SERBAC: Uma Estratégia para Definição de Requisitos

Antônio de Pádua Albuquerque Oliveira
PETROBRÁS - SERINF / DISUS / SETEC
Av. Chile, 65 sala 1556 - Fax:(021) 220-2686
20035-900 , Rio de Janeiro , RJ - Brasil
Telefone:(021) 534-4253

Julio Cesar Sampaio do Prado Leite
Depto. de Informática, PUC - Rio
R. Marquês de São Vicente 225
22453 - Rio de Janeiro , RJ - Brasil
e-mail:julio@inf.puc-rio.br

RESUMO

SERBAC é uma estratégia para definição de requisitos apoiada por uma ferramenta que registra requisitos. A estratégia SERBAC utiliza várias propostas da literatura para montar um esquema de representação de requisitos baseado em linguagem natural. Utiliza a idéia de vocabulário da aplicação, a idéia de que as ações dos atores de uma organização são o ponto central do estudo da informação, a idéia de que um sistema responde a eventos externos e a idéia que um sistema pode ser definido por seus relacionamentos, por suas entradas e por suas saídas. Essas idéias de diferentes autores foram agrupadas e sistematizadas de forma a produzir um esquema onde os requisitos são definidos de uma maneira padrão. A estratégia SERBAC é apoiada por um software e foi avaliada em um caso real.

Palavras-chaves: *Requisitos, Engenharia de Requisitos, Sistemas de Informação, Engenharia de Software.*

ABSTRACT

SERBAC is a Requirement Definition process suported by a software tool. The process uses the Language Extended Lexicon as the main source of "External Requirements" of an Information System or of a Software System. The External System Requirements are originated from: the Concret Actions realized by the system clients, the External Events answered, and also from Inputs and Outpus of the system. In the work, we discuss the chief problems that the requirements engineer faces during the Elicitation and during the System Requirement Modeling. We also presented a taxonomy for requirements and a survey of the most frequent mistakes in the Systems Requirements. SERBAC was used in the External Requirements Elicitation of the AVP - Autorizações de Viagens no Pais, a real system for trip management.

Key words: *Requirements, Requirement Engineering, Information System, Software Engineering.*

1. MOTIVAÇÃO

Existem inúmeros casos de fracassos no desenvolvimento de sistemas de software. Um forum onde normalmente encontramos relatos sobre esses fracassos é a coluna ***Risks*** da ***Communications of the ACM*** da revista do grupo de interesse em engenharia de software - ***Software Engineering Notes***. Um dos exemplos é o caso do "B of A". O "Bank of América" gastou 23 milhões de dólares no desenvolvimento inicial (5 anos) do "MasterNet", um novo e confiável sistema computadorizado de contabilidade. Após abandonar o sistema antigo, eles gastaram mais 60 milhões de dólares tentando fazer o novo sistema funcionar corretamente e no final desistiram da idéia, voltando ao sistema original.

Estes relatos dão conta de que muitos sistemas fracassam porque não atendem seus clientes e, em conseqüência, demandam recursos de manutenção que não são previstos antecipadamente. Grande parte dos fracassos podem ser atribuídos a um trabalho de definição do sistema mal feito. Porém, a definição do sistema depende da definição de seus requisitos.

Apesar de acreditarmos que a Definição de Requisitos é uma atividade perene (vide figura seguinte), é fato que um esforço inicial é necessário para deixar claro e documentado o desejo daqueles que demandam o sistema. Mas, quando um novo requisito surge, o engenheiro de software tem a tendência a considerá-lo imediatamente, fazendo sua incorporação ao modelo de dados ou ao esquema funcional.

Entretanto essas ferramentas não são adequadas à formulação de problema, porque elas antecipam uma etapa de solução. Portanto achamos que é necessário se ter um processo de definição de requisitos para descobrir e reduzir, ou eliminar, os erros e omissões em requisitos inerentes a atividade.

Por outro lado, os desenvolvedores do sistema devem fazer revisões para verificar se os requisitos definidos e formalmente aceitos pelos clientes foram devidamente incorporados aos modelos construídos. Esses modelos, ou especificações, são no nosso entender linguagem própria dos profissionais de sistemas e não de seus clientes. Acreditamos que os requisitos (documento de requisitos) devem ser objeto de validação pelos clientes e a memória da origem e da história dos requisitos do sistema.

O SERBAC é uma proposta que torna os requisitos um objeto passível de validação pelo cliente e auditável do ponto de vista do registro da história da evolução dos requisitos.

2. A ENGENHARIA DE REQUISITOS

Nesta seção fazemos uma revisão de vários dos pontos importantes da área de engenharia de requisitos, cujo objetivo principal é a de propor métodos, técnicas e ferramentas para apoiar a tarefa de definição de requisitos. Neste apanhado de características procuramos ressaltar nosso entendimento sobre requisitos e apontar aspectos que nos levaram a montar uma lista de tipos de defeitos em requisitos. Concluímos, a seção, apresentando uma taxonomia para requisitos e ressaltamos os chamados **"Requisitos Externos"**, para os quais o SERBAC está direcionado.

2.1 ALGUMAS CARACTERÍSTICAS DA DEFINIÇÃO DE REQUISITOS

Requisitos: Acreditamos que a melhor definição de requisito é a seguinte: ***"Condição necessária para a obtenção de certo objetivo, ou para preenchimento de certo objetivo."*** Portanto deixamos claro que o requisito não é o objetivo do sistema, mas o meio para se alcançar o objetivo, e que somente através do cumprimento dos requisitos chega-se a atingir o objetivo do sistema.

Existe uma diferença entre o que chamamos requisitos e o que normalmente outros autores chamam de especificação de requisitos. Para nós, requisitos são expressões dos clientes sobre a funcionalidade e sobre as restrições do sistema a ser construído. Essas expressões são descritas fundamentalmente como sentenças em linguagem natural formatada [12]. O SERBAC procura organizar dentro de uma sistematização bem definida o documento que conterá estas sentenças.

