Sistemas de Apoio à Decisão: análise do processo de software

The Decision-Making Systems: the software process analysis

*Tenente Coronel Aviador Amilton Ferreira da Silva[1,2]

1 Aluno do Curso de Comando e Estado Maior 2007 ECEMAR/ UNIFA
2 Aluno do MBA de Gestão de Processos pela Universidade Federal Fluminense - UFF

RESUMO

Este artigo objetivou analisar os principais fatores que impactam o processo de desenvolvimento de sistemas do Centro de Computação da Aeronáutica de Brasília - CCA BR, em virtude da necessidade de se desenvolver sistemas de apoio à decisão (SAD). O estudo ocorreu sob uma abordagem analítica e foi motivado em função dos seguintes fatos: aprovação da Norma de Sistema do Comando da Aeronáutica (NSCA 7-6) que atribuiu ao CCA BR diretrizes específicas para prover soluções de apoio à decisão da alta administração; o modelo de processo de software do CCA BR destina-se aos sistemas transacionais e não aos SAD. Para conduzir o trabalho foi selecionado o tipo de pesquisa bibliográfica, utilizando-se da metodologia qualitativa. Inicialmente, buscou-se descrever os principais aspectos da Engenharia de *Software* com ênfase no Processo. Na seqüência, relataram-se as principais características do modelo de Processo Unificado da *Rational* e apresentou-se uma visão conceitual sobre os sistemas de informação, destacando-se o SAD. Finalmente, à luz dos conceitos, os principais fatores foram identificados e descritos na forma de análise das fases do Processo, em prol da adaptação às características do novo tipo de software. Concluiu-se que o desenvolvimento de SAD com o modelo de processo da Rational é factível, desde que ocorram ajustes na metodologia, principalmente no que se refere ao levantamento de requisitos e à tecnologia de banco de dados.

Palavras-chave: Engenharia de software. Processo de software. Processo rotacional unificado. Sistemas de Apoio à Decisão.

Recebido: 26/03/2008 **Revisado:** 25/08/2008 **Aceito:** 01/09/2008

***Autor**: Tenente Coronel Aviador Amilton Ferreira da Silva, formado pela Academia da Força Aérea em 1988; Curso de Especialização em Análise de Sistemas (ITA); MBA em Administração Estratégica de Sistemas de Informação (FGV); Aluno do Curso de Comando e Estado-Maior 2007; MBA em Gestão de Processos (UFF-2007). **Contato**: e-mail: amiltonfs@bol.com.br.

ABSTRACT

The present article aimed to analyze the main factors the impacting the software development systems in the Brasilia Aeronautical Computer Center – CCA BR, in order to the necessity in develop the decision-making systems (SAD). The study was made under an analytic approach and was motivated by the following factors: the Aeronautical Command System Rule (NSCA 7-6) which assign to the CCA BR specifics lines of direction to provide any solutions that can support the High Direction decisions; the current CCA BR's software process model was developed in view of the ordinary decisions and not to the SAD. The research was conducted by the literature review with the qualitative methodology approach. Initially were described the main aspects related with software engineering emphasizing the development process. Afterwards were commented the main model features of Rational Unified Process (RUP) and were presented a conceiting view about the information systems, pointing out the SAD. Finally in light of the concepts, the main factors were identified and described in the process phase's analysis' pattern in order to the new kind of software features adaptation. Was concluded that the SAD development using the RUP model is possible, once some methodology adjustment can be done, mainly at the part related to the requirements identification and the database technology.

Keywords: *Software engeneering. Software process. Rational Unified Process. Decision-making Systems.*

INTRODUÇÃO

Nos dias atuais, uma das atribuições mais desafiadoras para comandantes, chefes ou diretores é a tomada de decisão. A escolha definitiva da melhor solução para um problema requer uma análise criteriosa das opções, norteada pelo exame das vantagens consideradas mais significativas.

Numa organização desprovida de sistemas de informação que auxiliem o processo decisório, os gerentes assumem suas decisões baseadas em dados históricos e experiências individuais.

No passado, o empirismo era bem mais eficiente, pois os eventos ocorriam com morosidade, permitindo o método da tentativa e erro. Na atualidade, a dinâmica dos acontecimentos que envolvem a tomada de decisão e a evolução tecnológica delineia um cenário caracterizado pela análise de grande volume de dados e exigüidade do tempo. Com isso, a ausência de recursos que ajudem os gerentes pode ocasionar resultados indesejáveis. Segundo Laudon (2004), os sistemas de apoio à decisão (SAD) favorecem os gerentes nos processos decisórios considerados não triviais, ou seja, aqueles em que as decisões mudam com rapidez e há dificuldades para se especificar o procedimento de obtenção das respostas às questões gerenciais, antecipadamente.

Laudon (2004) destaca, ainda, que os SAD sustentam a tomada de decisão por meio de um conjunto flexível de ferramentas e capacidades para analisar dados, possibilitando melhores condições para essa atividade. Essa observação alinha-se com o fato do administrador necessitar de informação com qualidade e agilidade a respeito de operações, tendências e mudanças no dia-a-dia dos negócios. Portanto, o desenvolvimento de um SAD que atenda às gerências da organização com robustez e eficiência requer o suporte de modernos princípios tecnológicos.

No âmbito do Comando da Aeronáutica – COMAER, o Centro de Computação da Aeronáutica de Brasília – CCA BR é a organização diretamente subordinada ao Departamento de Controle do Espaço Aéreo – DECEA que tem a missão de apoiar a execução dos processos e atividades da alta gerência da Aeronáutica, por meio de soluções baseadas na tecnologia da informação.

No que se refere ao desenvolvimento de sistemas informatizados, o CCA BR especializou-se na implementação de sistemas de informação do tipo transacional, utilizando o modelo de Processo Unificado da Rational, em inglês *Rational Unified Process* (RUP). Esses sistemas provêem suporte ao dia-a-dia das atividades administrativas do COMAER,

De acordo com Laudon (2004), os sistemas transacionais são caracterizados por dados muito acessados, detalhados e com foco nas situações instantâneas e não em registros históricos.

Em termos organizacionais, recentemente o DECEA aprovou a Norma de Sistema do Comando da Aeronáutica " DIRETRIZES ESPECÍFICAS PARA OS CENTROS DE COMPUTAÇÃO DA AERONÁUTICA (CCA) - NSCA 7-6 , de 27 de outubro de 2005, que estabelece atribuições específicas para cada CCA, dentre as diversas atividades abrangidas pela tecnologia da informação.

