INFORMÁTICA 83

XVI CONGRESSO NACIONAL DE INFORMÁTICA

SÃO PAULO OUTUBRO 83

SEGURANÇA EM SISTEMAS AUTOMATIZADOS

Arndt von Staa

PUC/RJ, Departamento de Informática
Rua Marques de São Vicente, 225
22453 - Rio de Janeiro, RJ
Tel.: 274-9922 - R. 386

Palavras-chave: agressões, classificação de agressões, contramedidas, erros, falhas, fraudes, proteção, qualidade de software, sabotagens, segurança, sistemas de software.

Resumo

É definido um modelo de classificação de agressões. É ilustrado o uso deste modelo. São apresentadas classes de contramedidas. Cada classe é examinada sucintamente quanto à sua eficácia. É apresentado, em linhas gerais, um método para o desenvolvimento de sistemas satisfatoriamente seguros por construção.

1. Introdução

O objetivo principal deste artigo é apresentar um modelo para a classificação de agressões e, também, apresentar um método de especificação baseado neste modelo que nos permita assegurar o alcance de níveis satisfatórios de segurança. Deseja-se com isto apresentar um método racional e economicamente justificável capaz de conduzir a sistemas seguros por construção. Um tal método é desejável, uma vez que, em geral, a introdução caso a caso de mecanismos de proteção em sistemas já desenvolvidos tem-se mostrado ineficaz e caro. Apesar de primariamente dirigido à construção de sistemas, o método aplica-se também a adaptação de sistemas, sendo que possui a virtude de resultar em projetos de alteração levando a níveis de segurança desejáveis e não simplesmente protegendo de modo casuístico.

Sistemas automatizados estão se tornando cada vez mais ubíquos. Cada vez mais as pessoas, as empresas e a sociedade dependem dos serviços prestados por sistemas de computação. Porém, à medida que pessoas, empresas e sociedade vão se tornando mais dependentes de computadores, mais sensíveis elas se tornam a agressões a estes sistemas de computação.

Gradativamente a sociedade vem percebendo que, embora intrinsicamente quase imateriais, programas, dados e resultados possuem elevado valor. A não disponibilidade na oportunidade certa, a baixa qualidade, a perda e/ou ou mau uso de programas, dados e resultados podem constituir uma séria ameaça à sociedade e à vida das empresas. Similarmente o acesso, cópia e uso de programas, dados e resultados por pessoas não autorizadas podem levar a consideráveis prejuízos.

A revolução do processamento de dados trouxe consigo enormes benefícios, além da possibilidade de resolução de problemas nunca antes imaginados. Porém, trouxe consigo também uma formidável fonte de riscos. No entanto, a velocidade com que evoluem os sistemas de computação tem reduzido a capacidade de avaliação destes riscos e dos danos possivelmente causados. Pode-se dizer então que, de modo geral, segurança de sistemas automatizados não tem recebido a atenção que merece.

Todos nós já fomos vítimas de algum defeito de processamento ou de manipulação do equipamento, levianamente posto de lado como fato de pouca monta e contra o qual nada há que possa ser feito. Quem ainda não escutou a famosa frase: "Foi erro de computador"? Ora sabemos muito bem que o erro foi de um sistema incorretamente projetado, incorretamente operado, incorretamente usado, incapaz de proteger-se contra as mais simples agressões, e/ou outro defeito qualquer.

Estes mesmos sistemas são os que controlam usinas nucleares, destilarias, tráfego aéreo, contas correntes etc. Temos pois que aprender como estabelecer níveis de segurança que nos permitam conviver com sistemas automatizados, bem como que impeçam o mau uso ou uso desautorizado destes sistemas.

Para podermos entender melhor o problema, temos que identificar o objetivo da segurança de sistemas que é: assegurar que os riscos inerentes à existência e à utilização do sistema sejam aceitáveis e que o custo do estabelecimento da segurança seja compatível com o valor (benefícios) do serviço deste sistema e compatível ainda com o valor dos recursos, programas e dados que manipula.

Para tal precisamos alcançar os seguintes objetivos básicos da segurança de sistemas automatizados:

a - reduzir a probabilidade de sucesso das agressões ao sistema automatizado;

b - manter sob controle o dano causado por agressões que tenham alcançado êxito;

c - assegurar a possibilidade de recuperação do sistema danificado de modo que possa voltar a operar realizando um serviço de qualidade satisfatória.