Contexto: É importante deixar claro que estamos lidando com sistemas, isto é estamos descrevendo aspectos da engenharia de requisitos relacionados a sistemas. Portanto nosso alvo pode ser tanto um sistema de informação ou um sistema de software. É importante termos em mente que um sistema de informação quando utiliza processos automatizados tem como subsistema um sistema de software. Por isso sempre que tratamos de sistemas de apoio à organização temos que lidar em dois niveis, um na definição do sistema de informação e outro na definição do sistema de software que faz parte desse sistema de informação. Esta diferença entre sistema de informação e sistema de software é um dos graves problemas no processo de construção de sistemas de apoio para organizações [11].

UdI - Universo de Informações: - é o contexto geral no qual o software deverá ser desenvolvido. O Universo de Informações inclui todas as fontes de informação e todas as pessoas relacionadas ao software. Estas pessoas são também conhecidas como atores desse universo. O Universo de Informações é a realidade circunstanciada pelo conjunto de objetivos definidos pelos que demandam o software [11].

Clientes: Talvez o pior erro no desenvolvimento de sistemas seja deixar o cliente fora do processo. Às vezes ocorre que o cliente mais indicado, o melhor representante, não participa inteiramente do processo. Esse tipo de erro de identificação do verdadeiro cliente é provocado pela definição prematura da abrangência do sistema. A abrangência, os clientes e os requisitos do sistema, estão sempre juntos, eles são a própria definição do sistema. Burstin [2] detalhou estes aspectos e propôs um esquema que procura mapear os clientes de um sistema, através do que ele chamou de árvore de usuários abstratos.

O Nome do Sistema: Coisas têm nome. Portanto devemos dar importância a escolha adequada de um nome para o sistema para o qual estamos procurando definir os requisitos [7]. O nome do sistema que às vezes surge sem se saber como, não é tratado com a devida importância. Mesmo antes do projeto ser iniciado, dos objetivos estarem definidos, o nome do sistema já existe. Este nome dado prematuramente, quando não é alterado, no caso de ser inadequado, pode influenciar negativamente nos resultados, devido ao entendimento ambíguo do problema. O nome do sistema pode e deve ser alterado pelos seus clientes e engenheiros de software ao longo do seu desenvolvimento.

Relacionamos a seguir alguns erros que encontramos em nomes de sistemas:

(a) mencionar somente uma parte dos objetivos;
(b) dar privilégio ao componente computadorizado;
(c) receber o nome de um componente (normalmente o de algum arquivo);
(d) possuir nome muito abstrato, o qual dá idéia de funcionalidade infinita.

O nome reflete a expectativa dos clientes, exerce influência sobre o trabalho dos desenvolvedores, é fundamental para a percepção global do sistema.

A Validação: Será que o cliente deve validar a especificação do sistema ? O documento que define os detalhes de um novo sistema, chamado de "Especificação do Sistema", é muitas vezes impossível de ser compreendido pelos clientes, por causa da extensão e devido aos conceitos técnicos a ele incorporados. Então concluímos: o que interessa ao cliente não é a especificação do sistema, isto é importante para o projetista. O que o cliente quer é o sistema, através de seus produtos e requisitos, e não o modelo ou a especificação detalhada do sistema. Para que a validação dos requisitos de um sistema ocorra com sucesso, é necessário que os requisitos e suas definições sejam apresentadas com uma comunicação clara e na linguagem do cliente. Além da clareza, os clientes devem encontrar facilidade para realizar a validação.

Elicitação: Elicitar é um neologismo que procura justamente enfatizar um aspecto que vem sendo constantemente negligenciado no processo de definição de sistemas. Elicitar significa descobrir, tornar explícito, obter o máximo de informações para o conhecimento do objeto em questão.

A engenharia de software e a área de sistemas de informação têm dado pouca atenção as técnicas que podem ajudar a tarefa de elicitação. Essas técnicas só agora começam a freqüentar o mundo dos engenheiros de software, já que fundamentalmente são oriundas das ciências sociais. Essas técnicas são aquelas que exigem uma iteração forte entre clientes e desenvolvedores como: entrevistas, questionários, etnografia, seleção de cartões e reuniões [8], [7], [10]. Além da pouca difusão dessas técnicas elas requerem muito trabalho por parte do engenheiro de software. Tradicionalmente, os profissionais de sistemas partem logo para a modelagem do sistema usando ferramentas gráficas, que não são a forma adequada de se representar grande parte dos requisitos.

Para se determinar os requisitos de um sistema, algumas variáveis como: a estabilidade dos requisitos, a capacidade dos clientes em expressar os requisitos e a capacidade dos engenheiros de software em elicitar os requisitos que influenciam no processo devem ser avaliadas. Segundo Davis [5], estas influências quando negativas devem ser contornadas e diminuídas com a adoção de uma estratégia adequada a incerteza do processo.

As Ferramentas: Existem muitas ferramentas e métodos para a engenharia de software e para a engenharia de informação. Muitos recursos foram encaminhados para a área de ferramentas "CASE" pois o mito é de que o software se tornou um fator limitante na construção dos sistemas baseados em computador, segundo Pressman [17]. Observamos muitas facilidades em ferramentas para modelagem do sistema e para a geração de código, tais como linguagens de quarta geração, editores de diagramas e para apoio a métodos como os de orientação a objetos, porém são reduzidos os avanços no campo da Engenharia de Requisitos.

2.2 DEFEITOS EM REQUISITOS

Para se reduzir os erros em requisitos é necessário conhecer quais são as origens dos problemas em requisitos, para então podermos enfrentá-los com métodos e técnicas adequadas.

O conjunto de requisitos produzidos pelo processo de elicitação nunca estará livre de erros. Como afirma Davis [5], existem 3 grandes dificuldades para se obter os requisitos de informação corretos e completos para os sistemas. A primeira dificuldade é a limitação dos seres humanos como processadores da informação e solucionadores de problemas. A segunda é devida a grande variedade e complexidade dos requisitos de S.I.. E a terceira, é devida aos padrões complexos para a comunicação entre os usuários e engenheiros de software na definição dos requisitos.

Os erros mais perigosos cometidos no Desenvolvimento de Sistemas são ordenados a seguir.