Assim, cabe ao CCA BR, de acordo com o item 3.2.1 da NSCA 7-6, "disponibilizar os dados corporativos e a infra-estrutura de processamento necessária ao apoio à decisão da alta administração da Força" (BRASIL, 2005, p.9). Constata-se, portanto, que a NSCA 7-6 legitima a competência do CCA BR para desenvolver sistemas do tipo SAD, voltados para a área administrativa do COMAER.

O processo de desenvolvimento de software, pautado no modelo RUP, e o SAD tornam-se as idéias centrais deste artigo. No que se refere à metodologia, a partir de uma revisão da literatura buscar-se-á identificar os possíveis fatores que contribuam para a conformidade entre o RUP e os SAD.

De acordo com a classificação proposta por Gil (2002), o presente trabalho pode ser qualificado quanto à finalidade como uma pesquisa exploratória, uma vez que a partir do objetivo geral o autor deste estudo pretende torná-lo mais familiar com o intuito de analisar os principais aspectos do problema.

Inicialmente, descrevem-se os principais aspectos da Engenharia de Software com ênfase no Processo, abordando-se a evolução dos diversos modelos até o RUP, sobre o qual se evidenciarão suas características mais relevantes.

Na seqüência, apresentam-se os seguintes tópicos: uma visão conceitual sobre os sistemas de informação, o papel desse recurso nas organizações e a classificação segundo a qual os cientistas os caracterizam, destacando-se os SAD.

Finalmente, à luz dos conceitos, proceder-se-ão inferências sobre os possíveis fatores a serem observados nas diversas fases do RUP, a fim de fornecer a adequação necessária para desenvolver SAD, de acordo com esse modelo de processo de software.

1 ENGENHARIA DE SOFTWARE

Segundo o Institute of Electrical and Electronics Engineers - IEEE (1993), define-se a Engenharia de Software como a aplicação de uma abordagem sistemática, disciplinada e quantificável, para o desenvolvimento, operação e manutenção do software. O significado da definição deve considerar, ainda, as características peculiares do software, quando comparado a um produto manufaturado.

A Engenharia de *Software* é uma disciplina que aplica os princípios de engenharia com o objetivo de produzir software de alta qualidade a baixo custo, utilizando-se de modelos de processo de software para atingir tal objetivo.

Segundo Pressman (2002), visualiza-se a Engenharia de Software como uma tecnologia em camadas, apoiada num compromisso organizacional com a qualidade e fundamentada na camada de processo (vide Figura 1).

Figura 1 – Camadas da Engenharia de Software.
Fonte: Pressman (2002, p.19).

Dessa forma, a camada de processo fornece a união das camadas de tecnologia (métodos e ferramentas) com o intuito de tornar possível o desenvolvimento de *software*, de forma racional e oportuna. Os métodos fornecem a técnica de como fazer pra construir software, enquanto que as ferramentas propiciam o apoio automatizado ou semi-automatizado para o processo e métodos.

Enquanto Pressman (2002) destaca a qualidade e a racionalidade do desenvolvimento de *software*, Sommerville (2003) enfatiza a complexidade e a boa relação custo-benefício. Percebe-se que os pensamentos de ambos complementam-se na medida em que a qualidade torna-se cada vez mais um requisito essencial e os custos requerem um efetivo controle.

1.1 PROCESSO DE SOFTWARE

Em um processo de desenvolvimento de software, identifica-se um conjunto de três elementos fundamentais: métodos, ferramentas e procedimentos para projetar, construir e manter grandes sistemas de software de forma profissional.

Segundo Pressman (2002, p.17),

> O processo é um diálogo no qual o conhecimento, que deve se transformar em software, é reunido e embutido no software. O processo provê interação entre usuários e projetistas, entre usuários e ferramentas em desenvolvimento e entre projetistas e ferramentas em desenvolvimento [tecnologia]. É um processo interativo no qual a própria ferramenta serve como meio de comunicação, com cada nova rodada de diálogo atraindo mais conhecimento útil do pessoal envolvido.

Pressman (2002) destaca a questão da interatividade entre as pessoas envolvidas no processo de software. Na prática, constata-se que a qualidade do produto software resulta de uma comunicação clara, objetiva e padronizada, a fim de proporcionar a transformação do conhecimento humano em linguagem de computador. Esse entrosamento no campo cognitivo, pautado na comunicação e interatividade diferencia o processo de software de um processo industrial em que existe uma especificidade bem definida.

De acordo com Sommerville (2003, p.7), "Um processo de software é um conjunto de atividades e resultados associados que geram um produto de software. Essas atividades são, em sua maioria, executadas por engenheiros de software". O autor salienta as atividades e seus resultados com foco na obtenção do produto final. Sob essa perspectiva menciona alguns aspectos, tais como: a organização das atividades, os níveis de detalhamento, os prazos e a adequação dos processos, admitindo variações para diferentes organizações que produzam o mesmo tipo de produto. (SOMMERVILLE, 2003).

1.2 MODELOS DE PROCESSO DE SOFTWARE

Uma vez compreendido o processo de *software*, adota-se um modelo que contemplará o conjunto das atividades, métodos, técnicas e ferramentas que garantem que o software seja produzido com alta qualidade e baixo custo.

Segundo Raccoon (1995, p.56), o desenvolvimento de software pode ser caracterizado como um "ciclo de solução de problema linear". Neste sentido, Raccoon busca generalizar o trabalho de Engenharia de Software, adotando o ciclo em diferentes níveis de resolução. Para tanto, classifica o ciclo em "quatro estágios distintos: situação atual (status quo), definição do problema, desenvolvimento técnico e integração da solução" (vide Figura 2).

Figura 2 – Fases de um ciclo de solução de problema linear.
Fonte: Adaptada de Raccoon (1995)

A partir do ciclo de solução de problema linear, Raccoon (1995, p.56) sugere um "modelo de caos" em que os estágios são aplicados recursivamente às necessidades do usuário e à especificação técnica do *software*. Esse ciclo de solução de problema adequa-se ao trabalho de Engenharia de Software em muitos diferentes níveis de resolução. Pode ser usado em nível macro, quando se considera o produto software com um todo; em nível intermediário, quando os componentes de programa estão passando por engenharia e até mesmo no nível de linha de código.

