramenta Enterprise Architect;
    - **Suporte:** qualquer dúvida entrar em contato com CASNAV, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: CASNAV;
    - **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;
- Características de Avaliação:
    - **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
    - **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

        Total de itens do *checklist*: 33;

        Total de itens verificados: 28;

        Quantidade de itens atendidos: 28;

        Quantidade de itens não atendidos: 0;

        Total de itens não aplicáveis: 5;

        Total de defeitos detectados: 0.

- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;
- **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso

▪ Características Complementares:

- **Indexação:** não há índice para *assets* complexos;
- **Referências:** documentos sobre Logística publicados pelo EMA.

Assets Relacionados – ArD03:

Análise de Domínio do Sistema de Apoio à Decisão Logística - ArD03; Componente de Domínio do Sistema de Apoio à Decisão Logística - CoD03; e Componente do Sistema de Apoio à Decisão Logística - Co03.

Diferenças_Assets – ArD03:

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

Profile – CoD03:

▪ **Nome do *asset*:** Componente de Domínio do Sistema de Apoio à Decisão Logística;

▪ **Cod Ident:** CoD03

▪ **Autor:** CASNAV

- **Data:** 2005
- **Descrição:** Esse Asset de componente de domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução – CoD03:

- Modelo de Componente ProvideInterface1 - CoD03-01;
- Modelo de Componente RequiredeInterface- CoD03-02;
- Modelo de Componente DetalheMeioNaval- CoD03-03;
- Modelo de Componente ListarMeios- CoD03-04;
- Modelo de Componente Interface- CoD03-05;
- Modelo de Componente Avaria- CoD03-06

Utilização – CoD03:

- Características de Reutilização:
  - **Usabilidade:** esse *asset* deve ser utilizado na etapa de especificação do componente de domínio de um sistema de apoio à decisão logística;
  - **Instalação:** esse *asse*t pode ser adaptado para um novo sistema utilizando as ferramentas descritas no “item instalação”;
  - **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de projeto de domínio no desenvolvimento de um novo sistema de apoio à decisão logística;
  - **Implementação:** foi utilizada a ferramenta Enterprise Architect;
  - **Suporte:** qualquer dúvida entrar em contato com CASNAV, Marinha do Brasil.
- Características Administrativas:
  - **Histórico:** versão: 1.0 – responsável: CASNAV;

- **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;

- Características de Avaliação:
  - **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
  - **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

    Total de itens do *checklist*: 35;

    Total de itens verificados: 34;

    Quantidade de itens atendidos: 32;

    Quantidade de itens não atendidos: 2;

    Total de itens não aplicáveis: 1;

    Total de defeitos detectados: 0.
  - **Performance:** micro-computador pentium IV (configuração mínima);
  - **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
  - **Problemas:** não há registro de problemas;
  - **Interdependências:** o *asset* pode ser usado stand-alone;
  - **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso
- Características Complementares:
  - **Indexação:** não há índice para *assets* complexos;
  - **Referências:** documentos sobre Logística publicados pelo EMA.

Assets Relacionados – CoD03:

Análise de Domínio do Sistema de Apoio à Decisão Logística - ArD03;

Arquitetura de Domínio do Sistema de Apoio à Decisão Logística - CoD03; e

Componente do Sistema de Apoio à Decisão Logística - Co03.

Diferenças_Assets – CoD03:

Versão Anterior:	-
Diferenças Profile:	-

Diferenças Solução:	-
Diferenças Utilização: **Características de Reutilização**	-
Diferenças Utilização: **Características Administrativas**	-
Diferenças Utilização: **Características de Avaliação**	-
Diferenças Utilização: **Características Complementares**	-

Profile – Co03:

- **Nome do *asset*:** Componente do Sistema de Apoio à Decisão Logística;
- **Cod Ident:** Co03
- **Autor:** CASNAV
- **Data:** 2005
- **Descrição:** Descrição - Esse Asset de componente apresenta os artefatos gerados durante a atividade de construção de software do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições
- **Versão:** 1.0
- **Acesso:** SC (Sistema Científico)

Solução – Co03:

- Classe Condef – Co03-01;
- Classe Evolução – Co03-02;
- Classe ItemRestrição – Co03-03;
- Classe Meio – Co03-04;
- Classe MeioNaval – Co03-05;
- Classe Mensagem – Co03-06;
- Classe OM – Co03-07;
- Classe Organização – Co03-08;

- Classe Papel – Co03-09;
- Classe Permissão – Co03-10;
- Classe Restrição – Co03-11;
- Classe RMC – Co03-12;
- Classe Serie - Co03-13;
- Classe Tipo - Co03-14;
- Classe Transmissor – Co03-15;
- Classe UF – Co03-16;
- Classe Usuario - Co03-17;
- Classe UsuarioSessão – Co03-18;

Utilização – Co03:

- Características de Reutilização:
    - **Usabilidade:** esse *asset* deve ser utilizado na etapa de construção de software de um sistema de apoio à decisão logística;
    - **Instalação:** esse *asse*t pode ser adaptado para um novo sistema utilizando as ferramentas descritas no "item instalação";
    - **Adaptação:** esse *asset* pode ser adaptado para ser utilizado na atividade de projeto de domínio no desenvolvimento de um novo sistema de apoio à decisão logística;
    - **Implementação:** foi utilizada a ferramenta Enterprise Architect;
    - **Suporte:** qualquer dúvida entrar em contato com CASNAV, Marinha do Brasil.
- Características Administrativas:
    - **Histórico:** versão: 1.0 – responsável: CASNAV;
    - **Restrições comerciais e legais:** utilização restrita, por tratar-se de um sistema confidencial. esse *asset* só poderá ser utilizado em sistemas no contexto do Ministério da Defesa e seus Comandos Subordinados;
- Características de Avaliação:
    - **Restrições técnicas:** uso do sistema operacional Windows XP, NT ou 2000;
    - **Qualidade:** relatório de verificação - critérios de aprovação – via checklist:

Total de itens do *checklist*: 37;

Total de itens verificados: 35;

Quantidade de itens atendidos: 32;

Quantidade de itens não atendidos: 3;

Total de itens não aplicáveis: 2;

Total de defeitos detectados: 0.

- **Performance:** micro-computador pentium IV (configuração mínima);
- **Alternativas:** não há, até a presente data, relação de *assets* semelhantes;
- **Problemas:** não há registro de problemas;
- **Interdependências:** o *asset* pode ser usado stand-alone;
- **Custo recomendado:** não existe levantamento de custo para melhorar a performance, tornar o *asset* mais robusto e/ou estender o escopo do reuso

▪ Características Complementares:

- **Indexação:** não há índice para *assets* complexos;
- **Referências:** documentos sobre Logística publicados pelo EMA.

Assets Relacionados – Co03:

Análise de Domínio do Sistema de Apoio à Decisão Logística - ArD03;

Arquitetura de Domínio do Sistema de Apoio à Decisão Logística - CoD03; e

Componente de Domínio do Sistema de Apoio à Decisão Logística - Co03.