1) ignorar um grupo de clientes;
2) ignorar um único cliente;
3) omitir um grupo de requisitos;
4) permitir inconsistências entre grupos de requisitos;
5) aceitar requisito inadequado ou não pertinente;
6) aceitar requisito incorreto ou indefinido;
7) aceitar requisito impreciso;
8) aceitar requisito ambíguo ou inconsistente.

Para a detecção dos defeitos mencionados acima, não existe uma única técnica mais indicada. Davis [5] recomenda que não se deve usar uma única abordagem para todos os sistemas. Indo além, recomendamos que não se deve usar uma mesma técnica para se abordar todos os tipos possíveis de erros em requisitos em um mesmo sistema.

As técnicas são importantes pela característica de assegurar o entendimento do problema ou defeito. Com o uso de técnicas adequadas poderemos estabelecer um acordo explícito de aceitação de cada imposição existente, de cada hipótese formulada e de cada escolha decidida entre os atores do UdI. A lista acima é uma especialização da lista sugerida por Burstin [2] com os erros de maior importância em requisitos. Esses erros estão detalhados em Oliveira [15].

2.3 TAXONOMIA DE REQUISITOS

Identificamos quatro referenciais para a representação dos requisitos, que são: ***(1) quanto ao contexto***, que dá a perspectiva "física": ***(2) quanto à abstração***, que dá a perspectiva do genérico para o detalhado, ***(3) quanto à fase de desenvolvimento ou quanto à modelagem***, que se preocupa com a incorporação do requisito aos modelos do sistema e ***(4) quanto à validação***, que se ocupa da validação dos requisitos pelos clientes.

Acreditamos que os requisitos quanto ao contexto do sistema devam ser formulados em dois grupos: **requisitos externos**, que se decompõe em ***requisitos de utilidade e requisitos impostos ou restrições externas***, e **requisitos internos**, que se decompõe em ***requisitos funcionais e restrições operacionais.***

Da mesma forma que a modelagem de um sistema utiliza o artifício da abstração, a facilidade de se agrupar objetos em "famílias", para domínio da complexidade, a elicitação dos requisitos deve fazer uso dessa importante capacidade do raciocínio humano. Os requisitos devem ser gradualmente enriquecidos em detalhes, até que se possa compreênde-los com toda precisão e rigor. Assim temos dois outros grupos de requisitos que são os **requisitos abstratos** e os **requisitos unitários ou elementares**.

Exemplo: "Cobrar as faturas atrasadas segundo a política de cobrança.
Para que o sistema apoie a cobrança de faturas atrasadas é necessário que o sistema possa reconhecer uma fatura atrasada. Então, o que é uma fatura atrasada ? É preciso também que os detalhes da política de cobrança fiquem definidos para se ter uma saída do sistema que apoie de maneira eficaz a ação do cliente.

Um requisito pode ser obtido a qualquer momento no ciclo de desenvolvimento do sistema. Quando o cliente expressa alguma necessidade que implicaria em um novo requisito pertinente à etapa de construção do sistema (por exemplo: um requisito de performance ou uma restrição externa, tipo a flexibilidade da alteração de um item antes de uma simulação). O engenheiro de software não poderia pedir ao cliente guardar a sua manifestação para a fase de projeto, ou de implementação, só porque, somente lá, o requisito poderá ser incorporado ao modelo construído, ou mesmo agregado diretamente ao sistema. Então temos os **requisitos incorporados** e os **requisitos não incorporados**.

Exemplo: "A resposta à consulta ao saldo em caixa deve ser imediata"

Quando um requisito do sistema surge como uma restrição ou como um atributo de um produto, ou mesmo uma decisão de como o sistema deve operar, é necessário que o engenheiro de software providencie o seu registro para que oportunamente este requisito seja analisado e validado por todos os clientes envolvidos. A importância desta classificação vem do fato que os requisitos surgem ao longo de todo o processo de elicitação e também durante o desenvolvimento do sistema, e por isso, antes de se incorporar ao sistema qualquer requisito, ele precisa ser discutido e aceito pelos clientes. A partir dessa observação concluímos que podemos ter **requisitos validados** e **requisitos não validados** pelos clientes.

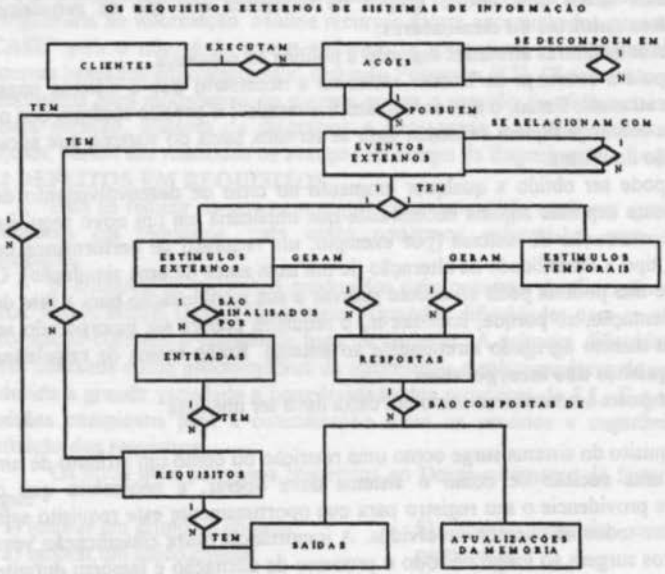
2.4 MELHORANDO A QUALIDADE DOS REQUISITOS

A primeira medida para se encaminhar uma solução é: **"Todos os requisitos devem ser registrados"**. Como os requisitos são provenientes de imposições, de hipóteses e também de escolhas; como o trabalho envolve muita interação do cliente com o engenheiro de software, como as vezes também pode levar muito tempo, assumimos que muitos requisitos são perdidos ao longo do processo e pressupomos que para se tentar chegar a um conjunto de requisitos pelo menos concretos e palpáveis, é mandatório que todos os requisitos que estão fluindo sejam formulados concretamente para que no final do processo possam ser transformados em acordos escritos.