A literatura descreve vários modelos de processo de software, idealizados para ajudar no controle e na coordenação de um projeto de software. Observa-se que embora apresentem paradigmas distintos, trazem na raiz de suas concepções características do "modelo de caos" definido por Raccoon (1995). Nesse modelo, o autor descreve o desenvolvimento de software como um continuum[2] desde a concepção macro do projeto até cada linha de código, envolvendo

[2] [Lat.] Conjunto compacto e conexo; conjunto contínuo.

aspectos humanos e técnicos em todos os níveis. Tal descrição utiliza uma estrutura flexível que reflete o padrão intricado que ocorre em projetos reais. Relata, ainda, que os padrões caóticos entre os níveis de um projeto explicam a complexidade do desenvolvimento de software.

Segundo Pressman (2002, p.26), "os modelos representam uma tentativa de trazer ordem para uma atividade inerentemente caótica". Já Sommerville (2003) refere-se ao caos de forma mais branda, descrevendo os modelos como uma simplificação da realidade. Ambos os autores emergem o grande desafio dos engenheiros de software na busca pelo modelo genérico de baixa complexidade.

De acordo com Pressman (2002), destacam-se os seguintes modelos de processo de software: seqüencial linear, prototipagem, rapid application development – RAD (desenvolvimento rápido de aplicações), evolucionário, baseado em componentes, métodos formais e técnicas de quarta geração.

Para Sommerville (2003), os modelos de processo de software consistem em: cascata, evolucionário, formal e orientado ao reuso.

Para efeito deste trabalho, o enfoque teórico destacará o modelo de prototipagem e o modelo evolucionário específico denominado Unified Process – UP (Processo Unificado) e o RUP, este último aplicado no CCA BR.

1.2.1 MODELO DE PROTOTIPAGEM

O CCA BR utilizou o modelo de prototipagem durante as décadas de 80 e 90. A principal característica deste paradigma consiste em estabelecer os objetivos gerais do software (Pressman, 2002).

Inicialmente, identificam-se as necessidades conhecidas e esboçam-se áreas que precisam de definições mais apuradas. A partir dos requisitos iniciais acertados entre o desenvolvedor e o usuário, realiza-se a construção de um projeto rápido, pautado nos aspectos visíveis do software, do ponto de vista do cliente.

Assim, o projeto rápido demanda um protótipo. Este, por sua vez, torna-se um importante instrumento de entendimento, ajuste e avaliação dos requisitos, através de sucessivas interações entre o desenvolvedor e o usuário.

De acordo com Pressman (2002), o modelo de prototipagem proporciona aos clientes uma primeira visão do sistema real e confere aos desenvolvedores a possibilidade de construir algo de forma imediata. A figura 3 a seguir ilustra o ciclo da prototipagem.

A assertiva de Pressman corresponde à realidade desde que o sistema a ser desenvolvido

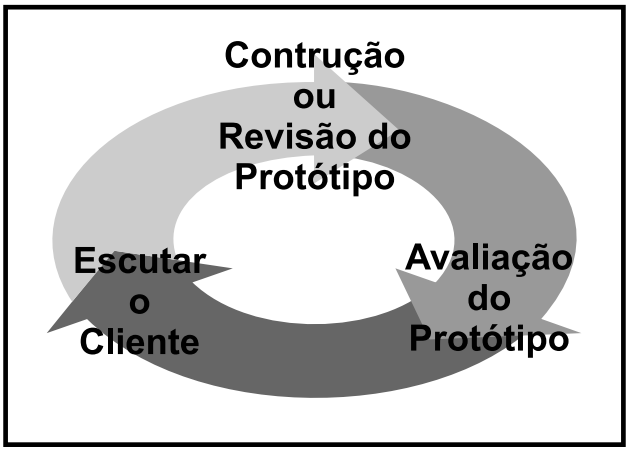

Figura 3 – O paradigma da prototipagem.
Fonte: Adaptado de Pressman (2002).

apresente pequenas dimensões, baixa complexidade e o escopo da solução bem definido. O avanço da tecnologia e o reconhecimento pelo usuário dos benefícios proporcionados pelo software levaram este produto a níveis elevados de complexidade e amplitude, de modo que se torna inviável um desenvolvimento fundamentado apenas no paradigma de prototipagem.

1.2.2 MODELO EVOLUCIONÁRIO

Neste tipo de modelo considera-se a natureza evolutiva do software, sobressaindo-se o aspecto da interatividade. Caracteriza-se por possibilitar aos engenheiros de software o desenvolvimento de versões cada vez mais completas do produto.

A idéia base consiste em elaborar uma implementação inicial, apresentar o resultado ao usuário, obter os comentários e fazer o aprimoramento por meio de versões incrementais até o desenvolvimento do sistema adequado. Tanto Pressman (2002) quanto Sommerville (2003) concebem o modelo evolucionário nas situações em que a especificação do software será obtida

gradativamente ou se necessita acomodar um produto que evolui com o tempo.

O modelo evolucionário apresenta-se como um tipo genérico subdividindo-se em especializações. Pressman (2002) classifica-as da seguinte forma: incremental, espiral, espiral ganha-ganha e concorrente. Sommerville (2003) adotou duas denominações: desenvolvimento exploratório e protótipos descartáveis.

Para fins deste artigo, torna-se relevante abordar o modelo incremental da *Rational,* segundo o qual se fundamenta o processo de desenvolvimento de software do CCA BR.

1.2.3 PROCESSO UNIFICADO

Nos dias atuais, os computadores apresentam-se cada vez mais poderosos, e seus usuários demandam grandes expectativas sobre a utilização dessas máquinas. Isto justifica a necessidade de desenvolvimento de software com características de grandeza e complexidade crescentes.

O advento da Internet contribuiu para elevar a sofisticação do software, na medida em que as informações passaram a trafegar em diversos formatos, tais como: texto, som, imagem e multimídia.

Observa-se que embora o hardware e o software tenham evoluído, muitos desenvolvedores continuam usando os mesmos métodos de décadas anteriores. Assim, evidencia-se um problema, pois sem atualizar os métodos não se consegue desenvolver os softwares complexos necessários na atualidade.