Nas seções a seguir apresentamos um modelo de identificação de agressões e, após, esboçaremos como proceder para incorporar contramedidas (proteção) visando o alcance de níveis de segurança especificados.

2. Agressões

Nesta seção descreveremos um método de identificação das diversas agressões a que são sujeitos os sistemas automatizados.

Cabe observar que agressões podem ocorrer por acaso (erro fortuito), portanto não sendo necessariamente fraudes ou sabotagens. Cabe salientar também que agressões fortuitas são as mais frequentes e que podem ser tão ou mais danosas que fraudes e sabotagens.

A inventividade do homem e a sua proverbial habilidade de errar transcende à capacidade de imaginação de indivíduos e mesmo de grupos de indivíduos. Sendo, assim, o emprego de exemplos, reais ou imaginados, não serve como um mecanismo para assegurar a especificação e posterior implementação de níveis de segurança satisfatórios. Tal é verdade não só para agressões efetuadas conscientemente, como é verdade também para agressões fortuitas, ou seja decorrentes de ações realizadas sem intenção de agredir.

Como já mencionamos, exemplos de agressões são meios inadequados para projetar níveis de segurança, pois a qualquer momento poderá surgir alguém capaz de inventar (voluntária ou involuntariamente) uma nova agressão não antes imaginada, capaz de burlar os mecanismos de proteção existentes e capaz de causar sérios danos. Precisamos pois de um método para sistematizar agressões a sistemas automatizados de modo que possamos especificar e posteriormente verificar o nível de proteção e segurança alcançado pelo sistema.

Para podermos identificar agressões, precisamos antes saber o que vêm a ser. No contexto deste artigo, entendemos agressões como sendo ações, premedidas ou não, que:

i - sejam possíveis em virtude do uso e/ou da mera existência do sistema;

ii - tenham por alvo algum componente ou serviço do sistema, ou, então, o ambiente operacional deste sistema;

iii - sejam capazes de causar, direta ou indiretamente, danos materiais, financeiros, sociais e/ou morais;

sendo que tais ações podem ser realizadas por pessoas vinculadas ou não ao sistema, pelo ambiente operacional e até pelo próprio sistema.

Dizemos que uma agressão obtém êxito (ou sucesso) caso consiga perpetrar o dano correspondente. Note que isto se aplica também ao caso de agressões acidentais, apesar destas não causarem danos premeditados.

Para ilustrar a abrangência desta definição vejamos alguns exemplos de agressões:

1 - falta de confiabilidade nos programas; leva freqüentemente a resultados errados, provocando assim a perda do esforço de computação, bem como tornando necessários acertos, recuperação e reexecução.

2 - dados errados: podem corromper os bancos de dados levando os usários do serviço prestado pelo sistema a realizarem ações e/ou decisões incorretas, muitas vezes requerendo considerável esforço de recuperação.

3 - uso fraudulento do sistema: utilizar o sistema com dados aparentemente válidos com o intuito de auferir benefícios do serviço deste sistema. Por exemplo, foi noticiado há alguns anos o uso do sistema de restituição de imposto de renda retido na fonte, onde os fraudadores preenchiam declarações fictícias com o intuito de receberem o retorno de imposto "retido" na fonte e que, evidentemente jamais havia sido retido.

4 - falta de energia ou quebra de equipamento em sistemas "on-line": pode levar à perda de informação e certamente leva a alguma perda de esforço realizado.

5 - corrupção de operadores: pode levar à suspensão de controles e, desta forma, pode tornar possível que agressões antes controladas venham a ter êxito.

As agressões podem ser classificadas quanto

1 - ao alvo:

a - o serviço, ex.: utilizar o sistema de foram aparentemente legal para obter benefícios ilegais

b - o ambiente onde opera o sistema, ex.: falta de energia

c - o hardware, ex.: equipamento defeituoso
d - o pessoal, ex.: suborno
e - os procedimentos, ex.: instruções de aquisição e preparação de dados incorretos
f - os programas, ex.: programa mal especificado
g - os dados, ex.: dados não válidos
h - os documentos, ex.: falta de atualização na documentação

2 - à motivação:
a - fortuito, ex.: erro de especificação
b - negligência, ex.: não obedece às normas de programação
c - intencional, ex.: acesso a dados não autorizados

3 - à forma:
a - erro de uso, ex.: erros nos resultados em função de dados de entrada incorretos
b - adulteração, ex.: corrupção do banco de dados através de atualizações com dados errados
c - destruição, ex.: delir acidentalmente alguns arquivos

4 - ao modo:
a - ativo, ex.: corrupção do banco de dados
b - passivo, ex.: coleta de lixo

5 - à origem
a - interna, ex.: manutenção mal feita, programa com falhas
b - ambiente operacional, ex.: erro de operação, software suporte em erro
c - externa, ex.: preparação de dados mal feita; interpretação errada dos resultados.