Os requisitos em sua maioria não existem no inicio do desenvolvimento e somente um processo interativo e organizado entre os atores do UdI é que pode fazer surgir este conjunto de requisitos. A qualidade do sistema a ser desenvolvido deve ser explicitada, para que além dos requisitos, também as possíveis alternativas de concepção não sejam perdidas.

SERBAC é uma estratégia que pretende atacar alguns desses problemas a que nos referimos, colocando em evidência os Requisitos Externos do Sistema. Pensamos que os requisitos de utilidade e as restrições externas do sistema não são questionados e debatidos suficientemente nas fases iniciais do desenvolvimento, porque a maioria das metodologias não possuem uma âncora que permita um referencial sem ambigüidade para as questões básicas da Definição de Requisitos: Que fatos devem ser coletados do UdI; comunicados e validados pelos clientes para a posterior modelagem pelos engenheiros de software ?

Consideramos que os **Requisitos Externos do Sistema**: formulados pelas ações concretas dos clientes, pelos eventos externos, pelas saídas, e pelas entradas do sistema, são um meio eficaz para auxiliar o problema da falta de qualidade, principalmente por sua característica de uso da linguagem natural.



Somente de posse dos **Requisitos Externos** (requisitos de utilidade e das restrições externas), é que os desenvolvedores do sistema poderão definir os requisitos internos decorrentes da avaliação de alternativas de operacionalidade.

A figura anterior mostra um modelo entidades-relacionamentos [4] aplicado aos requisitos externos de um sistema, que fundamentam nossa estratégia. A interdependência entre os requisitos está presente na figura e é comentada a seguir.

Os Requisitos Externos do Sistema estão modelados pelas entidades: AÇÕES, EVENTOS EXTERNOS, ENTRADAS e SAÍDAS, que estão relacionadas direta ou indiretamente com a entidade REQUISITOS, que modela os requisitos de detalhe dos Requisitos Externos. As demais entidades representadas somente têm o papel de completar logicamente o modelo porque não fazem parte do método SERBAC.

2.4.1 Requisitos de Utilidade

Descrevemos abaixo os conceitos básicos relacionados aos requisitos de utilidade. Em seguida detalharemos de que maneira as restrições externas são associadas a esses requisitos de utilidade.

Exemplo de repasse de informações: ***FORNECER informações sobre o estoque de matéria-prima corrosiva ao Setor de Segurança Industrial.***
Requisito: Sempre que houver alteração importante deste estoque.

Restrição - Requisitos dos Eventos Externos: Definem, e qualificam com precisão, quais são as interações do Sistema com seu UdI, para atendimento das Ações Concretas.
Exemplo: ***Fornecedor faz a cobrança de pedidos de matéria-prima.***
Requisito: O vencimento deve estar coerente com a política de pagamentos.

Obs.: O evento deve possuir um requisito de detalhe para que o sistema reconheça somente as cobranças de pedidos efetivados e além disso, eles devem ter sido atendidos pelo fornecedor. O ciclo ação-eventos, que possui uma conotação semântica semelhante ao "passo da entidade-estrutura" [9], reflete o encadeamento destas restrições, onde cada evento pode ter seu requisito.

Exemplo: Ação = PAGAR, sem atraso, as faturas de fornecedores de matéria-prima da matriz.

Evento 1: ***Setor de Compras faz pedido de matéria-prima.***
requisito: Considerando as restrições de estoque do produto.

Evento 2: ***Fornecedor entrega pedido de matéria-prima.***
requisito: A entrega deve corresponder ao pedido formulado.

Evento 3: ***Fornecedor faz a cobrança de pedidos de matéria-prima.***
requisito: Contém entregas de pedidos ainda não pagas.

Evento 4: ***É dia de pagar fatura de fornecedor.***
requisito: Provoca uma prévia semanal e uma outra com 48 horas.

Restrição - Requisitos das Saídas:
Contém os detalhes relevantes para o apoio as Ações Concretas.
Exemplo de saída: ***Relação de faturas de matéria-prima a pagar***
Requisito: Deve preparar notificação com 48 horas de antecedência ao pagamento.

Restrição - Requisitos das Entradas:
Contribuem para a geração das saídas que o Sistema de Informação deve preparar.
Exemplo de entrada: ***Fatura de matéria-prima da matriz***
Requisito: - A data do vencimento deve ter prazo mínimo de 30 dias para os produtos A1 e B1; e 45 dias para os demais produtos.

Ações Concretas: As ações dos clientes que serão apoiadas pelo sistema desejado [3], fornecem a âncora necessária para se garantir a utilidade do sistema. Quando ficar claramente definido quais as ações e quais os requisitos dessas ações, não restarão dúvidas sobre a serventia do sistema. Dentre todos os requisitos de um sistema os únicos que não podem deixar de serem atendidos, ou mesmo podem ser atendidos parcialmente, são os requisitos de utilidade. Carvalho [3] sugere a sintaxe para a lista de ações concretas de modo a oferecer uma imagem clara, precisa e fiel do que os clientes podem esperar do novo sistema.

(AÇÃO) (Predicado) [(Atributos de abrangência) (Atributos de qualidade)]

Carvalho prescreve que a Ação não pode ser descrita por verbo pouco preciso (do tipo controlar, apurar, administrar, acompanhar, avaliar, ...) e sim verbos ativos concretos bem definidos (do tipo: comprar, vender, cobrar, multar, produzir, treinar, alocar, ...), que especificam com clareza a ação final que deverá ser apoiada pelo Sistema .

Exemplo de ação concreta:

PAGAR, sem atraso, as faturas de fornecedores de matéria-prima da matriz.

O atributo *"sem atraso"* é um requisito de detalhe da ação, e os atributos: *"de matéria-prima da matriz"*, são requisitos de abrangência do sistema.

Eventos Externos: Os eventos que interagem com o sistema contribuem com a percepção mecânica e concreta dentro da enorme abstração que o processo de elicitar requisitos requer [13]. Os Eventos Externos definem claramente as interações do Sistema de Informação com seu UdI para o atendimento das Ações Concretas. Além disso, os requisitos dos eventos fornecem as evidências para a posterior especificação da funcionalidade do Sistema.