O processo de desenvolvimento de software unificado surgiu como uma resposta ao problema. Constitui-se num conjunto de atividades necessárias para transformar um requerimento de usuário em um sistema de software (JACOBSON; BOOCH; RUMBAUGH, 1999). Trata-se de um processo genérico que pode ser especializado para diversas categorias de sistemas de software.

1.2.4 MODELO INCREMENTAL RUP

O Processo Unificado da *Rational* consiste num modelo de processo de software criado pela *Rational Software Corporation* que herda todas as características do Processo Unificado citado no tópico anterior, incorporando práticas como a modelagem de negócios. O RUP descreve como desenvolver software efetivamente, usando técnicas testadas e aprovadas comercialmente, sendo particularmente aplicável ao desenvolvimento de grandes projetos de software.

Esse modelo apresenta a característica fundamental de ser baseado em componentes, ou seja, o software desenvolvido constitui-se em partes menores denominadas componentes de software que se comunicam através de interfaces bem definidas (KRUCHTEN, 2000). O padrão adotado para representação dos modelos é a linguagem de modelagem unificada (Unified Modeling Language - UML). A UML refere-se apenas a uma linguagem para representação e não constitui o processo de desenvolvimento propriamente dito.

1.3 CONCEITOS FUNDAMENTAIS DOS PROCESSOS UP E RUP

O Processo Unificado e, conseqüentemente, o RUP foram concebidos tomando como base três conceitos fundamentais: dirigido por *use case* (caso de uso), centrado na arquitetura, iterativo e incremental (JACOBSON; BOOCH; RUMBAUGH, 1999), os quais passam a ser descritos a seguir:

1- Dirigido por caso de uso: os casos de uso são aplicados na captura e definição dos requisitos funcionais do sistema de software, facilitando a comunicação e o entendimento entre os principais envolvidos no processo. São também utilizados para projetar os casos de teste. "Um *use case* é uma seqüência de ações que o sistema executa para fornecer um resultado de valor a um ator" (JACOBSON; BOOCH; RUMBAUGH, 1999, p.35).

Na verdade, quando os autores referem-se a uma seqüência de ações transparecem a idéia de se captar uma parte de uma funcionalidade do sistema. Quanto ao ator, entende-se como sendo aquele elemento externo ao sistema que vai interagir com o mesmo, podendo ser uma pessoa ou outro sistema:

2- Centrado na arquitetura: o RUP, desenvolvem-se os casos de uso e a arquitetura em paralelo. O conceito de arquitetura engloba os aspectos mais relevantes do sistema, tanto os estáticos quanto os dinâmicos.

3- Iterativo e Incremental: o RUP utiliza pequenos ciclos de projeto (mini-projetos) os quais correspondem a uma iteração e que resultam em um incremento no software.

As iterações encontram-se vinculadas às fases e referem-se aos passos no fluxo de trabalho e incrementos para a evolução do produto.

1.4 FASES DO RUP

Considerando a característica incremental do processo, cada acréscimo de atividade realizada ocorre por meio de quatro fases: *inception* (iniciação), *elaboration* (elaboração), *construction* (construção) e *transiction* (transição), denominadas ciclo de desenvolvimento (KRUCHTEN, 2000). Após a transição, o produto pode voltar a percorrer todo o ciclo, constituindo uma evolução.

Descrevem-se a seguir as fases do modelo:

1- Iniciação: fase de compreensão do problema e da tecnologia por intermédio da definição dos casos de uso mais críticos. No final desta fase, deve-se ter determinado o escopo do produto, os riscos e ter demonstrado que o projeto é viável do ponto de vista do negócio da organização;

2 - Elaboração: fase de descrição da arquitetura do *software*, na qual se capturam os requisitos que mais impactam na arquitetura, em forma de casos de uso. No final da fase de elaboração deve ser possível estimar custos, elaborar o cronograma e o plano de construção;

3 – Construção: fase na qual o *software* é construído e preparado para a transição para os usuários. Além do código, propriamente dito, também são produzidos os casos de teste e a documentação;

4 – Transição: fase de treinamento dos usuários e transição do produto para utilização.

1.5 PRINCIPAIS FLUXOS DE TRABALHO DE PROCESSO DO RUP

Segundo Kruchten (2000), cada uma das quatro fases do RUP divide-se, adicionalmente, em iterações e finaliza-se com um ponto de checagem que verifica se os objetivos daquela fase foram alcançados. Organiza-se toda iteração em termos de fluxos de trabalho de processo, que consistem em conjuntos de atividades realizadas por responsáveis que produzem artefatos (documento, modelos gráficos, etc.), conforme ilustrado na figura 4.

A figura 4 mostra a estrutura de processos do RUP. Observam-se duas dimensões do processo.

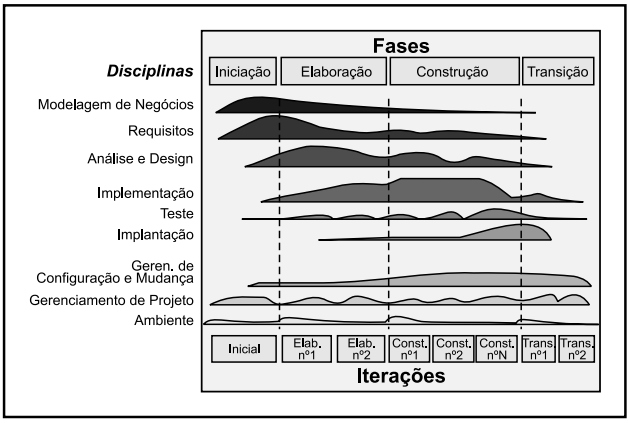

Figura 4 – Estrutura de processos do RUP.
Fonte: Adaptado de Kruchten (2000).

Uma dimensão corresponde ao eixo horizontal, representando o tempo e como os aspectos do ciclo de vida se desdobram. A outra dimensão está representada pelo eixo vertical e refere-se aos fluxos essenciais do processo, em que há uma classificação das atividades por natureza.