Na figura 1 mostramos o modelo de classificação de agressões. Cada caminho da origem ao alvo representa uma classe de possíveis agressões. Cabe observar que uma mesma agressão pode percorrer vários destes caminhos. A seguir ilustramos o uso do modelo através de alguns exemplos.

ambiente - ativo - uso - negligente - hardware:
corresponde a danos causados no hardware específico do sistema provocados por negligência do operador.

externo - ativo - uso - intencional - serviço:
corresponde às fraudes decorrentes do uso do sistema com dados aparentemente válidos, visando obter benefícios não autorizados.
interno - ativo - destruição - intencional - dados:
corresponde à sabotagem do banco de dados efetuada por pessoas diretamente vinculadas ao sistema.
externo - passivo - uso - intencional - dados:
corresponde à aquisição premeditada de dados não autorizadas via coleta de lixo ou através da escuta nas linhas de teleprocessamento.
externo - passivo - uso - acidental - dados:
corresponde à aquisição acidental de dados não autorizados. Isto pode ocorrer, por exemplo, ao solicitar, em outro sistema, um "dump" de memória contendo dados remanescentes do sistema agredido. Neste caso uma pessoa que, em princípio não desejava fazer mau uso do sistema agredido, poderá vir a fazê-lo em função dos dados acidentalmente adquiridos.
externo - ativo - adulteração - intencional - pessoal e
externo - ativo - adulteração - intencional - programa:
suborno de programadores com o intuito de remover medidas de proteção contidas em programas.

Para cada agressão existe também pelo menos um caminho da origem ao alvo. Ilustramos isto através de alguns exemplos:
falta de energia devida à falta de manutenção de estabilizadores de tensão:
ambiente operacional - ativo - adulteração - negligente - ambiente
erro de processamento devido à baixa qualidade de manutenção:
interno - ativo - adulteração - negligência - programas
depósito fraudulento em conta corrente:
externo - ativo - adulteração - intencional - dados
erro por acaso na preparação de dados e que afetou somente os resultados:
externo - ativo - uso - fortuito - serviço
erro por acaso na preparação de dados levando à corrupção do banco de dados:
externo - ativo - adulteração - fortuito - dados, e
externo - ativo - adulteração - fortuito - serviço

3. Contramedidas

Nesta seção esboçamos como utilizar a classificação de agressões para selecionar contramedidas. Uma contramedida é um mecanismo colocado em determinado ponto do sistema visando:

a - impedir o êxito de agressões detectando-as antes de causarem os danos que tinham por objetivo; ou
b - dissuadir as tentativas de agressões, reduzindo assim a probabilidade das ocorrências de tentativas de agressão; ou
c - detectar agressões que lograram êxito, mantendo sob controle os possíveis danos.

A palavra mecanismo é utilizada aqui num sentido extenso significando coisa tais como: código de controle, treinamento de pessoal, guardas, controle de qualidade, rotinas de prevenção de acidentes, avisos, cofres, etc.
O custo do estabelecimento de determinado nível de segurança é função do nível de segurança desejado. É pois necessário escolher contramedidas que permitam o alcance de níveis satisfatórios e economicamente justificáveis. Uma contramedida deve ser então entendida como um filtro. Ela é capaz de controlar e impedir a passagem de determinadas categorias de agressões, sendo insensível a outras. Por exemplo, uma rotina de validação de dados (validade estrutural) impede que dados fora de determinados padrões prossigam no processamento, porém uma tal rotina é incapaz de identificar dados válidos mas cujos significados (valores) estejam em erro (validade semântica). Na figura 2 apresentamos uma visão integrada dos componentes e das interfaces de um sistema automatizado, evidenciado onde colocar os diversos filtros.
Tendo em vista a falibilidade das contramedidas, logo percebemos que devemos estabelecer um conjunto de contramedidas de modo a reduzir e probabilidade de êxito das diversas agressões. Para isto devemos reduzir a probabilidade de tentativas da agressão e aumentar a probabilidade de deteção da agressão.
Observando o modelo de classificação de agressões, é evidente que contramedidas visando a forma e a motivação das agressões são muito mais eficazes que as visando somente alvos específicos. Um bom projeto de contramedidas conterá pois um conjunto de mecanismos visando estabelecer barreiras nas cinco classes: origem, modo, forma, motivação e alvo.