Maffeo [14] propõe a Lista de Eventos Externos - LEE como uma lista ordenada de elementos textuais numerados seqüencialmente, onde cada elemento modela um evento externo. Na LEE proposta por Maffeo há dois tipos de eventos externos, e cada tipo deve corresponder a um formato específico para o elemento textual que o identifica:

Exemplo de evento externo: ***Fornecedor faz a cobrança de pedidos de matéria-prima.***

Saídas do sistema: As saídas preparadas pelo Sistema de Informação são o principal produto da criatividade do engenheiro de software. Porque as saídas fornecem as informações relevantes para o apoio às Ações Concretas.

Exemplo de saída: ***Relação de faturas de matéria-prima à pagar.***

Entradas do sistema: As entradas ou estímulos merecem uma atenção especial, porque elas contribuem para a geração das saídas do Sistema de Informação.

Exemplo de entrada: ***Fatura de matéria-prima da matriz.***

2.4.2 Restrições Externas

As Restrições Externas definem as imposições do UdI ao sistema. Elas ficam representadas, no SERBAC, como detalhes dos requisitos de utilidade ou sob a forma de um repasse de informação, que é tratado como uma exceção na formulação das Ações Concretas.

Restrição - Repasse de Informações: Existe uma exceção na formulação das ações concretas, que é o caso de se cumprir uma restrição externa ao sistema, de se fazer um repasse ou REPASSAR alguma informação exigida sem que o engenheiro de software necessite estudar o uso ou a aplicação dessa informação. Para este caso Carvalho [3] recomenda o seguinte enunciado:

A seguir detalharemos o SERBAC enfatizando tanto o processo a ser seguido como o esquema de representação utilizado.

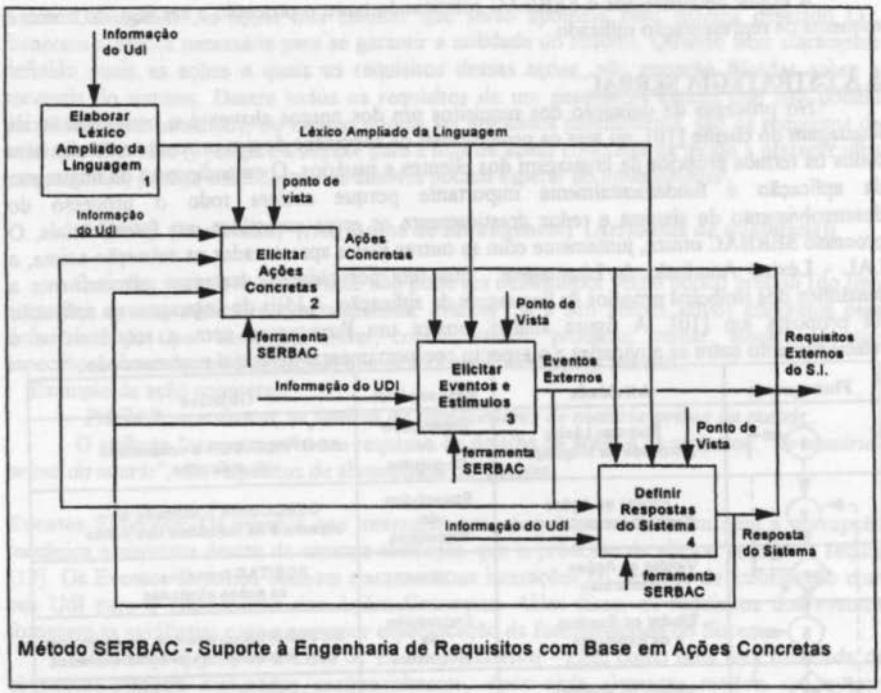
3. A ESTRATÉGIA SERBAC

No processo de definição dos requisitos um dos pontos chaves é a compreensão da linguagem do cliente [10], ou seja os profissionais de requisitos devem procurar entender bem todos os termos próprios da linguagem dos clientes e usuários. O entendimento da linguagem da aplicação é fundamentalmente importante porque acelera todo o processo do desenvolvimento do sistema e reduz drasticamente os erros cometidos nas fases iniciais. O processo SERBAC utiliza, juntamente com as outras idéias apresentadas na subseção acima, o LAL - Léxico Ampliado da Linguagem , que tem por objetivo descrever precisamente a semântica dos símbolos próprios da linguagem da aplicação. A idéia de linguagem da aplicação foi proposta em [10]. A figura abaixo mostra um fluxograma com a seqüência e o relacionamento entre as atividades e o aspecto comportamental-gerencial recomendado.

Fluxograma	Atividade	Responsável	Objetivo
1	Elaborar Léxico Ampliado da Linguagem	Engenheiro de Requisitos	COMPREENDER o vocabulário da aplicação
2	Elicitar as Ações Concretas dos Clientes	Engenheiro de Requisitos	DESCOBRIR a utilidade do sistema e os requisitos das ações
OK N S	Validar as Ações Concretas	Cliente	ACEITAR formalmente as ações elicitadas
3	Elicitar os Eventos e os Estímulos	Engenheiro de Requisitos	DESCOBRIR os eventos, os estímulos e os requisitos dos ebentos elicitados
N OK S	Validar os Eventos os e Estímulos	Cliente	ACEITAR formalmente os eventos elicitados
4	Definir as Respostas Geradas pelo Sistema	Engenheiro de Requisitos	PROPOR as respostas aos eventos e os requisitos das respostas
OK N S	Validar as Respostas Geradas pelo Sistema	Cliente	ACEITAR formalmente os requisitos externos do sistema

A figura da próxima página fornece uma outra visão através de um diagrama SADT [18]. Este diagrama serve para evidenciar as sub-atividades. A validação dos produtos está dividida ao longo do processo para que os atores se defrontem com os problemas dos requisitos o mais cedo possível, além de diluir a carga deste trabalho dos clientes ao longo do processo de elicitação. Na próxima figura podemos observar também que: a) O controle de nome "ponto de vista" (PV) representa que o processo deve ser feito considerando apenas um ponto de vista de um grupo homogêneo de clientes. Se o engenheiro de software desejar, e se for objeto do trabalho, poderá aplicar o processo para outro ponto de vista. Após a Elicitação dos Requisitos Externos de um sistema para mais de um ponto de vista, ao final, o engenheiro de software deverá fazer mais uma sessão JAD [1] para conciliar as possíveis divergências. b) O controle "Léxico Ampliado da Linguagem" é um guia para o processo. c) As Ações Concretas depois de elicitadas e aprovadas pelos clientes na atividade número 2, funcionam como referencial para a elicitação dos demais requisitos externos. d) Os Eventos Externos elicitados, bem como as Respostas do Sistema fornecem uma retro-alimentação para as atividades iniciais do processo em caso de erros ou divergências.