Para compreender melhor o conjunto dessas atividades, torna-se relevante listar os principais objetivos de cada fluxo essencial do processo RUP, de acordo com Kruchten (2000). Uma breve análise comparativa de alguns desses fluxos, ante ao Processo Unificado, acompanha as descrições a seguir:

1- Modelagem de Negócio (Bussines Modeling): no Processo Unificado (JACOBSON; BOOCH; RUMBAUGH, 1999) não havia o fluxo de trabalho de modelagem de negócios, partindo-se diretamente para o fluxo de trabalho de requisitos. Esse fluxo provê um entendimento comum entre os envolvidos com poder de definir os rumos do sistema, acerca dos quais os processos de negócio devem ser apoiados. A modelagem dos processos de negócio é feita através dos casos de uso de negócio:

b) Requisitos (Requirements)

Neste fluxo, busca-se capturar os requisitos que serão atendidos pelo produto de *software*. Nas fases de iniciação e elaboração, a ênfase será maior neste fluxo de trabalho de requisitos, pois o objetivo

dessas fases é o entendimento e a delimitação do escopo do produto de software;

c) Análise e Projeto (Analysis & Design)

No Processo Unificado (JACOBSON; BOOCH; RUMBAUGH, 1999), os fluxos de trabalho de análise e projeto apareciam separadamente, ressaltando a importância de se efetuar o modelo de análise. Apresenta como objetivo compreender mais precisamente os casos de uso definidos no fluxo de trabalho de requisitos, produzindo um modelo que deverá estar detalhado e adequado ao ambiente de implementação. Esse fluxo de trabalho será bastante utilizado na fase de elaboração e durante o início da fase de construção;

d) Implementação (Implementation)

O propósito desse fluxo consiste na organização do código no sentido de programar os subsistemas. Os componentes do software são implementados e testados ainda de como unidades isoladas, posteriormente se integram os códigos produzidos;

e) Teste (Test)

Neste ponto, o esforço concentra-se em analisar, por meio de testes, se os requisitos foram atendidos e contribuir para que os defeitos sejam removidos antes da implantação.

Os modelos de testes são criados para descrever como os testes serão realizados. Sua ênfase será maior no final da fase de construção e no início da fase de transição;

f) Entrega (Deployment)

As atividades deste fluxo objetivam produzir versões parciais do produto e entregá-los aos usuários finais. Isto pode incluir atividades de beta-teste, migração de dados ou software existente e aceitação formal.

Percebe-se que os fluxos essenciais do RUP apresentam uma evolução a partir do Processo Unificado. Ao contemplar a modelagem de negócio, o RUP possibilita uma compreensão conjunta entre as pessoas que detêm o poder de definir as regras contextuais do sistema. Já a unificação dos fluxos de análise e projeto estabelece um forte vínculo entre os requisitos e a implementação, provendo um modelo de construção de código consistente e adequado à realidade do ambiente tecnológico da organização desenvolvedora de software.

2 SISTEMAS DE INFORMAÇÃO

Segundo Polloni (2000), sistema de informação diz respeito a qualquer sistema utilizado para fornecer informações, independente do fim a que se propõe. Destaca-se aqui a questão do provimento da informação ainda sem o foco da área que pretende apoiar.

De acordo com Laudon (2004), a definição de sistema de informação, do ponto de vista técnico, refere-se a um conjunto de componentes inter-relacionados que recupera, processa, armazena e distribui informações destinadas a apoiar a tomada de decisão, coordenação e controle de uma organização. O conceito assume, portanto, um caráter generalista no tratamento das informações, tornando-se especialista quanto ao seu propósito.

Atualmente, os sistemas de informação são, quase sem exceção, baseados no computador e apóiam as funções gerenciais e de tomada de decisão. Podem ser vistos, tecnicamente, como um conjunto de programas e de estruturas de dados. Destaca-se, portanto, o conceito de Laudon tanto pela abordagem técnica quanto pelo foco na aplicação do software, caracterizando uma idéia mais completa.

No universo desses sistemas, torna-se relevante diferenciar o significado dos termos dado e informação. Em Laudon (2004), verifica-se que os dados são fatos brutos, representando eventos que ocorrem nas organizações ou no ambiente físico. Ao passo que a informação se traduz em dados apresentados numa forma significativa e útil para os seres humanos.

Uma diferença fundamental entre dado e informação é que o primeiro é puramente simbólico enquanto que o segundo tem significado.

A figura 5 demonstra as funções de um sistema de informação em que dados brutos registrados por uma caixa de supermercado podem ser processados e organizados de modo a produzir informações úteis, exemplificando a relação entre dado e informação.

Os sistemas de informação podem conter informações sobre pessoas, locais e coisas

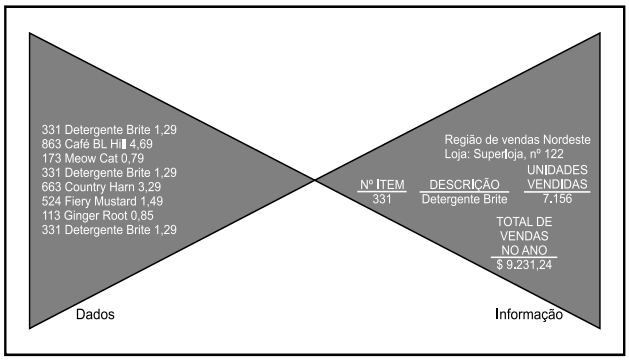

Figura 5 – Dados e informação.
Fonte: Adaptado de LAUDON (2004).

significativas para a organização ou para o ambiente que a cerca. São resultados obtidos a partir do arranjo dos dados de forma que se possa entendê-los e usá-los.

Para produzir as informações de que as organizações necessitam para tomar decisões, controlar operações, analisar problemas e criar novos produtos e serviços, Laudon (2004) estabelece três atividades: entrada, processamento e saída.

A entrada coleta os fatos brutos (dados), o processamento converte os dados em uma forma mais significativa e a saída transfere as informações às pessoas.

Além dessas atividades, os sistemas de informação requerem um *feedback,* que é a saída que volta a determinadas pessoas da organização para ajudá-los a avaliar ou corrigir a entrada.

A seguir, a figura 6 representa as três atividades de um sistema de informação.

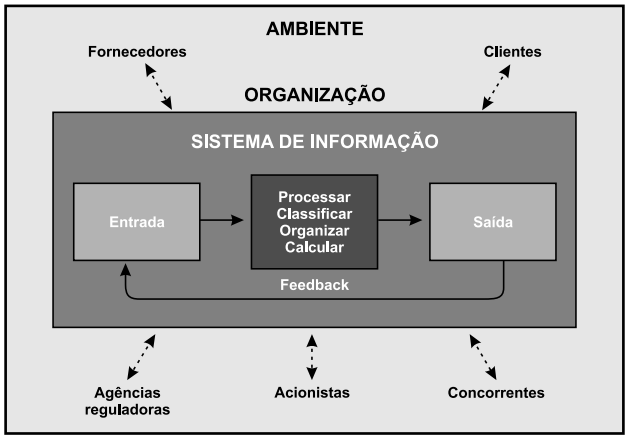

Figura 6 - Funções de um Sistema de Informação.
Fonte: Adaptado de LAUDON (2004, p. 8).