3.1 - Detecção do sucesso de agressões

A detecção de agressões que obtiveram êxito é o mecanismo menos eficaz uma vez que somente gera alertas **após** o sucesso da agressão. Consequentemente o dano já ocorreu e passamos a ter que controlar a sua propagação, bem como passamos a ter que realizar tarefas de recuperação visando repor em operação o sistema agredido de modo que possa voltar a produzir um serviço de qualidade satisfatória.
Para podermos detectar o êxito de agressões, necessitamos a incorporação de instrumentos de monitoramento no sistema e no ambiente operacional. É evidente que tais instrumentos precisam ser projetados especificamente já durante a construção do sistema, uma vez que podem requerer dados de acompanhamento específicos (ex.: "logs", "auit trail"). Para poder gerar tais dados de acompanhamento podem ser necessárias técnicas especiais de projeto e construção do sistema.
Após à detecção do êxito de uma agressão torna-se, em geral, necessário realizar uma recuperação a fim de eliminar as conseqüências desta agressão. Portanto mecanismos de detecção devem sempre estar acompanhados de instrumentos de recuperação capazes de repor o sistema em funcionamento a um nível satisfatório de qualidade. Isto por sua vez requer a incorporação, já durante o projeto do sistema, de uma série de instrumentos, capazes de gerarem os dados necessários para realizarmos a recuperação (ex.: "restart point"). Note que os dados de acompanhamento não são necessariamente os mesmo que os de recuperação. Necessitam porém estar bem sintonizados de modo que seja possível a detecção e a posterior recuperação.
Os mecanismos de detecção do êxito de uma agressão veêm o sistema como se fosse uma caixa preta. Requerem pois instrumentos que examinem os dados de acompanhamento das operações do sistema e/ou de instrumentos que examinem as estruturas, os arquivos, ou banco de dados utilizados pelo sistema. Exemplos de tais instrumentos são: programas capazes de percorrer os arquivos gerando mensagens de erro ou advertência sempre que encontrarem alguma condição não admissível ou pouco plausível. Estes erros são tipicamente erros de organização (estrutura) dos dados, validade dos dados e validade semântica (significado) dos dados. Para podermos criar tais instrumentos é óbvio que necessitamos de especificações precisas das organizações dos dados e dos programas.
Os instrumentos de detecção após a agressão são semelhantes aos empregados quando efetuamos auditoria de sistemas em torno do computador, possivelmente utilizando ferramentas específicas (auditoria com o computador). Aos leitores interessados em mais detalhes sugerimos as seguintes referências [Morais 80, Parentes 78].

3.2 - Detecção de tentativas de agressão

A detecção de tentativas de agressão têm por objetivo evitar que o dano objetivado pela agressão venha a ocorrer. É claro que representa custo o esforço despendido para detectar, bem como o esforço despendido para identificar as causas da agressão. Em um sistema de contramedidas bem projetado, este custo no entanto é bem menor do que o dano que a agressão poderia vir a