Diferenças_Assets – Co03:

Versão Anterior:	-
Diferenças Profile:	-
Diferenças Solução:	-
Diferenças Utilização: Características de Reutilização	-
Diferenças Utilização: Características Administrativas	-
Diferenças Utilização: Características de Avaliação	-
Diferenças Utilização: Características Complementares	-

7.1.4 CONTROLE DE VERSÕES

TAB.4.4 - Atividade Controlar Versões – passo 2

TAB 4.12 - Atividade Controlar Versões – passo 2

Nome do Asset:	**Análise de Domínio do Controlador Tático**
Cod Ident:	AnD01

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASOP

Nome do Usuário:	Alexandre Magno
Email:	alexmagno@ime.eb.br
Orgão_Usuario:	IME
Sistema:	Sistema Tático da Divisão de Tanques

Nome do *Asset:*	**Arquitetura de Domínio do Controlador Tático**
Cod Ident:	ArD01

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASOP

Nome do Usuário:	-
Email:	-
Orgão_Usuario:	-
Sistema:	-

Nome do Asset:	**Componentes de Domínio do Controlador Tático**
Cod Ident:	CoD01

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASOP

Nome do Usuário:	-
Email:	-
Orgão_Usuario:	-
Sistema:	-

Nome do Asset:	**Análise de Domínio do Treinador de Ataque**
Cod Ident:	AnD02

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	IPQM

Nome do Usuário:	-
Email:	-
Orgão_Usuario:	-
Sistema:	-

==========

Nome do Asset:	**Arquitetura de Domínio do Treinador de Ataque**
Cod Ident:	ArD03

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	IPQM

Nome do Usuário:	Luis Pontes
Email:	lpontes@casop.mar.mil.br
Orgão_Usuario:	CASOP
Sistema:	Treinador de Ataque do CIAMA

==========

Nome:	**Componente de Domínio do Treinador de Ataque**

Cod Ident:	CoD03

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	IPQM

Nome do Usuário:	Luis Pontes
Email:	lpontes@casop.mar.mil.br
Orgão_Usuario:	CASOP
Sistema:	Treinador de Ataque do CIAMA

Nome do Asset:	**Componente do Treinador de Ataque**
Cod Ident:	Co02

Versão/Versões Origem:	-
Versão:	1.0
Orgão	IPQM

Nome do Usuário:	
Email:	
Orgão_Usuario:	
Sistema:	

Nome do Asset:	**Análise de Domínio do Sistema de Apoio à Decisão Logística**
Cod Ident:	AnD03

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASNAV

Nome do Usuário:	-
Email:	-
Orgão_Usuario:	-
Sistema:	-

==========

Nome:	**Arquitetura de Domínio Sistema de Apoio à Decisão Logística**

Cod Ident:	ArD03

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASNAV

Nome do Usuário:	-
Email:	-
Orgão_Usuario:	-
Sistema:	-

Nome do Asset:	**Componente de Domínio Sistema de Apoio à Decisão Logística**
Cod Ident:	CoD04
Orgão:	CASNAV

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASNAV

Nome do Usuário:	-
Email:	-
Orgão_Usuario:	-
Sistema:	-

Nome:	**Componente do Sistema de Apoio à Decisão Logística**
Cod Ident:	Co04

Versão/Versões Origem:	-
Versão:	1.0
Orgão:	CASNAV

Nome do Usuário:	Renato Alves
Email:	Renatoalves@ita.aer.br
Orgão_Usuario:	ITA
Sistema:	Sistema de Apoio à Decisão Logistica de Meios Aeronáuticos

7.1.5 CADASTRO DE USUÁRIOS

TAB. 4.5 – Atividade Cadastrar Usuário – passo 3

Nome:	Luis Pontes
Login:	Lpontes
Senha:	55221144
Email:	Lpontes@casop.mar.mil.br
Orgão:	CASOP
Comando:	Marinha
Função:	Tecnologista Sênior III
Endereço:	Base Naval – Ilha de Mocanguê
Telefone:	(21)2187-1459
Cidade:	Rio de Janeiro
Estado:	Rio de Janeiro
Acesso	Sistemas Científicos (SC)

Nome:	Alexandre Magno
Login:	Alexmagno
Senha:	22334455
Email:	Alexmagno@ime.eb.br
Orgão:	Instituto Militar de Engenharia
Comando:	Exército
Função:	Major
Endereço:	Praia Vermelha – Urca
Telefone:	(21)2546-1111
Cidade:	Rio de Janeiro
Estado:	Rio de Janeiro
Acesso	Sistemas Científicos (SC)

Nome:	Renato Alves
Login:	Renatoalves
Senha:	55667788
Email:	Renatoalves@ita.aer.br
Orgão:	ITA
Comando:	Aeronáutica
Função:	Primeiro Tenente Aviador

Endereço:	CTA
Telefone:	(11)5264-6676
Cidade:	São José dos Campos
Estado:	São Paulo
Acesso	Sistemas Científicos (SC)

7.1.6 LISTA DE ASSETS CANDIDATOS

TAB. 4.6 – Atividade Buscar *Asset* – passo 2

Nome do Asset:	Análise de Domínio do Controlador Tático
Artefatos:	Necessidades do Cliente – AnD01-01 Glossário – AnD01-02 Modelo de Contexto do Domínio do Controlador Tático - AnD01-03 Modelo Conceitual do Controlador Tático - AnD01-04 Diagrama de Casos de Uso de Domínio do Controle Tático - AnD01-05 Cenário Inserir Dados de Ambiente - AnD01—06 Cenário Abrir Acompanhamento – AnD01-07 Cenário Atualizar Dados da Plataforma – AnD01-08; Cenário Extrair Dados – AnD01-09 Cenário Definir Configuração – AnD01-10 Cenário Cancelar Acompanhamento – AnD01-11 Cenário Alarme de Colisão – AnD01-12 Modelo de Características Parcial do Controle Tático – AnD01-13
Autor:	CASOP
Data:	Novembro/2003
Identificação:	Esse *Asset* apresenta os artefatos gerados na Análise de Domínio de um Controlador Tático. A definição do domínio baseou-se na documentação utilizada no Terminal Tático Inteligente (TTI). A finalidade desse asset é a representação do conhecimento da análise de domínio do Controlador Tático
Versão:	1.0
Acesso:	SC

Nome do Asset:	Componentes do Sistema de Apoio à Decisão Logística
Artefatos:	Classe Condef – Co04-01 Classe Evolução – Co04-02

	Classe ItemRestrição – Co04-03 Classe Meio – Co04-04 Classe MeioNaval – Co04-05 Classe Mensagem – Co04-06 Classe OM – Co04-07 Classe Organização – Co04-08 Classe Papel – Co04-09 Classe Permissão – Co04-10 Classe Restrição – Co04-11 Classe RMC – Co04-12 Classe Serie - Co04-13 Classe Tipo - Co04-14 Classe Transmissor – Co04-15 Classe UF – Co04-16 Classe Usuario - Co04-17 Classe UsuarioSessão – Co04-18
Autor:	CASNAV
Data:	Novembro/2003
Identificação:	Esse Asset de componente apresenta os artefatos gerados durante a atividade de construção de software do Sistema de Apoio à Decisão Logística. Esse sistema possibilita o acesso ao histórico de falhas de equipamento dos meios navais; a extração de dados a partir das mensagens (MICONDEF) que hoje trafegam no sistema; a disponibilização das informações acerca das restrições dos meios e a também tem a possibilidade de acompanhar a disponibilidade dos meios a partir das informações de CONDEF e das restrições
Versão:	1.0
Acesso:	SC