Em seguida resumiremos cada atividade do processo SERBAC.



Método SERBAC - Suporte à Engenharia de Requisitos com Base em Ações Concretas

3.1 ELABORAR LÉXICO AMPLIADO DA LINGUAGEM

Tem a responsabilidade de identificar o "vocabulário da aplicação", segundo [10], no qual procura-se respeitar a forma sintática das palavras usadas no jargão do UdI. Através do entendimento da Linguagem da Aplicação é possível uma melhor comunicação com os clientes, o que irá facilitar o trabalho posterior de validação dos fatos elicitados.

LAL do Sistema AVP

O LAL demonstrou nesta experiência ser um produto de elaboração simples e de imediato entendimento pelo usuário do sistema AVP. Conseguimos um rápido entrosamento com a técnica.

A duração da confecção do LAL para o Sistema AVP foi equivalente a 10 dias úteis, incluindo neste período a conferência da documentação. Na elaboração do LAL o mais difícil foi conseguir a validação informal dos termos elicitados. Pedimos aos clientes uma conferência dos termos elicitados e só conseguimos melhor retorno quando o esclarecimento das dúvidas foram trocadas entre os 2 clientes disponíveis.

O LAL tem a tarefa de identificar o "vocabulário da aplicação", segundo [10]. No LAL procura-se respeitar a forma sintática das palavras usadas no jargão do UdI.

Através do entendimento da Linguagem da Aplicação podemos conseguir uma comunicação melhor com os clientes, o que irá facilitar o trabalho posterior de validação dos fatos elicitados. Porém comprovamos, neste exemplo, que o LAL vai mais além: ele forneceu a matéria-prima para a elicitação de todos os Requisitos Externos do Sistema AVP.

A atividade Elaborar Léxico Ampliado da Linguagem é apoiada por uma ferramenta de hipertexto - O HiperLex -, que implementa a estrutura proposta em [6].

No nosso caso, como o objetivo é a descoberta dos requisitos externos, na elaboração do LAL do Sistema AVP demos prioridade para a pesquisa e seleção dos componentes do UdI que estavam fora do sistema ou que poderiam fornecer ajuda na elicitação destes requisitos. Além disso, na descrição dos impactos foram indicadas somente as implicações com o UdI; ou seja, não foram registradas as implicações internas do funcionamento do sistema.

Na proposição do LAL é recomendado que seja elicitado cada "símbolo" que represente uma palavra ou uma frase da linguagem da aplicação, a "noção" que descreve o símbolo e os "impactos" que são as possíveis implicações ou interrelações causadas pelo símbolo no domínio da aplicação.

Abaixo listamos algumas das entradas do LAL do sistema AVP.

AVP disponível
>- **Pedido de viagem** à serviço.
>- **AVP** correta no preenchimento, esperando **autorização**.
- **A AVP disponível** após a **autorização**, se torna **AVP liberada** ou **AVP autorizada** para as **providências de uma viagem**.

AVP / Autorização de Viagem no País
>- Nome do sistema em estudo o qual lida com Autorizações de Viagens a serviço no País.
>- Nome do **pedido de viagem** à serviço ou nome de uma **programação de viagem** à ser autorizada.
- Fica em situação de **AVP disponível** após o preenchimento.
- Requer a **autorização** por **chefe credenciado**.

Cancelamento de AVP / Cancelamento da viagem / Cancelar AVP
>- Operação de cancelamento da **AVP** para alterar alguma informação da **AVP liberada** ou por **cancelamento da viagem**.
- Provoca o **cancelamento de reserva de passagens**.
- Provoca o **cancelamento de reserva de hotéis**.

Cancelamento de reserva de hotel
>- Desistência da **reserva de hotel**.
- Provoca um desistência da **reserva de hotel**, devido a alteração da **viagem a serviço** ou devido ao **cancelamento da viagem**.
- Provoca alteração no **pagamento de diárias**.

Solicitação de viagem / Pedido de viagem / Solicitar viagem
>- Pedido ou **programação de viagem** a receber **autorização** por **chefe credenciado**.
- Implica no preenchimento de uma **AVP** para **autorização**.
- Fica em situação de **AVP disponível** após o preenchimento.
- Requer a **autorização** por **chefe credenciado**.

3.2 ELICITAR AS AÇÕES CONCRETAS DOS CLIENTES

Tem por finalidade descobrir a utilidade do sistema em função das Ações Concretas dos Clientes, as quais serão apoiadas pelo sistema. Definirá também os requisitos e as restrições externas que qualificam e delimitam a atuação das Ações Concretas identificadas.

Devem ser elicitadas: **as ações concretas dos clientes apoiadas pelo sistema; os requisitos das ações concretas; e os diagnósticos das ações.**

Exemplo: tela de inclusão ou alteração de uma ação concreta.

SERBAC **Atualizar Ações Concretas**

Cliente :[Órgão Financeiro]
Ação :[PAGAR]
Objeto :[diárias de AVPs]
Requisito : 1)[com segurança - via Sistema de Contas a Pagar;]
2)[em tempo hábil - na conta corrente do empregado.]
3)[ ]

Exemplo: tela da apresentação de Ações Concretas.