causar caso obtivesse êxito.
Uma técnica essencial para a detecção de tentativas de agressão é a validação. A validação pode ser dividida em 3 classes:
1 - **validação léxica:** onde verificamos se o dado satisfaz uma ou mais condições, ex.: numérico e 1.
2 - **validação sintática:** onde verificamos se um conjunto de dados satisfaz um conjunto de relações entre eles, ex.: soma é igual ao total de controle, número da peça existe no arquivo de peças, etc.
3 - **validação semântica:** onde procuramos verificar se o dado corresponde à realidade e/ou ao que desejamos que represente (correção, fidelidade).
Para simplificar a discussão a seguir, o par validação léxica e validação sintática será chamado de **validação estrutural.**
Os instrumentos de detecção de tentativas de agressão são semelhantes aos instrumentos de auditoria de sistemas de computação, quando esta é realizada através do sistema [Morais 80]. São semelhantes também a instrumentos que incorporamos a sistemas tolerantes a falhas [Magalhães 79], e a instrumentos que visam coletar dados para fins da depuração de programas [Staa 83].
Dentre os melhores instrumentos encontram-se as "assertivas mecanizadas".
Uma assertiva de entrada é um conjunto de afirmações determinando condições a serem satisfeitas para que o processo a seguir possa operar corretamente.
Uma assertiva de saída é um conjunto de afirmações determinando as condições que serão satisfetitas caso o processo precedente tenha operado corretamente.
Uma boa assertiva de saída assegura que, se todas as condições forem satisfeitas ao terminar o processo precedente, então este processo terá executado corretamente. É óbvio que isto poderá ser assegurado somente se a assertiva de saída for abrangente, o que em alguns casos é difícil de conseguir.
Na figura 3 ilustramos a colocação de assertivas em uma especificação.
Com relação a assertivas de entrada também pode ser difícil identificar todas as condições a serem satisfeitas. Este problema é agravado ainda mais quando interligamos vários sistemas e/ou quando o sistema contém malhas onde ocorram processos realizados manualmente.
Para tornarmos um sistema capaz de detectar agressões antes destas alcaçarem êxito, devemos efetuar o seguinte procedimento relativo a qualquer uma das funções e/ou componentes do sistema:
1 - determinar todas as condições que devem ser satisfeitas pelos dados de entrada a cada um dos processos do sistemas assumindo que estes processos não realizarão a validação estrutural e semântica. Não basta indicar a validade estrutural (sintaxe, condições, relações, etc.) destes dados, mas deve-se atentar também para a validade semântica (significado, fidelidade, precisão) destes dados. Por exemplo, no caso da fraude de imposto de renda mencionada anteriormente, não basta saber que o CPF e adeclaração estejam válidos, é necessário também identificar declarações contendo CPF's e outros campos válidos porém fraudados, como por exemplo: multiplicidade de declarações, rendimentos não auferidos por este declarante, etc.
2 - reduzir as exigências de validade estrutural e semântica dos dados de entrada a cada um dos processos. Para tal estes processos deverão incorporar mecanismos capazes de detectar todos os desvios da validade ideal. Note que é essencial podermos assegurar que qualquer desvio da validade ideal será detectado. Cabe observar que se os dados de entrada não satisfizerem estas novas condições de validade (com rigor reduzido), o processo poderá operar erroneamente. Consequentemente o caso ideal será o de permitir o processo ser ativado com dados quaisquer, sendo os filtros internos a este processo suficientemente poderosos para detectar todos os desvios da especificação de validade original. Desta forma o processo somente será completado caso os dados de entrada sejam achados estruturalmente e semanticamente válidos, sendo suspenso caso contrário, impedindo assim a consecução do dano. Note que quanto maior o rigor de definição da assertiva de entrada, no passo 1 acima, menor será a probabilidade de agressões passarem despercebidas pelo processo.
3 - introduzir respostas apropriadas a agressões detectadas. Não basta indicar que foi detectada uma agressão, é necessário também identificar quais as suas possíveis causas. Estas podem variar desde um mau projeto da interface homem/sistema até tentativas de fraude. Também é amplamente sabido que simplesmente suspender o processamento do sistema com alguma mensagem de erro, ou suspender a transação em curso nem sempre é a melhor resposta. Consequentemente todos os processos contendo filtros passarão e sinalizar exceções. Estas deverão receber um tratamento adequado em outros processos (tratadores de exceções) de modo que possamos assegurar o nível de segurança desejado, bem como assegurar a identificação das possíveis causas para a agressão detectada.
4 - prever e incorporar mecanismos de controle dos controles, uma vez que os próprios filtros podem estar em erro, ou podem ter sido adulterados por alguma outra agressão.

Como regra geral deve-se realizar alterações em arquivos somente após uma ampla e minuciosa validação léxica, sintática e semântica.
Cabe salientar que o espírito dos passos acima descritos pode e deve ser empregado com relação aos demais componentes (alvos) do sistema. Assim devemos efetuar um rigoroso controle de qualidade dos programas, procedimentos e documentação, contratar pessoal competente e dar-lhe um treinamento rigoroso etc. A aplicação destas regras a outros componentes é, no entanto, uma modalidade de dissuação de agressões.