Nome do Asset:	Arquitetura de Domínio do Treinador de Ataque
Artefatos:	Diagrama de pacote da camada de negócio e tipo de dados do AplicIhmInstrutor – ArD03-01 Diagrama de pacote da camada de negócio e tipo de dados do AplicIhmVeiculo – ArD03-02 Diagrama de pacote da camada de negócio e tipo de dados do AplicVeiculo – ArD03-03 Diagrama de pacote da camada de negócio e tipo de dados do AplicCCM – ArD03-04 Diagrama de pacote da camada de negócio e tipo de dados do Arma – ArD03-05 Diagrama de distribuição Instrutor Master - ArD03-06 Diagrama de distribuição Instrutor Slave 01 - ArD03-07 Diagrama de distribuição Instrutor Slave 02 - ArD03-08

	Diagrama de distribuição Instrutor Slave 03 - ArD03-09 Diagrama de distribuição Aluno Manobra - ArD03-10 Diagrama de distribuição Aluno CIC 01 - ArD03-11 Diagrama de distribuição Aluno CIC 02 - ArD03-12 Diagrama de distribuição Aluno CIC 03 - ArD03-13
Autor:	IPQM
Data:	2002
Identificação:	Esse Asset de arquitetura de domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)
Versão:	1.0
Acesso:	SC

==========

Nome do Asset:	Componente de Domínio do Treinador de Ataque
Artefatos:	Diagrama de pacote dos componentes - CoD02-01 Diagrama do componente armas - CoD02-02 Diagrama do componente foguete - CoD02-03 Diagrama do componente torpedo AS - CoD02-04 Diagrama do componente bomba_prof - CoD02-05 Diagrama do componente lançador - CoD02-06 Diagrama do componente Sistemas_armas - CoD02-07 Diagrama do componente projetil - CoD02-08 Diagrama do componente consoles - CoD02-09 Diagrama do componente veículo - CoD02-10 Diagrama do componente simulador - CoD02-11 Diagrama do componente ElemNav - CoD02-12 Diagrama do componente CadastroAlvo - CoD02-13 Diagrama do componente Alvo - CoD02-14 Diagrama do componente CadastroConsoleInst - CoD02-15 Diagrama do componente ControleDeDanos - CoD02-16 Diagrama do componente Servidor LC - CoD02-17 Diagrama do componente Reprodutor - CoD02-18 Diagrama do componente Gravador - CoD02-19
Autor:	IPQM
Data:	2002
Identificação:	Esse Asset de Componente de Domínio apresenta os artefatos gerados durante a atividade de projeto de domínio do Treinador de Ataque Anti-Submarino. Esse sistema possibilita o treinamento das equipes de guerra A/S de navios sem a movimentação de unidades (navios)

Versão:	1.0
Acesso:	SC

7.1.7 REGISTRO DE PROBLEMAS

TAB.4.9 - Atividade Registrar Problemas – passo 2

Asset:	Componente do Sistema de Apoio à Decisão Logística
Cod Ident:	Co04
Versão:	1.0

Notificado:	Renato Alves
Email:	Renatoalves@gmail.com
Órgão:	ITA
Sistema:	Sistema de Apoio à Decisão Logistica de Meios Aeronáuticos
Problema:	O método classifitemrestr, Classe ItemRestrição – Co04-03, não está ordenado as restrições em ordem alfabética, crescente.

7.1.8 NOTIFICAÇÕES

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Análise de Domínio do Controlador Tático
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração

ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Análise de Domínio do Controlador Tático
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Arquitetura de Domínio do Controlador Tático
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Arquitetura de Domínio do Controlador Tático
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Componente de Domínio do Controlador Tático
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Componente de Domínio do Controlador Tático

VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Análise de Domínio do Treinador de Ataque
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Análise de Domínio do Treinador de Ataque
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Arquitetura de Domínio do Treinador de Ataque
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Arquitetura de Domínio do Treinador de Ataque
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR

	ASSETS

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Componente de Domínio do Treinador de Ataque
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Componente de Domínio do Treinador de Ataque
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Componente do Treinador de Ataque
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Componente do Treinador de Ataque
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS* (Notificação 10)

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Análise de Domínio do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Análise de Domínio do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Arquitetura de Domínio do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Arquitetura de Domínio do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade

PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Componente de Domínio do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Componente de Domínio do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

NOTIFICAÇÃO 2:	Atividade ARMAZENAR *ASSETS*
DE:	Gerente de Qualidade
PARA:	Gerente do Repositório
ASSUNTO:	Incluir novo *asset* no repositório
NOME DO ASSET:	Componente do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	-

NOTIFICAÇÃO 3:	Atividade CONTROLAR VERSÕES – CADASTRAR *ASSETS*
DE:	Gerente do Repositório
PARA:	Gerente de Configuração
ASSUNTO:	Cadastrar *asset* Controle de Versões
NOME DO ASSET:	Componente do Sistema de Apoio à Decisão Logística
VERSÃO:	1.0
OBS:	Essa notificação é enviada para a atividade CONTROLAR VERSÕES - CADASTRAR *ASSETS*

7.1.9 RELATÓRIO DE MÉTRICAS

Atividade Controle de Métricas

Número de *Assets* no Repositório:	11
Número de Acessos (sem download) ao Repositório, no trimestre: (dd/mm/aa até dd/mm/aa):	0
Número de Downloads do Repositório, no trimestre: (dd/mm/aa até dd/mm/aa)	4
Número Acumulado de Reuso para todos os *Assets* do Repositório	4

NÚMERO ACUMULADO DE REUSO POR *ASSET*:

Nome:	Análise de Domínio do Controlador Tático
Código:	AnD01
Reutilizado:	1

Nome:	Arquitetura de Domínio do Controlador Tático
Código:	ArD01
Reutilizado:	0

Nome:	Componentes de Domínio do Controlador Tático
Código:	CoD01
Reutilizado:	0

Nome:	Análise de Domínio do Treinador de Ataque
Código:	AnD02
Reutilizado:	0

Nome:	Arquitetura de Domínio do Treinador de Ataque
Código:	ArD02
Reutilizado:	1

Nome:	Componente de Domínio do Treinador de Ataque
Código:	CoD02
Reutilizado:	1

Nome:	Componente do Treinador de Ataque
Código:	Co02
Reutilizado:	0

Nome:	Análise de Domínio do Sistema de Apoio à Decisão Logística
Código:	AnD03
Reutilizado:	0

Nome:	Arquitetura de Domínio Sistema de Apoio à Decisão Logística
Código:	ArD03
Reutilizado:	0

Nome:	Componente de Domínio Sistema de Apoio à Decisão Logística
Código:	CoD04
Reutilizado:	0

Nome:	Componente do Sistema de Apoio à Decisão Logística
Código:	Co03
Reutilizado:	1

NÚMERO DE *ASSET*, DESENVOLVIDOS, POR ORGANIZAÇÃO, NO PERÍODO:

Nome da organização:	CASOP
Número de *assets*:	3

Nome da organização:	IPQM
Número de *assets*:	4

Nome da organização:	CASNAV
Número de *assets*:	4

NÚMERO DE REUSO, POR ORGANIZAÇÃO, NO PERÍODO:

Nome da organização:	CASOP
Número de reuso:	2

Nome da organização:	IME
Número de reuso:	1

Nome da organização:	ITA
Número de reuso:	1

7.2 APÊNDICE 2: FERRAMENTA DE APOIO AO PROCESSO DE GERÊNCIA DE *ASSETS*

O processo de desenvolvimento baseado em reuso necessita de ferramentas que auxiliem no armazenamento, busca, divulgação, compartilhamento e entendimento de *assets* para a construção de sistemas de software a partir dos mesmos.