2.1 SISTEMAS DE INFORMAÇÃO NAS ORGANIZAÇÕES

Nas modernas organizações, os administradores não podem desconhecer os sistemas de informação porque estes desempenham um papel fundamental nas suas funções, afetando diretamente o modo como realizam o planejamento, a execução e o controle das atividades, em prol de uma estratégia com vistas à obtenção de produtos e/ou serviços.

A importância desse papel revela-se numa crescente interdependência entre a estratégia, regras e processos, de um lado, e programas, equipamentos, banco de dados e telecomunicações, de outro. A figura 7 modela a relação de dependência entre a organização e os sistemas de informação.

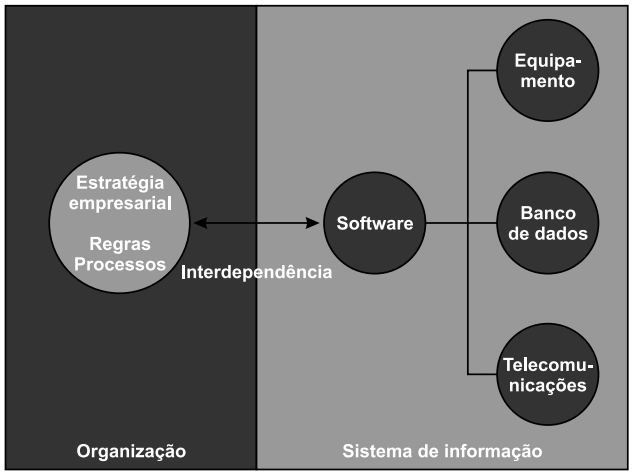

Figura 7 - Interdependência entre organizações e sistemas de informação.
Fonte: LAUDON (2004, p. 16).

Essa interdependência se revela na perspectiva de que mudanças no lado organizacional implicam cada vez mais em alterações nos sistemas de informação, visando à manutenção do alinhamento estratégico.

Com isso, os sistemas podem funcionar como uma limitação para as organizações. Aquilo que a instituição gostaria de realizar, muitas vezes, depende das possibilidades oferecidas pelos seus sistemas.

Ao longo do tempo, os sistemas de informação evoluíram do papel de fomentadores de mudanças tecnológicas, relativamente fáceis de serem obtidas, para atuarem como protagonistas que afetam o controle e o comportamento gerencial,

repercutindo em atividades institucionais como, por exemplo, a tomada de decisão.

A evolução dos sistemas e sua influência na vida organizacional possibilitaram uma classificação que mantém estreita ligação entre a finalidade que se destinam e o perfil profissional dos usuários no ambiente de trabalho.

2.2 TIPOS DE SISTEMAS DE INFORMAÇÃO

Existem vários tipos de sistema de informação destinados a apoiar os distintos níveis organizacionais, funções e processos de negócios.

Segundo Anthony (1965), os sistemas são estruturados para atender aos interesses da organização em seus diversos níveis. Para tanto, classificou a organização em nível estratégico, gerencial, de conhecimento e operacional, bem como em áreas funcionais: vendas e *marketing*, fabricação, finanças, contabilidade e recursos humanos.

A partir dessa premissa, Anthony (1965) define que há quatro tipos principais de sistemas de informação: sistemas do nível operacional, do nível de conhecimento, do nível gerencial e do nível estratégico. A figura 8 descreve os tipos de sistemas encontrados na organização e corroborados por Laudon (2004).

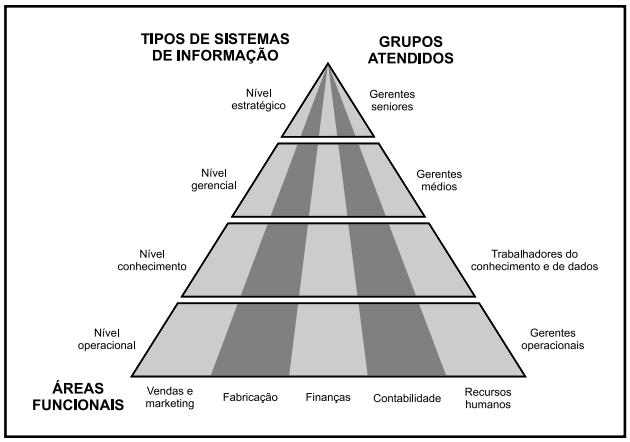

Figura 8 - Tipos de sistemas de informação.
Fonte: LAUDON (2004, p. 40).

Observa-se na figura 8 que as áreas funcionais são apoiadas pelos sistemas de informação, conforme a necessidade de cada nível organizacional.

Considerando o contexto deste trabalho, a ênfase recai sobre o nível gerencial. Os sistemas de informação do nível gerencial atendem às atividades de monitoração, controle, tomada de decisões e procedimentos administrativos dos gerentes médios.

Os sistemas de informação do nível gerencial têm a característica de produzir relatórios periódicos sobre as operações, em vez de informações instantâneas. A principal consulta destinada a esses sistemas consiste na indagação: *as coisas estão indo bem?*

De acordo com Keen e Morton (1978), alguns sistemas gerenciais apóiam a tomada de decisões não-rotineiras. Tendem a focar decisões menos estruturadas, nas quais as exigências de informação nem sempre são claras.

Nesse caso específico, o sistema de informação do nível gerencial, freqüentemente, o tipo de pergunta: *E se?* Por exemplo: qual seria o impacto sobre a disponibilidade de aeronaves, nos diversos esquadrões aéreos da Força Aérea Brasileira, se houvesse um aumento de 50% na dotação das horas de vôo para o ano de 2008?

As respostas para perguntas desse tipo quase sempre exigem novos dados de fora da organização, bem como dados internos, que não podem ser facilmente retirados dos aplicativos do nível operacional.