3.3 - Dissuação de tentativas de agressão

Métodos de dissuação de tentativas de agressão têm por objetivo a redução significativa da freqüência de tentativas (probabilidade) de agresssão, consequentemente reduzindo os custos e os transtornos decorrentes, mesmo quando a agressão for detectada antes de alcançar êxito. É pois o método mais eficaz de estabelecimento de níveis de segurança satisfatórios.
Cabe salientar que a maior parte das tentativas de agressão têm motivos fortuitos, o que não significa que sejam pequenos os danos que possam causar. Uma grande parcela destas agressões pode ser eliminada através de uma preocupação maior com a qualidade do sistema [Staa 83, Rocha 83]. Em linhas gerais devemos preocupar para que o sistema seja:.
útil - atende às necessidades do usuário, é confiável, é robusto
utilizável - é fácil de utilizar, requer pouco treinamento, permite recuperação e correção de erros
monitorável - produz dados de acompanhamento do desempenho, do uso e de recuperação
evolutível - pode ser facilmente ampliado e/ou adaptado a novas condições operacionais
rentável - é econômico no uso de recursos, produz benefícios compatíveis com os custos, oferece pequeno risco.
Nunca é demais frisar a importância de confiabilidade, de robustez, do treinamento dos diversos usuários e da utilizabilidade (interface homem/sistema). A grande maioria das agressões está diretamente relacionada com baixa qualidade, especificamente nestes pontos.
Outro ponto importante para alcançar níveis de segurança satisfatórios é o cuidado com que deve ser realizada a manutenção (alteração) do sistema. Manutenção mal feita e/ou mal gerenciada é a principal razão para perda de qualidade no decorrer da vida do sistema. É pois fundamental que existam procedimentos de manutenção bem definidos, bem como é fundamental que o próprio sistema possua um sub-sistema de manutenção bem definido e cuidadosamente efetuado.
Um terceiro ponto essencial para assegurar níveis de segurança satisfatórios é a realização de auditoria de sistemas. Não basta que estejam redigidas boas especificações de qualidade do sistema e de requisitos de segurança; é essencial que seja verificado se tais especificações e requisitos são realmente satisfeitos pelo sistema. Como o sistema está em contínua evolução e pode ser vítima de adulterações, é fundamental que esta auditoria de sistemas seja realizada com alguma freqüência.
Existem diversos outros métodos e técnicas de dissuação. Mencioná-las e explorá-las encheria vários livros. Ao leitor interessado sugerimos as seguintes referências [Silva 77, Monteiro 82, Santos 80, Castilho 82].

4. Conclusão

Neste artigo esboçamos um modelo de classificação de agressões. Este modelo tem por objetivo auxiliar na especificação e verificação do grau de segurança alcançada pelo sistema. Mostramos, também, em linhas gerais, como proceder para especificar um nível de segurança satisfatório através da especificação de filtros capazes de detectar desvios das condições ideais de validade estrutural e semântica de entrada a cada processo.
Neste artigo não nos preocupamos com técnicas específicas de proteção, porém nos preocupamos com uma **atitude** profissional capaz de levar a sistemas satisfatoriamente seguros. O que quisemos enfatizar é a necessidade de especificar-se e projetar-se níveis de segurança ao invés de incorporar mecanismos de proteção à medida que vão sendo descobertas novas agressões que alcançaram êxito. É nosso argumento que a incorporação de proteção a posteriori é incompleta, imperfeita, ineficaz e cara, enquanto que desenvolvendo-se sistemas seguros por construção levará a sistemas suficientemente seguros e cujos custos são compatíveis com os riscos de ocorrência de agressões bem sucedidas.

Referências Bibliográficas

[Castilho 82] Castilho, J.M.V.
Especificações Formais para o Projeto de Aplicações de Bancos de

Dados; Tese de Doutorado, Informática, PUC/RJ; Rio de Janeiro, 1982

[Magalhães 79] Magalhães, J.A.C.
Técnicas para o Projeto e a Implementação de Sistemas Confiáveis; Dissertação de Mestrado, Informática, PC/RJ; Rio de Janeio; 1979

[Mandarino 82] Mandarino, P.R.N.
Segurança de Banco de Dados; Dissertação de Mestrado, Informática, PUC/RJ; Rio de Janeiro; 1982

[Monteiro 82] Monteiro, P.J.
Segurança de Sistemas Automatizados; Dissertação de Mestrado, Informática, PUC/RJ; Rio de Janeio; 1982

[Morais 80] Morais, J.B.
Auditoria de Sistemas Através do Computador; Dissertação de Mestrado, Informática, PUC/RJ; Rio de Janeiro; 1980