Esta especificação tem a finalidade de apresentar descrição da Ferramenta de Apoio ao Processo de Gerência de *Assets*, através dos requisitos necessários para a sua utilização, da definição do escopo e dos requisitos funcionais que a ferramenta contempla.

7.2.1 REQUISITOS

7.2.1.1 ATIVIDADES DO PROCESSO DE GERÊNCIA

A ferramenta deverá dar suporte a todas as atividades definidas no Processo de Gerência de *Assets*.

- Abaixo segue a relação das atividades definidas no processo:
  - Aprovar Assets (A1);
  - Armazenar Assets (A2);
  - Atualizar *Assets* (A3);
  - Buscar *Assets* (A4);
  - Recuperar *Assets* (A5);
  - Remover *Assets* (A6);
  - Cadastrar Usuário (A7);
  - Controlar Versões - Cadastrar (A8);
  - Controlar Versões - Recuperar (A9);
  - Registrar Problemas (A10); e
  - Calcular Métricas (A11).

7.2.1.2 DEFINIÇÃO DO ESCOPO

A ferramenta de apoio ao Processo de Gerência de *Assets* é uma ferramenta desenvolvida para fornecer suporte difusão de *assets* através do ambiente WEB e, desta forma, permitir o compartilhamento de um repositório de *assets* dentro do Contexto do Ministério da Defesa e seus Comandos Subordinados e promover a reutilização de *assets* entre os desenvolvedores de sistemas.

Os usuários desta ferramenta serão as organizações que desenvolvem sistema e que são subordinadas ao Ministério da Defesa.

Os Desenvolvedores poderão compartilhar os *assets* desenvolvidos em seus trabalhos entre si e, desta forma, o repositório terá o papel de centralizar e disponibilizar os *assets* para reuso.

A tecnologia da WEB vai permitir a independência de plataforma, acesso distribuído, informação compartilhada e facilidade de acesso. Um aspecto importante a ser ressaltado é a segurança de acesso como um dos pontos fundamentais, pois o compartilhamento do repositório será realizado através da infra-estrutura da WEB, que é uma rede pública.

Para isso, existe a necessidade de estabelecer um controle de acesso aos recursos da ferramenta somente aos usuários autorizados que pertencem às organizações subordinadas ao Ministério da Defesa e que já foram previamente selecionadas e, além disso, controlar os recursos que vão ser disponibilizados para cada tipo de categoria de usuário.

7.2.1.3 REQUISITOS DE SISTEMA

A ferramenta proposta deverá permitir:

- Certificação dos *assets*;
- Classificação dos *assets*;
- Catalogação dos *assets*;
- Armazenamento do *assets* e sua documentação em um repositório de dados;
- Atualização de um *asset* e sua documentação (nova versão);
- Busca de *assets* pelos usuários de sistema;

- Recuperação de *assets* pelos usuários do sistema;
- Submissão de *assets* ao sistema pelos usuários;
- Remoção de *assets* obsoletos;
- Cadastro de Usuários no sistema;
- Controle de acesso;
- Notificações para os usuários;
- Controle de Versões;
- Registro de Problemas pelos usuários; e
- Levantamento de Métricas.

7.2.1.4 DESCRIÇÃO DOS REQUISITOS

1. O Usuário do sistema vai ter disponíveis os seguintes recursos:

- Busca: possibilita que o usuário realize buscas de *assets* no repositório, através de recursos para estabelecer critérios e parâmetros sobre as características do *asset* que está sendo procurado. Isto é possível através da definição de um esquema de classificação para o repositório. As buscas podem ser:
- Busca simplificada pôr palavra chave; e
- Busca complexa (alto nível).
- Submissão: possibilita que o usuário submeta seus *assets* candidatos para serem certificados;
- Recuperação: disponibiliza recursos para auxiliar no processo de extrair o *assets* para a sua reutilização na construção de uma nova aplicação; e
- Registro de Problemas.

2. Já a equipe de Gerente de *Assets* vai ter disponíveis os seguintes recursos como suporte às atividades de gerência *assets*:

- Esquema de classificação: disponibiliza os recursos para criar e manter a estrutura do esquema de classificação dos *assets* no repositório;
- Catalogação: disponibiliza os recursos para a inclusão do *asset* e das informações relacionadas no repositório e permitir a manutenção do mesmo;

- Certificação dos *assets*: disponibilizam os recursos para validar os *assets* candidatos a fazerem parte do repositório ou as novas versões de *assets* que já fazem parte do repositório;
- Divulgação: disponibiliza os recursos para que o Gerente de *Assets* divulgue, para os usuários, informações sobre novos *assets*, atualização de versões, retiradas e, até mesmo, erros encontrados;
- Controle de Versões;
- Registro de Problemas; e
- Métricas do Repositório.

No ambiente de reutilização que está sendo proposto, o Usuário pode ser tanto o que produz *assets*, como o que consome. O Gerente de *Assets* é a pessoa responsável pela gerência dos *assets* armazenados no repositório.

A Ferramenta visa atender tanto as atividades dos Usuários como as do Gerente de *Assets*:

- Usuários (Engenheiros de Requisitos, Desenvolvedores e Montadores); e
- Gerente de *Assets* (Gerente do Repositório, Gerente de Configuração e Gerente de Qualidade).

7.2.2 DESCRIÇÃO DOS CASOS DE USO

Para melhor especificação e entendimento do sistema, ele foi dividido em três sub-sistemas:

- Sub-sistema de Produção;
- Sub-sistema de Gerência; e
- Sub-sistema de Consumo.

7.2.2.1 LISTA DE ATORES E OBJETIVOS

TAB. 7.1: Lista de atores e objetivos

ATOR	OBJETIVO
Usuário	Submeter *asset* candidato à certificação (1)
Usuário	Buscar *asset* (2)
Usuário	Recuperar *asset* (3)
Usuário	Registrar problema (4)
Usuário	Cadastra Usuário (5)
Gerente do Repositório	Armazenar *asset* (6)
Caso de Uso "Armazenar asset"	Classificar *asset* (7)
Caso de Uso "Armazenar asset"	Catalogar *asset* (8)
Gerente do Repositório	Remover *asset* (9)
Casos de Uso "Armazena, Recupera e Remove asset"	Controlar versão (10)
Gerente do Repositório	Calcular métricas (11)
Gerente de Qualidade	Certificar *asset* (12)