SERBAC **Ações Concretas**

1 - SOLICITAR viagem a serviço.
2 - CANCELAR AVP de viagem a serviço.
2.1 - CANCELAR reserva de passagem.
2.2 - CANCELAR reserva de hotéis.
2.3 - CANCELAR pagamento de diárias.
3 - RESERVAR hotéis de uma AVP.
4 - RESERVAR passagens de uma AVP.
5 - PAGAR diárias de uma AVP; com segurança; em tempo hábil.
6 - COBRAR prestação de contas de empregado; com pendência em uma AVP.
7 - FORNECER informações de passagens compradas ao Órgão Financeiro.
8 - FORNECER informações sobre viagens ao Órgão solicitante.

As sub-atividades de Elicitar as Ações Concretas dos Clientes

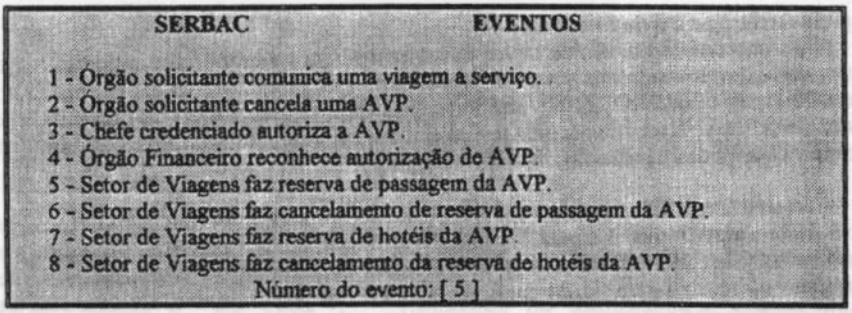
1. Produzir relação de Ações Concretas.
   a. Preparar relação de "Ações Candidatas", a partir do LAL.
   b. Ajustar a relação de "Ações Candidatas", com o representante dos clientes.
2. Complementar relação de Ações Concretas com requisitos de detalhe.
   a. Elaborar sugestão de requisitos das ações, a partir do LAL.
   b. Aprimorar os requisitos das ações com o representante do cliente.
   c. Realizar reunião, ou sessão JAD, para refinamento de diagnósticos.
   Diagnósticos servem para detalhar possíveis problemas na realização da ação [15].
3. Solicitar à apreciação do cliente de mais alto nível a Relação das Ações Concretas.
   - Tem por finalidade receber formalmente dos clientes a aprovação das Ações Concretas elicitadas na atividade. O cliente também aceitará os requisitos e as restrições externas que qualificam e delimitam a atuação das Ações Concretas identificadas.

Em caso de problema na aprovação, o engenheiro de software deverá retornar à atividade 1- Elaborar Léxico Ampliado da Linguagem para reiniciar o processo.

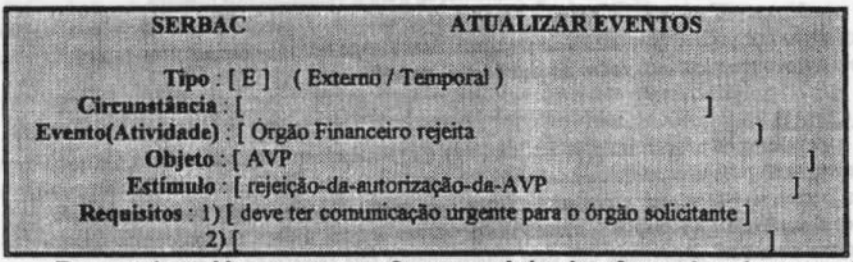
3.3 ELICITAR OS EVENTOS E OS ESTÍMULOS ASSOCIADOS ÀS AÇÕES

Tem por finalidade explicitar os eventos externos a que o sistema estará preparado à responder e definir os requisitos (e restrições) envolvidos. Devem ser elicitados: ***os eventos externos*** do sistema, ***os requisitos dos eventos*** (- restrições da ocorrência e atributos de qualidade do evento; - a restrição da seqüência de realização do eventos; - definição do estimulo (entrada) produzido pelo evento; - identificação da ação que o evento está associado; - identificação das saídas do sistema para as quais a entrada contribui), e ***os diagnósticos*** desses eventos.

No exemplo de tela abaixo, listamos alguns dos eventos do Sistema AVP.



Os requisitos de detalhe (- a restrição da seqüência de realização do evento; - identificação da ação que o evento está associado) são exemplificados em [15].



Em caso de problema na aprovação, o engenheiro de software deverá retornar à atividade 2- Elicitar as Ações Concretas dos Clientes.

3.4 DEFINIR AS RESPOSTAS GERADAS PELO SISTEMA

Tem por finalidade a definição das respostas planejadas do sistema aos eventos externos e aos eventos temporais. Definirá também os requisitos das saídas para o atendimento das Ações Concretas. Devem ser formuladas: **as saídas do sistema, os requisitos das saídas** (- restrições da ocorrência e atributos de qualidade da saída; - identificação da ação que a saída está apoiando) **e os diagnósticos das saídas.**

As sub-atividades de 4- Definir as Respostas Geradas pelo Sistema são semelhantes as sub-atividades da atividade 2- Elicitar Ações Concretas dos Clientes.

Em caso de problema na aprovação, o engenheiro de software deverá retornar à atividade 3- Elicitar os Eventos e os Estímulos Associados as Ações.

4. A FERRAMENTA SERBAC

A exemplificação do SERBAC através do uso da ferramenta construída foi feita na seção anterior. O exemplo foi contruido do estudo de caso "Elicitação dos Requisitos Externos do Sistema AVP". Os requisitos da ferramenta foram derivados dos estudos expostos nas Seções 2 e 3. Na ferramenta, foram incluídas as facilidades necessárias para que se possa apoiar um processo eficaz de validação. As facilidades de edição e alteração presentes na ferramenta fazem com que esse apoio se efetive.

O Sistema AVP é um Sistema de Informação de porte médio. Seus principais clientes são todos os órgãos da PETROBRÁS que possuem empregados em viagem à serviço no país.

Apresentação Sucinta da Ferramenta.