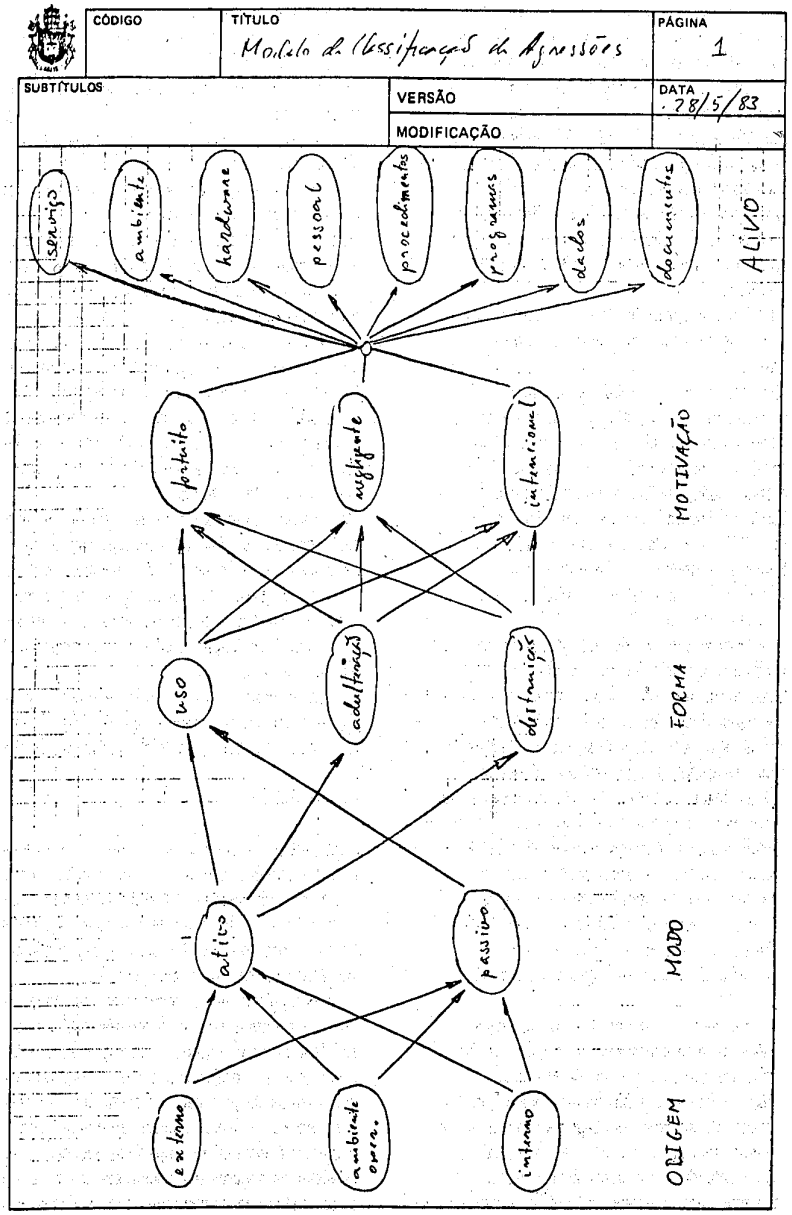
[Parente 81] Parente, F.A.F.
Auditoria de Sistema Automatizado; Dissertação de Mestrado, Informática, PUC/RJ, Rio de Janeiro, 1978

[Rocha 83] Rocha, A.R.C.
Um modelo de Avaliação de Qualidade de Especificações; Tese de Doutorado, Informática, PUC/RJ; Rio de Janeiro, 1983