7.2.2.2 CASOS DE USO – SUB-SISTEMA DE PRODUÇÃO

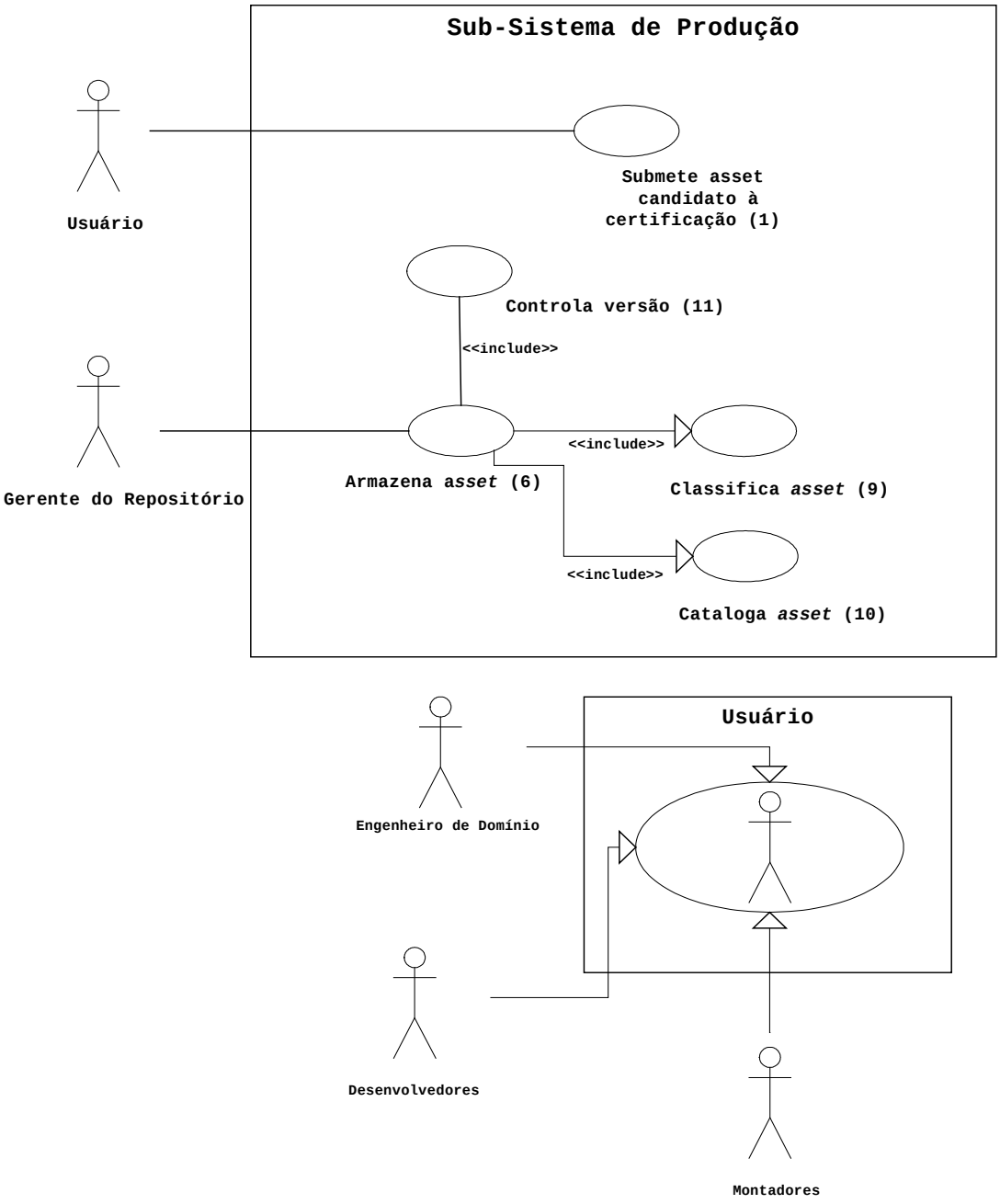


FIG. 7.1: *Use Case* do Sub-Sistema de Produção

7.2.2.3 CASOS DE USO – SUB-SISTEMA DE CONSUMO

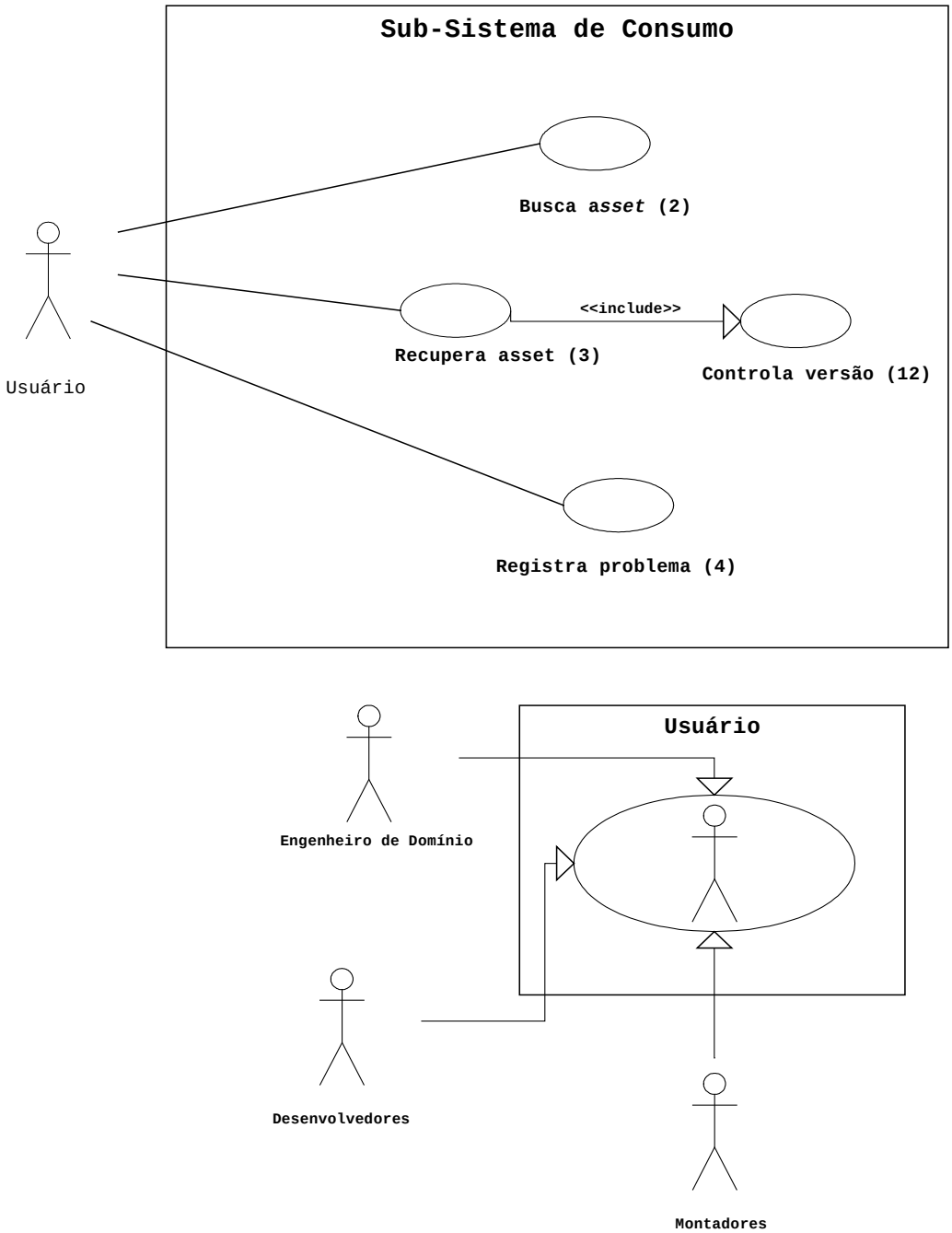


FIG. 7.2: *Use Case* do Sub-Sistema de Consumo

7.2.2.4 CASO DE USO – SUB-SISTEMA DE GERÊNCIA

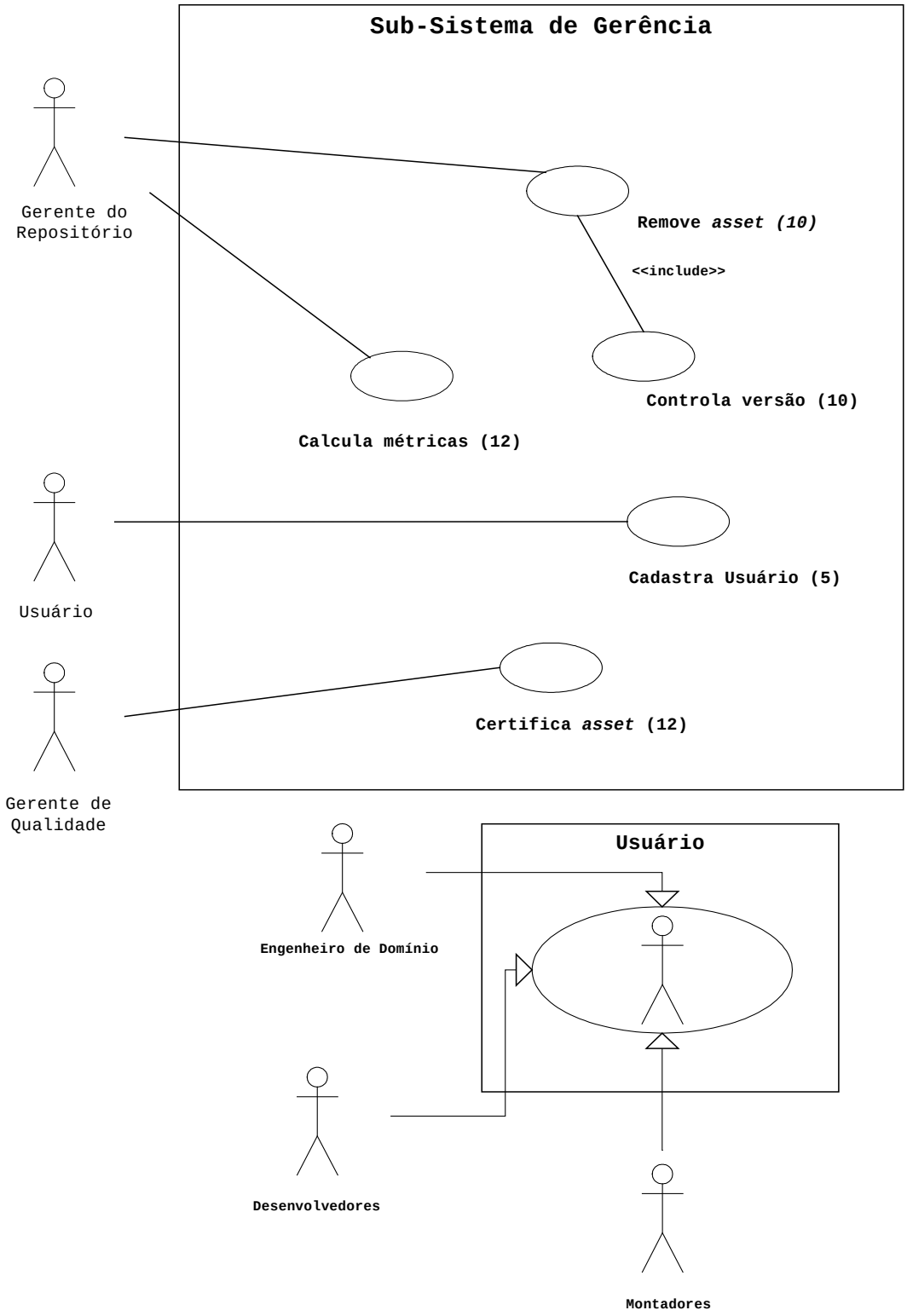


FIG. 7.3: *Use Case* do Sub-Sistema de Gerência

7.2.3 CENÁRIOS

Caso de Uso: Submete asset candidato a certificação;

Objetivo: Submeter o *asset* para ser certificado pelo Gerente de Qualidade;

Ator Primário: Usuário;

Ator Secundário: -

Gatilho: Certificar asset;

Pré-condição: Usuário estar logado no sistema;

Pós-condição: Asset enviado para certificação;

Seqüência Típica de Eventos:

Usuário armazena o *asset* na área de transferência de *assets* a serem certificados;

Sistema notifica Gerente de Qualidade;

Caso de Uso encerrado.

Caso de Uso: Busca asset;

Objetivo: Procurar *assets* candidatos à reutilização;

Ator Primário: Usuário;

Ator Secundário: -

Gatilho: Recuperar asset;

Pré-condição: Usuário estar logado no sistema;

Pós-condição: Busca efetuada;

Seqüência Típica de Eventos:

Usuário fornece as palavras-chaves conforme critérios de busca do sistema;

Sistema executa busca;

Sistema encontra *asset(s)* com as características definidas pelo Usuário;

Sistema fornece uma lista com a relação dos *asset* e sua descrição;

Caso de Uso encerrado.