Para facilitar o trabalho de elicitar os requisitos, sem perdas de informações ao longo do processo, construímos uma ferramenta de software (protótipo) para auxilio ao engenheiro de software: "O Sistema SERBAC". Este protótipo também ajuda os desenvolvedores durante todo o ciclo de vida do sistema, fornecendo a memória para a incorporação dos requisitos externos e para a análise das soluções dos problemas elicitados.

A ferramenta SERBAC é composta por 4 módulos para a atualização dos requisitos externos do sistema e por 4 módulos para a impressão de relatórios. Está implementada em Pascal e tem um total de 6.500 linhas de código. Nos módulos de atualização, o engenheiro de software registra os requisitos do componente, faz a ligação de interdependência com os outros componentes e faz também o registro dos diagnósticos de problemas encontrados no componente. Os módulos de impressão de relatórios estão preparados para fornecer saídas que mostram os relacionamentos entre os componentes registrados. E, de acordo com opções, podemos escolher subconjuntos da documentação para a validação dos clientes.

Na primeira versão da ferramenta e constatou-se que a parte de interface não incentivava o uso por parte dos analistas de sistemas. Atualmente estamos desenvolvendo uma nova versão aproveitando as facilidades do padrão "Windows".

5. CONCLUSÃO

Para se conseguir um processo de Definição de Requisitos bem sucedido é conveniente lembrarmos que são pessoas que definem os requisitos e os validam. Esses atores do processo têm portanto que se comunicar e portanto devem usar uma notação e uma estrutura para documentar o conhecimento elicitado numa forma comum e fácil de ser compreendida. No entanto é também importante que esse conhecimento seja registrado de maneira rigorosa para que oriente a posterior modelagem do sistema.. É importante também que existam procedimentos de gerência sobre o processo de modo a permitir a distribuição de recursos e controle do andamento das atividades.

Nosso trabalho tem por objetivo justamente propor um processo de definição de requisitos baseado em idéias já testadas e que são singulares por sua simplicidade. No entanto cabe observar os limites de nossa proposta, isto é, estamos fundamentalmente interessados nos **Requisitos Externos do Sistema**, e mesmo assim estamos enfatizando o registro de requisitos que se enquadram no nosso esquema semântico. Nossa estratégia objetiva a definição de requisitos para sistemas ligados à organização (administração) e serve tanto para definir sistemas de informação como para definir o software que será parte desse sistema de informação. Apesar de acreditarmos que nossa proposta possa ser útil também para outros tipos de sistemas (software de tempo real) não avaliamos o seu uso para essas aplicações.

Trabalhos futuros estarão direcionados a uma maior experimentação com o SERBAC e com a investigação mais detalhada dos aspectos cooperativos que o permeiam.. Uma das possibilidades de exploração é justamente os aspectos ligados a argumentação em favor e contra: ações concretas, eventos externos, entradas, saídas e diagnósticos. O uso de um esquema como o apresentado por Pereira [16] como apoio à um SERBAC cooperativo é uma atividade de exploração que pretendemos trilhar.

BIBLIOGRAFIA

[1] August, Judy H.; **JAD : Joint Aplication Design**, Yourdon Press; Prentice Hall Building, 1991.

[2] Burstin, Meir; **Requirements Analysis of Software Systems** PhD. Thesis - Tel-Aviv University; 1984.

[3] Carvalho, Luiz C. de Sá; **Análise de Sistemas: o outro lado da informática**, LTC - Rio de Janeiro, 1988.

[4] Chen, P.; **"The Entity-Relationship Model - Toward a unified view of data"** ACM Transations on Database Systems vol.1 n.1 p.9, 1976.

[5] Davis, Gordon B.; **"Strategies for Information Requirements Determination"**, in IBM Systems Journal, Vol.21, N.1, 1982.

[6] Franco, Ana Paula; **Métodos e Representações de Suporte à Aquisição de Linguagens da Aplicação**, Dissert. Mestrado, Depto. de Informática PUC RJ 1992.

[7] Gause D. C., e Weinberg, G. M.; **Explorando os Requerimentos de Sistemas**, Makron, McGraw-Hill, 1991.

[8] Goguem, J., and Linde, C.; **"Techniques for Requirements Elicitation"**, International Symposium on Requirements Engineering, IEEE Computer Society, San Diego CA; pp.152-164; 1993.

[9] Jackson, M. A.; **System Development**, Prentice-Hall, 1983.

[10] Leite, Julio. Cesar. Sampaio do P.; e Franco, Ana Paula; **"O Uso de Hipertexto na Elicitação de Linguagens de Aplicação"**, IV Simpósio Bras. Engenharia de Software - SBC , 1990.

[11] Leite, Julio C. S. P.; **"Sistemas de Informação e Engenharia de Software, o Elo Gerencial"** Anais XXIV Congresso Nacional de Informática, SUCESU-SP, 1991.

[12] Leite, Julio Cesar S. P.; **"Enhancing the Semantics of Requirements Statements"**, Proceedings of the XII International Conference of the Sociedad Chilena de Ciencia de la Computacion, Pags. 281-297, 1992.

[13] McMenamin S. M., e Palmer, J. F.; **Análise Essencial de Sistemas** McGraw-Hill São Paulo, 1991.

[14] Maffeo, Bruno; **Engenharia de Software e Especificação de Sistemas**, Editora Campus Ltda., Rio de Janeiro, 1992.

[15] Oliveira, Antônio de Pádua Albuquerque; **SERBAC - Suporte a Engenharia de Requisitos com Base em Ações Concretas**, Dissertação de Mestrado, Depto. de Informática - PUC - RJ 1994.

[16] Pereira, Florys F. A.; **CDR - Cooperação na Definição de Requisitos**, Dissertação de Mestrado, Depto. de Informática - PUC - RJ 1994.

[17] Pressman, Roger S.; **Software Engineering - A Practioner's Approach**; McGraw-Hill Book Co., 1987.

[18] Ross, D.; **"Structured Analysis (SA): A Language for Communicating Ideas"**, IEEE Trans. Software Eng. V. SE-3 1977.