Os sistemas de informação do nível gerencial auxiliam, de uma maneira ou de outra, as pessoas na tomada de decisão. Na década de 70, Keen e Morton (1978) evidenciaram essa necessidade em decorrência de diversos fatores, tais como:

1. Competição cada vez maior entre as organizações;
2. Necessidade de informações rápidas para auxiliar no processo de tomada de decisão;
3. Disponibilidade de tecnologias de *hardware* e *software* para armazenar e buscar rapidamente as informações;
4. Possibilidade de armazenar o conhecimento e as experiências de especialistas em bases de conhecimentos;
5. Necessidade da informática apoiar o processo de planejamento estratégico empresarial.

Esses fatores contribuíram para que as organizações começassem a desenvolver sistemas

de informação que possibilitassem auxiliar no processo de tomada de decisão.

2.4 SISTEMAS DE APOIO À DECISÃO

Segundo Sprague e Watson (1991), qualquer sistema de informação que forneça fundamentos para auxílio à decisão é um SAD. A literatura demonstra que a afirmação é questionável, pois está restrita apenas à obtenção das informações, enquanto há autores que enfatizam também a contribuição desses sistemas com o processo decisório.

De acordo com Lucas (1990), o SAD é baseado em computador e auxilia o processo de tomada de decisão, utilizando dados e modelos para resolver problemas que não apresentam um procedimento prévio para se chegar a uma solução. Neste sentido, ajudam a interpretar o que ocorreu e a decidir sobre estratégias futuras para a organização.

Em Laudon (2004), identificam-se dois tipos específicos de sistemas de informação do nível gerencial: sistemas de informação gerencial (SIG) e sistemas de apoio à decisão (SAD), de acordo com a figura 9 apresentada a seguir.

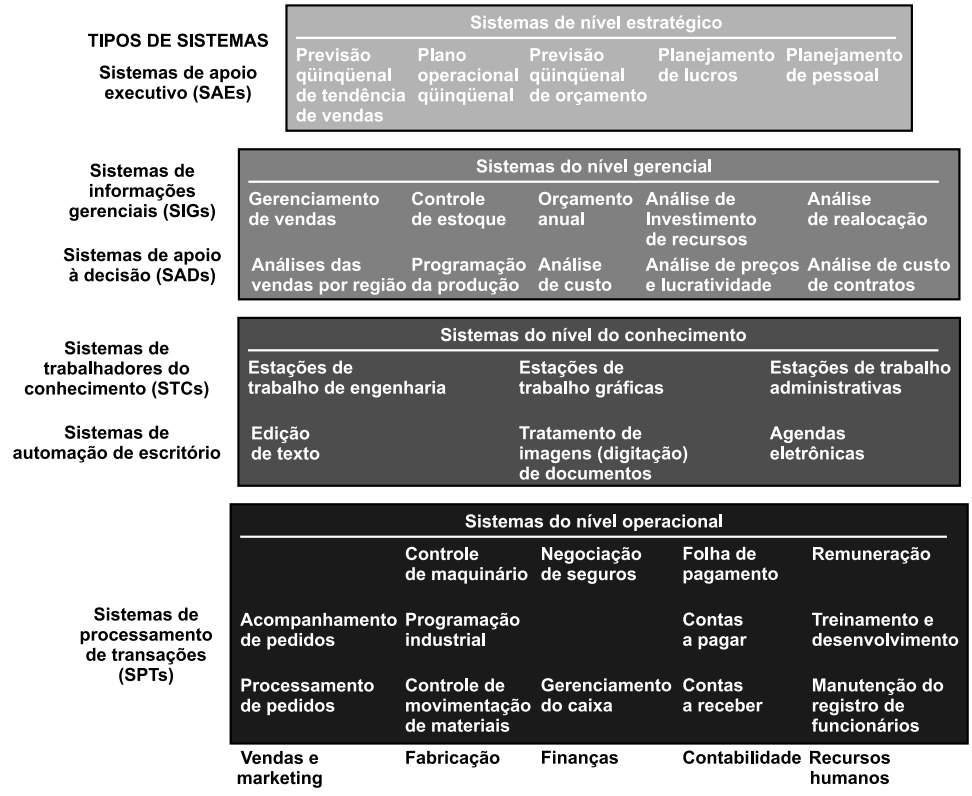

Figura 9 – Os seis tipos mais importantes de sistemas de informação.
Fonte: LAUDON (2004, p. 41).

Os SIG fornecem informações sobre o desempenho da organização para ajudar os gerentes a monitorá-la e controlá-a. Produzem relatórios fixos, ajustados periodicamente, com base em dados extraídos e resumidos dos sistemas do nível operacional, ou seja, sistemas de processamento de transações.

Os SAD oferecem novos conjuntos de capacitação para decisões não-rotineiras e controle do usuário. Abordam problemas em que os passos para a solução ainda não se encontram totalmente definidos.

Um SIG fornece aos gerentes relatórios baseados em fluxos rotineiros de dados e auxilia no controle geral da organização, ao passo que um SAD dá ênfase à mudança, flexibilidade e resposta rápida.

Com um SAD, o esforço é menor para ligar usuários a fluxos de informações estruturadas, e há uma ênfase correspondentemente maior em modelos, pressuposições, consultas específicas e apresentações gráficas.

A diferenciação entre SIG e SAD passa pelo entendimento sobre a classificação que os pesquisadores adotam para os tipos de decisão: estruturadas e não-estruturadas.

De acordo com Gorry e Morton (1971), as decisões não-estruturadas são aquelas em que o responsável por elas deve usar seu bom senso, sua capacidade de avaliação e sua perspicácia na definição do problema. São inusitadas, importantes, não-rotineiras e não há procedimentos bem entendidos ou predefinidos para serem executadas.

Em contrapartida, as decisões estruturadas são repetitivas, rotineiras e envolvem um procedimento predefinido, de modo que

não precisam ser tratadas a cada vez como se fossem novas.

Gorry e Scott-Morton (1971) acrescentam, ainda, que algumas decisões são consideradas semi-estruturadas, nas quais parte do problema tem uma resposta clara e precisa dada por um procedimento aceito.

Enquanto os SIG abordam problemas estruturados, os SAD provêem apoio à análise de problemas semi-estruturados e não-estruturados.

De acordo com Sprague e Watson (1991), as principais características dos SAD são evidenciadas pelos seguintes aspectos: os dados e os modelos devem ser organizados em função da decisão, flexibilidade e capacidade de adaptação às mudanças no ambiente e no estilo do responsável pela tomada de decisão; processamento interativo e interface com o usuário de fácil utilização.