Seqüência Alternativa:

Passo3;

Gatilho: Sistema não encontra *asset* com as características fornecidas pelo Usuário;

Sistema informa que não foram encontrados *assets* que atendam aos critérios estabelecidos pelo usuário;

Retorna ao passo 5 do curso normal.

Caso de Uso: Recupera asset;

Objetivo: Recuperar *asset* para reuso;

Gatilho: Asset selecionado;

Ator Primário: Usuário;

Ator Secundário: -

Pré-condição: Usuário ter selecionado o *asset* da lista de *assets* candidatos;

Pós-condição: Asset disponibilizado para reuso;

Seqüência Típica de Eventos:

Usuário solicita *asset* através da lista de *assets* candidatos;

Sistema disponibiliza o *asset* solicitado;

Sistema informa Caso de Uso Controla Versão para armazenar informações sobre *asset* recuperado;

Caso de Uso encerrado.

Caso de Uso: Registra problema;

Objetivo: Permitir que o Usuário reporte os problemas detectados no *asset*;

Ator Primário: Usuário;

Ator Secundário: -

Gatilho: Notificar problema;

Pré-condição: Usuário estar logado no sistema;

Pós-condição: Usuários notificados do problema detectado;

Seqüência Típica de Eventos:

Usuário registra no sistema o problema detectado;

Sistema procura usuários que estejam reutilizando o *asset*;

Sistema informa o(s) usuário(s) sobre o problema detectado

Caso de Uso encerrado.

Seqüência Alternativa:

Passo2;

Gatilho: Sistema não encontra usuários que estejam reutilizando o asset;

Retorna ao passo 4 do curso normal.