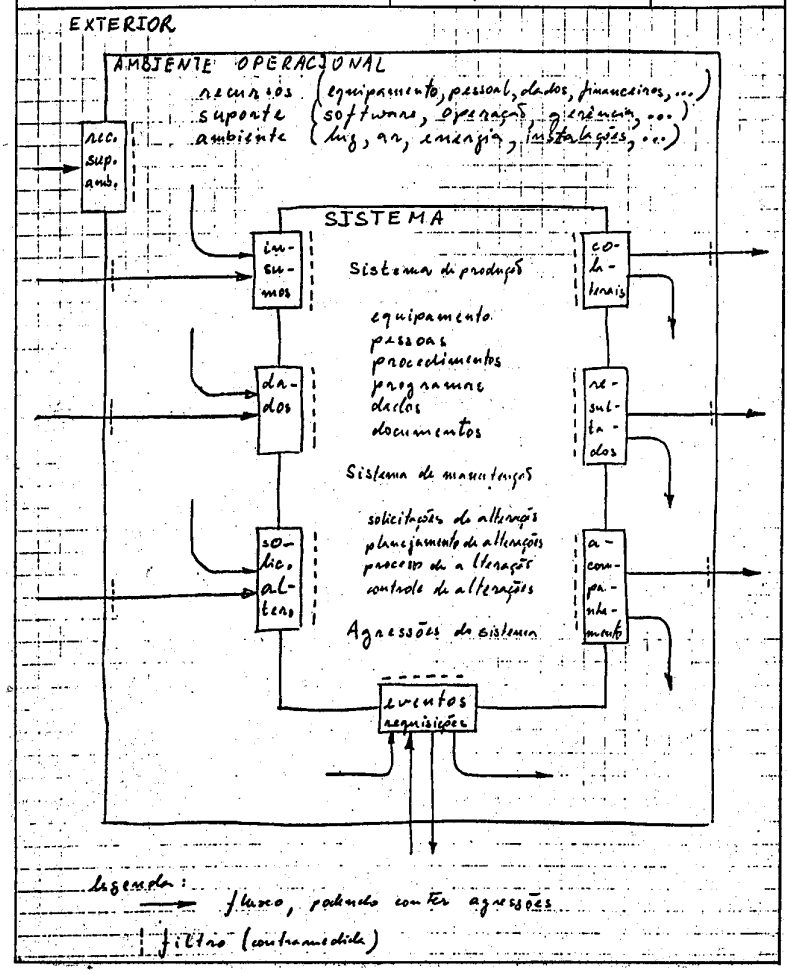
[Santos 80] Santos, C.L.
Caracterização Sistemática de Restrições de Integridade em Banco de Dados; Tese de Doutorado, Informática, PUC/RJ; Rio de Janeiro; 1980

[Silva 77] Silva, E.O.
Controle de Acesso e Uso das Informações de um Banco de Dados; Dissertação de Mestrado, Informática; PUC/RJ; Rio de Janeiro, 1977

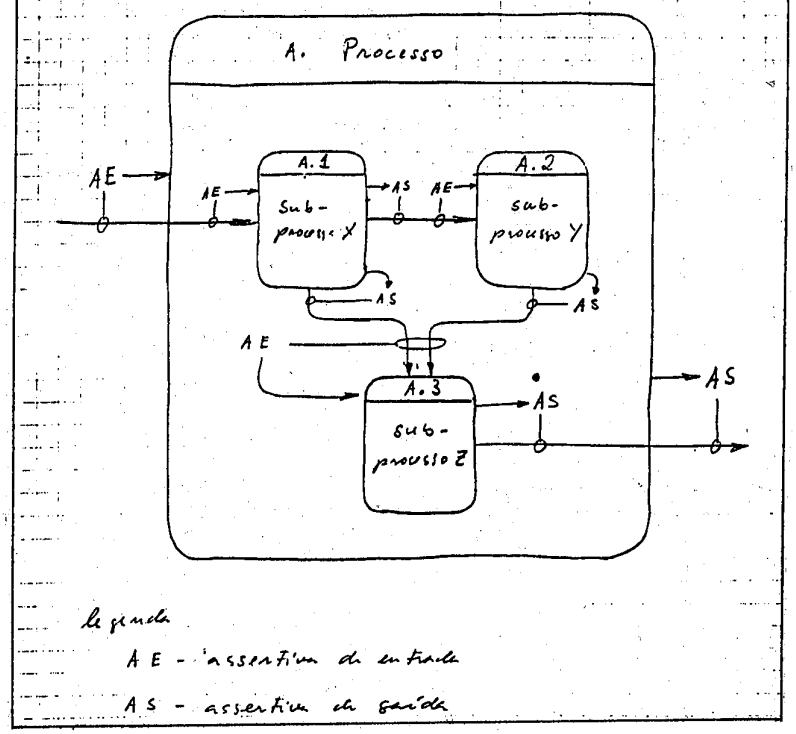
[Staa 83] Staa, A.v.
Engenharia de Programas; Livros Técnicos e Científicos; Rio de Janeiro; 1983.



CÓDIGO	TÍTULO	PÁGINA
	Visão integrada dos componentes e interfaces de um sistema automatizado	2
SUBTÍTULOS	VERSÃO	DATA 21/5/83
	MODIFICAÇÃO	



CÓDIGO	TÍTULO Ilustração da colocação de assertivas em uma especificação (fluxo de dados)	PÁGINA 3
SUBTÍTULOS	VERSÃO	DATA
	MODIFICAÇÃO	



legenda

AE - assertiva de entrada

AS - assertiva de saída