Os autores destacam, ainda, que o fundamento de um bom SAD consiste na utilização de uma moderna tecnologia de banco de dados que possibilite transformar uma grande base de dados em fonte de conhecimento gerencial, elaborando um sistema que atue no sentido de agrupar informações que demonstrem alterações de padrões e tendências. A origem dessa base de dados provém dos sistemas transacionais e constituem a biblioteca que disponibilizará as respostas às questões gerenciais.

Sinteticamente, essa concepção considera que a arquitetura de um SAD engloba um planejamento de *hardware*, *software* e interface com o usuário que venha ao encontro das possibilidades da organização e da sua cultura. Para isso existem diversas preocupações relacionadas ao levantamento de requisitos, extração e armazenamento da base de dados, bem como o formato que essas informações serão disponibilizadas para o usuário.

CONCLUSÃO

O processo de desenvolvimento de *software* tem evoluído pautado na comunicação entre usuários e projetistas, bem como nos ciclos em que essas interações ocorrem, visando ao domínio do conhecimento necessário à obtenção do produto final: o sistema de informação.

Os modelos de processo de software enfatizam a busca pela qualidade e o baixo custo na consecução do produto software, o que significa estar em conformidade com os requisitos e a ponderação de investimentos para obtê-lo.

Assim, verifica-se que o grande desafio dos engenheiros de software recai sobre a concepção de um modelo que seja genérico o suficiente para abranger as diversas peculiaridades dos sistemas de informação sem, no entanto, infligir complexidade que resulte em custos elevados.

Neste artigo, abordou-se a perspectiva do CCA BR utilizar o modelo RUP para desenvolver SAD, em virtude da recente atribuição de prover soluções voltadas ao apoio à decisão da alta administração da Força Aérea Brasileira. A análise buscou verificar a possibilidade de adequação entre o processo existente e esse novo tipo de *software,* sendo descrita na seqüência.

Em termos de Engenharia de *Software* e alinhado com Pressman (2002), verificou-se que a busca pela qualidade não implica mudanças no processo RUP, uma vez que essa concepção é inerente ao modelo.

Do ponto de vista da camada processo, o RUP é adequado ao desenvolvimento de SAD, desde que alguns aspectos sejam observados e adaptados ao modelo, tais como:

a) Direcionar uma atenção especial à Fase de Iniciação na qual se procura compreender o problema e a tecnologia necessária para solucioná-lo, destacando-se a atividade de levantamento de requisitos;

b) Adotar uma metodologia de levantamento de requisitos que identifique as questões gerenciais, pois a abordagem de caso de uso do RUP pressupõe um procedimento prévio que conduzirá a uma resposta, algo inexistente quando se trata de problemas do nível gerencial;

c) Explorar a iteratividade do modelo RUP no sentido de obter protótipos que simulem as principais questões gerenciais levantadas, provendo agilidade na pronta-resposta ao usuário;

d) Orientar a arquitetura para a integração dos sistemas transacionais que comporão o SAD, por meio da utilização da moderna tecnologia de banco de dados.

e) Na Fase de Elaboração do RUP, descrever a arquitetura considerando que um SAD requer investimentos consideráveis de hardware e software especializado, bem como de consultoria e treinamento, tratando-se de um sistema de uso corporativo;

f) Implementar políticas de comunicação e acesso às bases de dados dos sistemas que fornecerão os dados primários ao SAD. Esse procedimento facilitará a Fase de Construção, pois promovem a participação dos gerentes desses sistemas e, por conseqüência, a sinergia requerida.

Verificou-se, também, que a Fase de Implementação, mantém-se no propósito de construir os códigos do *software*, porém com o maior esforço voltado à extração, tratamento e carga de dados.

As Fases de Teste e Entrega englobam práticas amplamente reconhecidas e sedimentadas pela Engenharia de *Software*, tornando-se desnecessário qualquer adaptação no RUP.

Finalmente, constata-se que o desenvolvimento de SAD, de acordo com o modelo RUP apresenta-se factível, desde que os fatores analisados sejam considerados na metodologia, a fim de prover a conformidade necessária do Processo de Software às características desse tipo de sistema de informação, sobretudo no que tange ao levantamento de requisitos e à tecnologia de banco de dados.

REFERÊNCIAS

ANTHONY, R. N. **Planning and control systems:** a framework for analysis. Cambridge, MA: Harvard University Press, 1965.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. Subdepartamento de Tecnologia da Informação. **Diretrizes específicas para os centros de computação da aeronáutica (CCA)**: NSCA 7-6. Brasília, DF, 2005.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2002.

GORRY, G. A.; MORTON, M. S. S. **A framework for management.** Sloan Management Review 40, n. 4, verão 1998.

IBM Rational unified process: data sheet. Disponível em: <http://www3.software.ibm.com/ibmdl/pub/software/rational/web/datasheets/version6/rup.pdf>. Acesso em 21 jun. 2007.

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS - IEEE, **Standards Collection:** Software Engineering, IEEE Standard 610.12 – 1990. IEEE, 1993.

JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. **The unified Software development process**. Boston: Addison-Wesley, 1999.

KEEN, P. G. W.; MORTON, M. S. **Decision support systems:** an organizational perspective**.** Reading, Ma: Addison Wesley, 1978.

KRUCHTEN, P. **The Rational Unified Process:** an Introduction. 2nd ed. Boston: Addison-Wesley, 2000.

LAUDON, K. C.; J. P. **Sistemas de informação gerenciais:** administrando a empresa digital. 5.ed. São Paulo: Prentice Hall, 2004.

LUCAS, C. H. J. **Information systems concepts for management.** McGraw-Hill International, 1990.

POLLONI, E. G. F. **Administrando sistemas de informações.** São Paulo: Futura, 2000.

PRESSMAN, R. S. **Engenharia de software**. São Paulo: McGraw-Hill, 2002.

RACCOON, L. B. S. The caos model and the caos life cycle**. ACM software engineering notes,** v. 20, n. I, January, 1995.

SOMMERVILLE, I. **Engenharia de software.** São Paulo: Addison-Wesley, 2003.

SPRAGUE, R. H.; WATSON, H. J. **Sistemas de apoio à decisão.** Campus. 1991.