Caso de Uso: Cadastra Usuário;

Objetivo: Permitir o cadastro do Usuário no sistema;

Ator Primário: Usuário;

Ator Secundário: -

Gatilho: Solicitar cadastro;

Pré-condição: Usuário não estar cadastrado;

Pós-condição: Usuário cadastrado;

Seqüência Típica de Eventos:

Usuário requisita cadastro no sistema;

Sistema verifica dados do Usuário;

Sistema aceita cadastro e atribui o tipo de acesso ao sistema;

Sistema notifica usuário; e

Caso de Uso encerrado.

Seqüência Alternativa:

Passo2;

Gatilho: Sistema não aceita cadastro do usuário;

Retorna ao passo 4 do curso normal.

Caso de Uso: Armazena asset;

Objetivo: Permitir que o Gerente do Repositório armazene *assets* no repositório;

Ator Primário: Gerente do Repositório;

Ator Secundário: -

Gatilho: Asset Certificado

Pré-condição: Asset deve ter sido certificado pelo Gerente de Qualidade;

Pós-condição: Asset armazenado no repositório;

Seqüência Típica de Eventos:

Gerente do Repositório informa as palavras chaves do *asset* para classificação no sistema;

Include Classifica;

Gerente do Repositório informa o conteúdo da documentação do *asset* para a catalogação no sistema*;*

Include Cataloga;

Sistema armazena informação da versão no Controle de Versão

Include Controla Versão;

Sistema notifica Usuários

Caso de Uso encerrado.

Caso de Uso: Classifica asset;

Objetivo: Classificar o *asset* a partir de palavras-chaves fornecidas pelo Gerente do Repositório;

Ator Primário: Caso de Uso Armazena *asset*;

Ator Secundário: -

Gatilho: Armazenar *asset*;

Pré-condição: Asset certificado;

Pós-condição: Asset classificado;

Seqüência Típica de Eventos:

Gerente do Repositório recupera documentação do *asset*;

Gerente do Repositório, conforme classificação em facetas, gera esquema de classificação do *asset*, segundo as informações contidas na documentação;

Gerente do Repositório armazena o esquema de classificação no repositório;

Caso de Uso encerrado.

Caso de Uso: Cataloga asset;

Objetivo: Permitir que o Gerente do Repositório armazene as informações do *asset* no repositório;

Ator Primário: Gerente do Repositório;

Ator Secundário: -

Gatilho: Armazenar *asset*;

Pré-condição: Asset classificado;

Pós-condição: Asset catalogado;

Seqüência Típica de Eventos:

Gerente do Repositório recupera documentação do *asset*;

Gerente do Repositório, armazena a documentação do *asset* segundo a estrutura de armazenamento (catálogo) do repositório;

Caso de Uso encerrado.

Caso de Uso: Remove asset;

Objetivo: Permitir que o Gerente do Repositório remova *assets* do repositório;

Ator Primário: Gerente do Repositório;

Ator Secundário: -

Gatilho: Relatório de Métricas;

Pré-condição: A*sset* com baixo índice de reuso;

Pós-condição: A*sset* removido do repositório;

Seqüência Típica de Eventos:

Gerente do Repositório solicita índice de reuso dos *assets*;

Sistema, a partir do Relatório de Métricas, verifica *assets* com baixo índice de reuso;

Sistema verifica se os referidos *assets* estão sendo reutilizados;

Sistema informa ao Gerente do Repositório os *assets* que podem ser retirados do repositório;

Gerente do Repositório analisa e seleciona os *assets* que devem ser retirados;

Sistema exclui *assets* do repositório*;*

Include Controla Versão;

Caso de Uso encerrado.

Caso de Uso: Controla Versão;

Objetivo: Controlar as diferentes versões e diferentes usuários dos *assets* armazenados no repositório;

Ator Primário:

Ator Secundário: -

Gatilho: Incluir, alterar, remover ou reusar *asset* do repositório;

Pré-condição:

Pós-condição:

Seqüência Típica de Eventos:

Sistema verifica qual operação deve ser feita no Controle de Versões;

Sistema inclui dados do novo *asset* no Controle de Versões;

Sistema notifica Usuários sobre novo *asset* no repositório;

Caso de Uso encerrado.

Seqüência Alternativa:

Passo1;

Gatilho: Inclusão de uma nova versão;

Sistema inclui dados da nova versão do *asset* no Controle de Versões;

Sistema notifica Usuários sobre uma nova versão do *asset* no repositório;

Retorna ao passo 4 do curso normal.

Seqüência Alternativa:

Passo1;

Gatilho: Inclusão dos dados do usuário;

Sistema inclui dados do usuário, que está reutilizando o asset, no Controle de Versões;

Retorna ao passo 4 do curso normal.

Seqüência Alternativa:

Passo1;

Gatilho: Remoção dos dados do asset;

Sistema remove os dados do *asset* do Controle de Versões;

Retorna ao passo 4 do curso normal.

Caso de Uso: Calcula métricas;

Objetivo: Calcular as métricas do repositório;

Ator Primário: Gerente do Repositório;

Ator Secundário: -

Gatilho: Elaborar relatório trimestral de métricas do repositório;

Pré-condição: Ter atingido a data de emissão do relatório;

Pós-condição: Relatório concluído;

Seqüência Típica de Eventos:

Sistema calcula número de componentes no repositório;

Sistema calcula número de acessos (sem *downloads*) ao repositório, por período;

Sistema calcula número de *downloads* no repositório, por período;

Sistema calcula número acumulado de reuso, por *asset*;

Sistema calcula número acumulado de reuso para todos os assets do repositório;

Sistema gera relatório de métricas; e

Caso de Uso encerrado.

Caso de Uso: Certifica asset;

Objetivo: Atestar a qualidade dos *assets* para serem incluídos no repositório;

Ator Primário: Gerente de Qualidade;

Ator Secundário: -

Gatilho: Notificação do Usuário;

Pré-condição: Submeter *asset* candidato à certificação;

Pós-condição: Asset certificado;

Seqüência Típica de Eventos:

Gerente de Qualidade aplica check-list;

Gerente de Qualidade aplica técnica de leitura horizontal;

Gerente de Qualidade aplica técnica de leitura vertical;

Sistema emite relatório de verificação;

Sistema notifica Gerente do Repositório; e

Caso de Uso encerrado.

Seqüência Alternativa:

Passo;

Gatilho: Asset não foi aprovado nos teste de certificação;

Sistema notifica Usuário;

Retorna ao passo 6 do curso normal.

8 GLOSSÁRIO DE TERMOS TÉCNICOS E EXPRESSÕES USADAS

- **ANÁLISE DE DOMÍNIO:**

(A) A análise de sistemas, dentro de um domínio, para descobrir semelhanças e diferenças entre eles.

(B) O processo através do qual a informação utilizada no sistema de desenvolvimento de software é identificada, capturada e organizada de forma a poder ser reutilizada para criar novos sistemas dentro de um mesmo domínio.

(C) O resultado do processo (A) e (B)

- **ARQUITETURA DE DOMÍNIOS.** Uma genérica estrutura organizacional ou design de sistemas de software dentro de um domínio. A arquitetura de domínio contém os designes (projetos) com os quais se pretende satisfazer os requisitos especificados no modelo do domínio. A arquitetura de domínio documenta o design, assim como o modelo de domínio documenta requisitos. Uma arquitetura de domínios: 1) pode ser adaptada para criar designes para sistemas de software dentro de um domínio, e 2) prove um *framework* para configurar assets em sistemas de software individuais

- **ARTEFATO**, s. m. Uma peça de informação tangível que é criada, alterada e usada durante o desenvolvimento das atividades; representa uma área de responsabilidade; e é passível de ser colocada separadamente sobre um controle de versões. Um artefato pode ser um modelo, um elemento de modelo ou um documento. (JACOBSON et al., 1999)

- **ARTEFATO REUTILIZÁVEL**,s.m. Também denominado *"Asset"*.

- ***ASSET***, s. m. *Asset* é um artefato ou conjunto de artefatos que tenham sido criados com um propósito explícito de ser reutilizado em outros esforços de desenvolvimento. Neste sentido, ele é a unidade que irá ser produzida, utilizada e gerenciada para permitir o reuso (RATIONAL SOFTWARE CORPORATION, 2005).

- **ATIVO DE SOFTWARE**, s. m. Também denominado *"Asset"*.

- **CLASSIFICAÇÃO,** A maneira como assets são organizados para facilitar buscas e extrações em uma biblioteca de reuso (ISO/IEC 12207:1995/FDAM 1)

- **COMPONENTE**, s. m. Um pacote coerente de artefatos de *software* que podem ser desenvolvidos independentemente e entregues como uma unidade e que podem ser

compostos, ou trocados, com outros componentes para formar algo maior. (D´SOUZA et al., 1998)

- **DEFEITO**, s. m. Passo, processo ou definição de dados incorretos, por exemplo, um comando incorreto. (IEEE610.12, 1990)
- **DEFINIÇÃO DE DOMÍNIO,** O processo de definir o escopo e os limites de um domínio. O resultado deste processo.
- **DOMÍNIO**, s. m. O espaço do problema. (NIST, 1994)
- **ENGENHARIA DE DOMÍNIO**. Uma linha baseada em reuso para definir o escopo (definição do domínio), especificar a estrutura (arquitetura do domínio), e construir os *assets* (por exemplo, projeto de requisitos, código de *software*, documentação) para uma classe de sistemas, subsistemas ou aplicações. Engenharia de domínio pode incluir as seguintes atividades: definição do domínio, análise de domínio, desenvolver a arquitetura de domínio e implementação do domínio. (NIST, 1994)
- **ENGENHEIROS DE DOMÍNIO**. Um grupo que desempenha atividades de engenharia de domínio (incluindo análise de domínio, projeto de domínio, construção de *asset* e manutenção de *asset*). (IEEE1517, 1999)
- **GARANTIA DA QUALIDADE**. Conjunto de atividades planejadas e sistemáticas, implementadas no sistema da qualidade e demonstradas como necessárias para prover confiança adequada de que uma entidade atenderá os requisitos para a qualidade. (ISO8402, 1994)
- **LINHA DE PRODUÇÃO,** Uma coleção de sistemas que são potencialmente deriváveis de uma única arquitetura de domínios.
- **MODELO DE DOMÍNIO**. Um produto da análise de domínio que prove uma representação dos requisitos do domínio. O modelo de domínio identifica e descreve a estrutura dos dados, fluxo de informações, funções, restrições e controles do domínio que são incluídos no sistema de *software* dentro do domínio. O modelo de domínio descreve as semelhanças e variações entre os requisitos dos sistemas de *software* no domínio. (NIST, 1994)
- **PROCEDIMENTO**, s. m. Forma específica de executar uma atividade ou um processo. (ISO9000, 2001)
- **PROCESSO DE *SOFTWARE***. Conjunto de atividades, métodos, práticas e transformações que as pessoas empregam para desenvolver e manter *software* e os produtos

associados (por exemplo, planos de projeto, documentos de projeto, código, casos de teste, manual do usuário). (ISO8402, 1994)

- **QUALIDADE**, s. f. Grau no qual um conjunto de características inerentes satisfaz a requisitos. (ISO9000, 2001)
- **RASTREABILIDADE**, s. f. Capacidade de recuperar o histórico, a aplicação ou a localização daquilo que está sendo considerado. (ABNT9000, 2001)
- **REQUISITO**, s. m. Necessidade ou expectativa do cliente e de outra parte interessada, geralmente explicitada como condição de negócio no contrato com o fornecedor para resolver um problema ou alcançar um objetivo. É uma característica, tais como especificação técnica, prazo de entrega, garantia, que o cliente "requer" do produto. (ABNT9000, 2001)
- **RETRABALHO**, s. m. Ação sobre um produto não-conforme, afim de torná-lo conforme aos requisitos. (ABNT9000, 2001)
- **REUSABILIDADE**, s. f. (A) O grau que um *asset* pode ser utilizado em mais de um sistema de *software* ou na construção de outros *assets*. (B) Em uma biblioteca de reuso, a característica de um *asset* de ser utilizado em diferentes contextos, sistemas de software ou na construção de diferentes *assets*. (IEEE1517, 1999)
- **REUSO**, s. m. O uso de um *asset* na solução de diferentes problemas. (IEEE1517, 1999)
- **REUSO SISTEMÁTICO**. A prática de reuso de acordo com um processo bem definido e repetível. (IEEE1517, 1999)
- **SISTEMA**. Uma composição integrada que consiste de um ou mais processos, *hardware*, *software*, facilidades e pessoas, que provêem uma capacidade de atender uma necessidade ou objetivo definido. (ABNT12207,1998)
- ***STAKEHOLDER***. Um grupo ou indivíduo que possui um grande interesse no sucesso do sistema de *software* que está sendo desenvolvido e possui alguma influência direta ou indireta nos requisitos do sistema. (HEINEMAN et al.,2001)
- ***TEMPLATE***. Um *asset* com parâmetros ou *slots* que podem ser usados para construir uma instância de *asset*. (IEEE1517,1999)
- **USABILIDADE**, s. f. Conjunto de atributos que evidenciam o esforço necessário para se poder utilizar o *software*, bem como o julgamento individual desse uso, por um conjunto explícito ou implícito de usuários. (ISO9126-1,2000)

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Administração
Baixar livros de Agronomia
Baixar livros de Arquitetura
Baixar livros de Artes
Baixar livros de Astronomia
Baixar livros de Biologia Geral
Baixar livros de Ciência da Computação
Baixar livros de Ciência da Informação
Baixar livros de Ciência Política
Baixar livros de Ciências da Saúde
Baixar livros de Comunicação
Baixar livros do Conselho Nacional de Educação - CNE
Baixar livros de Defesa civil
Baixar livros de Direito
Baixar livros de Direitos humanos
Baixar livros de Economia
Baixar livros de Economia Doméstica
Baixar livros de Educação
Baixar livros de Educação - Trânsito
Baixar livros de Educação Física
Baixar livros de Engenharia Aeroespacial
Baixar livros de Farmácia
Baixar livros de Filosofia
Baixar livros de Física
Baixar livros de Geociências
Baixar livros de Geografia
Baixar livros de História
Baixar livros de Línguas

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo