MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m19/2010/10.05.14.48-RPQ

USO DE MINERAÇÃO DE DADOS PARA ANÁLISE DO PROCESSO DE TESTE DE SOFTWARE

Érica Ferreira de Souza

Relatório final da disciplina Princípios e Aplicações de Mineração de Dados (CAP-359) do Programa de Pós-Graduação em Computação Aplicada, ministrada pelo professor Dr. Rafael Santos.

URL do documento original:
<http://urlib.net/ 8JMKD3MGP7W/38CDFNB >

INPE
São José dos Campos
2010

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/6921
Fax: (012) 3208-6919
E-mail: pubtc@sid.inpe.br

MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m19/2010/10.05.14.48-RPQ

USO DE MINERAÇÃO DE DADOS PARA ANÁLISE DO PROCESSO DE TESTE DE SOFTWARE

Érica Ferreira de Souza

Relatório final da disciplina Princípios e Aplicações de Mineração de Dados (CAP-359) do Programa de Pós-Graduação em Computação Aplicada, ministrada pelo professor Dr. Rafael Santos.

URL do documento original:
<http://urlib.net/ 8JMKD3MGP7W/38CDFNB >

INPE
São José dos Campos
2010

RESUMO

Identificar padrões de comportamento em um processo de teste aplicado em um projeto considerado complexo, e elaborar estratégias de reuso do processo, é de fundamental importância para a qualidade do produto gerado, e para o processo de tomada de decisão. Este trabalho tem como objetivo apresentar resultados preliminares de um estudo realizado em dados gerados a partir de um processo de teste que foi aplicado em um projeto de desenvolvimento de software real. Serão apresentados neste trabalho características de um processo de teste de software, os dados utilizados como estudo de caso, bem como os resultados preliminares alcançados com a aplicação de técnicas de mineração de dados.

SUMÁRIO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

ICAMMH Integração e Cooperação Amazônica para Modernização do Monitoramento Hidrológico

IPH Instituto de Pesquisas Hidráulicas

ITA Instituto Tecnológico da Aeronáutica

KDD *knowledge discovery in databases*

OTM3 *Organizational Testing Management Maturity Model*

SQL *Structured Query Language*

WEKA *Waikato Environment for Knowledge Analysis*

1 INTRODUÇÃO

Sistemas críticos possuem como característica fundamental a complexidade de seus modelos e implementações. Desta forma, testes tornam-se uma atividade criteriosa, pois falhas nesses sistemas podem causar perdas econômicas, danos ao meio ambiente e pode até colocar em risco vidas humanas. Realizar testes significa em seguir um processo para que a qualidade do sistema seja garantida. Neste sentido, é importante também garantir a qualidade do processo de teste que esta sendo utilizado no projeto.

A atividade de teste pode gerar muitos dados referentes aos testes exercitados no projeto de desenvolvimento. Muitos destes dados são armazenadas em bases de dados, podendo conter informações importantes para melhoria do processo de teste aplicado. Áreas para melhoria do processo podem ser identificadas a partir de históricos de dados de teste realizados por um determinado período de tempo, sendo possível fornecer uma base para melhorar estimativas futuras do processo de teste aplicado, bem como medidas para tomadas de decisão, para isso podem ser utilizadas técnicas de mineração de dados.

Neste trabalho foi utilizado como estudo de caso, dados de teste gerados a partir do processo de teste *Organizational Testing Management Maturity Model* (OTM3) (LAMAS et al., 2010), aplicado no projeto de Integração e Cooperação Amazônica para Modernização do Monitoramento Hidrológico (ICAMMH). O projeto ICAMMH está sendo desenvolvido em parceria com o Instituto Tecnológico da Aeronáutica (ITA) e o Instituto de Pesquisas Hidráulicas (IPH), com recursos oriundos do Fundo Setorial CT-HIDRO (MCT/FINEP), que contempla a modernização da rede de estações telemétricas da Amazônia.

Foram analisadas duas bases de dados, que contém dados gerados a partir do processo de teste, tais como, resultados dos casos de teste exercitados, gravidade do defeito, prioridade de correção, dentre outros. As bases de dados amazenam dados que foram cadastrados a partir de ferramentas de gestão de teste de software. Neste processo foram utilizadas as ferramentas *Mantis* (MANTIS, 2010) e *Testlink* (TESTLINK, 2010).

Considerando o contexto acima, este trabalho está relacionado com técnicas de mineração de dados, afim de levantar informações relevantes para melhoria do processo de teste que está sendo aplicado a um projeto de desenvolvimento de software. Foi

utilizado o algoritmo de associação Apriori (AGRAWAL et al., 1993) para identificar regras ou padrões nos dados e para aplicação do algoritmo, foi utilizada como auxilio a ferramenta *Waikato Environment for Knowledge Analysis* (WEKA) (WITTEN, 2005).

1.1 Objetivo do trabalho

A concepção deste trabalho tem como objetivos principais:

- Identificar regras de associação dos defeitos registrados em bases de dados com a utiliação do processo de teste OTM3; e
- Analisar a conformidade do processo OTM3 com as associações encontradas.

1.2 Organização do texto

A divisão dos capítulos está descrita a seguir:

- *CAPÍTULO 2 - MATERIAIS E MÉTODOS:* Este capítulo mostra de forma sucinta os principais conceitos relacionados à área de pesquisa deste trabalho em termos de: processo de testes de *software*; ferramentas de apoio ao teste; e técnicas de mineração de dados.
- *CAPÍTULO 3 - RESULTADOS:* São apresentados nesse Capítulo os principais resultados alcançados.
- *CAPÍTULO 4 - CONCLUSÃO:* Sintetiza as principais conclusões deste trabalho, subsequentes a este capítulo encontram-se as referências bibliográficas utilizadas na elaboração deste documento.

2 MATERIAIS E MÉTODOS

Este capítulo apresenta sucintamente os principais conceitos relacionados às atividades de teste de *software* que auxiliarão no entendimento deste trabalho, bem como os dados utilizados para análise e a técnica de mineração de dados aplicada.

2.1 Terminologia e conceitos

É fundamental aos ambientes de desenvolvimento de *software* a utilização de práticas de teste de uma forma sistemática para que a garantia de qualidade do *software* seja considerada. Diante desse fato, atividades relacionadas à Verificação, Validação e Teste (VV&T), vêm sendo introduzidas ao longo de todo o processo de desenvolvimento de *software*. Estas atividades têm como finalidade verificar a conformidade do *software* construído com o que foi especificado, e garantir que ele alcance um alto nível de qualidade. Tais atividades podem ser aplicadas por meio de algum processo de teste de software.

Em relação às atividades de teste de software, é interessante destacar alguns conceitos importantes:

- Teste de software: De uma forma simples, testar um software significa verificar, através de uma execução controlada, se o seu comportamento ocorre de acordo com o que foi especificado (MYERS, 2004).
- Processo de teste: O processo de teste de software representa uma estrutura formada por etapas, atividades, artefatos, papéis e responsabilidades que buscam a padronização das atividades de forma a minimizar os riscos causados por defeitos provenientes do processo de desenvolvimento (BASTOS et al., 2007).
- Técnicas de teste: O processo de teste é composto por diversas etapas ou fases, sendo que as principais delas são: *planejamento, especificação, execução e entrega*. Cada etapa de teste é realizada mediante uma série de técnicas de teste e diversos critérios pertencentes a cada uma delas, que pode diferenciar-se de acordo com a origem das informações utilizadas (BASTOS et al., 2007).

Assim como um processo de desenvolvimento de software, o processo de teste tam-

bém necessita ser revisto de forma a garantir uma maior visibilidade e controle do processo. Conforme citado anteriormente, o presente trabalho, utilizou como estudo de caso os dados gerados pelo processo de teste OTM3, aplicados no projeto ICAMMH.

O processo OTM3 surgiu da necessidade de prover um alinhamento Operacional, Tatíco e Estratégico, envolvendo estágios de melhoria contínua através de Padrões, Medições, Controle e Aprimoramento contínuo. A Figura 2.1 apresenta o processo OTM3. Cada mapa de processos do OTM3 descreve, visual e logicamente, a melhor forma de execução de cada uma das fases do processo do teste de software, a saber: Planejamento, Construção, Execução do teste, Finalização do teste e Aprovação. Maiores detalhes sobre o processo podem ser encontrados no trabalho de Lamas (2010).

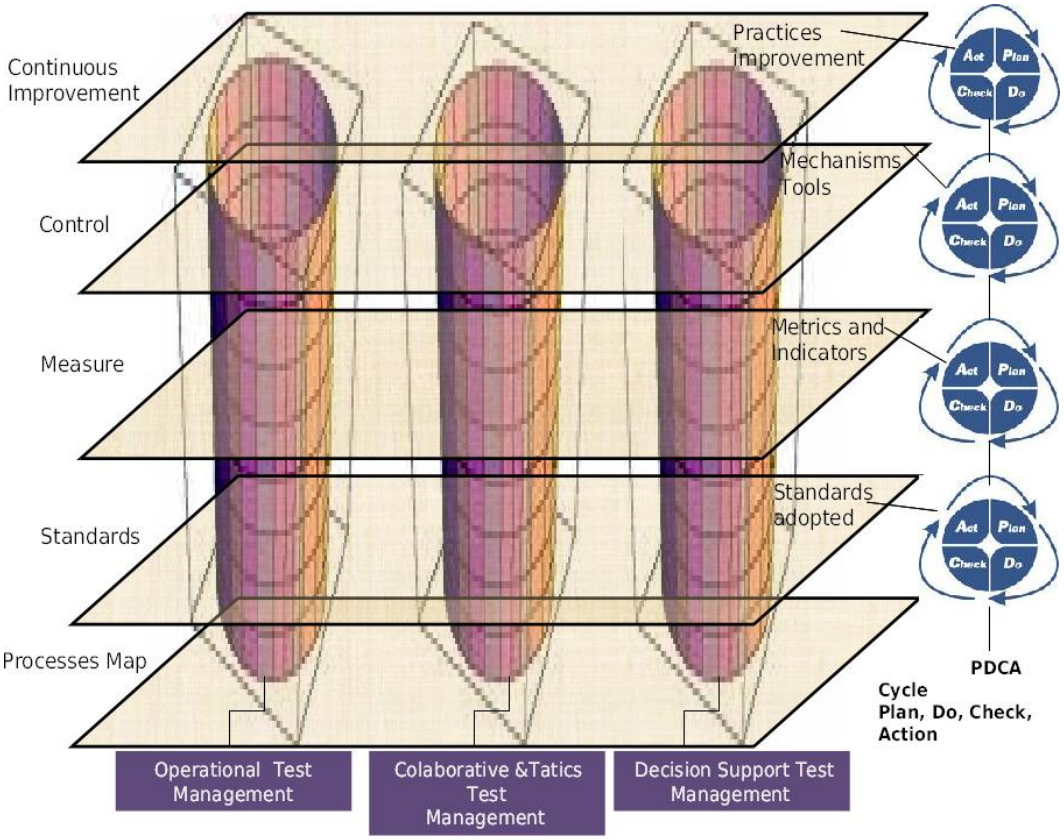

Figura 2.1 - Processo OTM3

Fonte: Lamas (2010)

Uma análise nos dados obtidos pelo processo foi realizada utilizando-se de técnicas e mineração de dados. A próxima seção descreve com mais detalhes, as caracteristícas e os dados analisados.

2.1.1 Dados analisados

Dentre as ferramentas de gestão de teste que são utilizadas pelo processo de teste OTM3, os dados gerados elas foram analisados neste trabalho:

- Ferramenta *TestLink*: Responsável pelo gerenciamento de casos de teste e execução. A Figura 2.2 apresenta um exemplo de tela de execução de um dos casos de teste e seu histórico de execuções.

Figura 2.2 - Tela do Testlink de execução dos casos de teste e o histórico de execuções.

- Ferramenta *Mantis*: Responsável pela gestão dos defeitos. A Figura 2.3 apresenta a tela de um cadastro de um defeito.

As ferramentas *Testlink* e *Mantis* podem ser integradas para melhorar o ambiente de testes, de forma a auxiliar na construção e no relacionamento entre as fases de testes, disponibilizando mecanismos de exportação de artefato e extração de métricas.

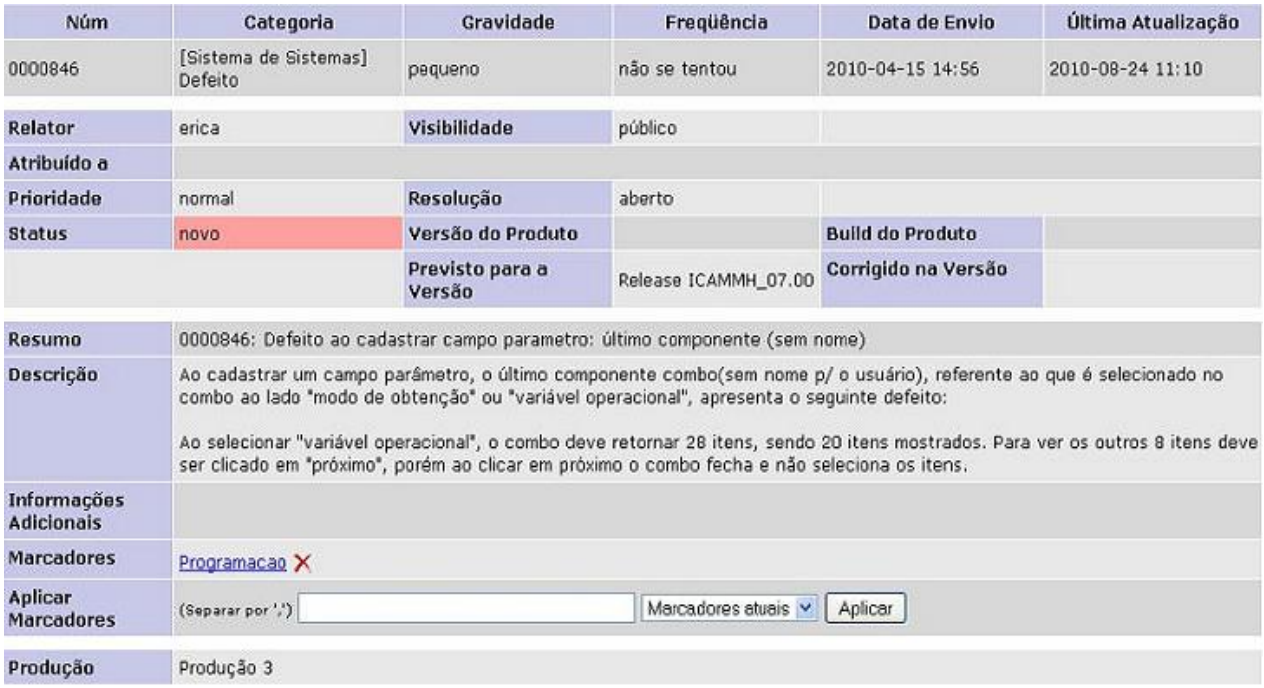

Figura 2.3 - Tela do Mantis referente ao cadastro de um defeito

Dentre os diversos atributos gerados pelas duas bases de dados das respectivas ferramentas, foi realizado um estudo de cada atributo, sua relação com o defeito, e sua importância. Dessa forma, foram selecionados por meio de *Structured Query Language* (SQL), os atributos relacionados apenas com a categoria de defeito na ferramenta *Mantis* e a qual caso de teste está relacionado na ferramenta *Testlink*.

Cada atributo é tratado no banco de dados com um respectivo identificador numérico. Por exemplo, o atributo *severity*, tem como um de seus dados discretos o valor *Pequeno*, que tem como identificador no banco dados o número 10, para *baixa* é 20 e para *normal* é 30. E o mesmo ocorre para todos os outros dados. A Tabela 2.1 apresenta os atributos selecionados com seus repectivos identificadores.

Tabela 2.1 - Atributos utilizados no processo de mineração de dados

Atributo	**Dados (Código)**	**Descrição**
priority	Nenhuma (10), baixa (20), normal (30), alta (40), urgente (50) e imediato (60)	Define o nível de prioridade para corrigir um defeito.

continua na próxima página

Tabela 2.1 - Atributos utilizados no processo de mineração de dados

Atributo	**Dados (Código)**	**Descrição**
severity	Nenhuma (10), recurso (20), trivial (30), texto (40), mínimo (50), pequeno (60), grande (70), travamentos (80), obstáculos (90) e N/A (100).	Define a severidade de um determinado defeito encontrado.
reproducibility	Sempre (10), às vezes (20), aleatório (30), não se tentou (40), incapaz de reproduzir (50), N/A (60).	Define se o defeito pode ser reproduzido.
status_mantis	Aberto (10), corrigido (20), reaberto (30), incapaz de reproduzir (40), não corrigível (50), duplicável (60), não é um caso (70), suspenso (80) e não será corrigido (90), N/A (100).	Define o *status* de uma determinada requisição na ferramenta *Mantis*. Esta requisição pode ser um defeito ou uma atividade. Este trabalho leva em conta somente os registros relacionados a defeitos.
resolution	Resolvido (10), fechado (20) e cancelado (30).	Define o *status* final de uma requisição, seja esta uma atividade ou um defeito.
fixed_in_version	Release ICAMMH_01.00 (1), Release ICAMMH_02.00 (2), Release ICAMMH_03.00 (3), Release ICAMMH_04.00 (4), Release ICAMMH_05.00 (5)	Define qual *release* está prevista a correção.
target_version	Release ICAMMH_01.00 (1), Release ICAMMH_02.00 (2), Release ICAMMH_03.00 (3), Release ICAMMH_04.00 (4), Release ICAMMH_05.00 (5).	Define qual *release* surgiu o defeito.
tester_id	Admin (1), Erica (2), Pascaly (4), Juliano (5), Breno (6), Etiene (7), Marcos (8), Sakaue (9), Melis (10), Juliana (11) e Felipe (12).	Define os usuários cadastrados na ferramenta *Testlink*.

continua na próxima página

Tabela 2.1 - Atributos utilizados no processo de mineração de dados

Atributo	**Dados (Código)**	**Descrição**
status_testlink	Passou (p), falhou (f), bloqueado (d).	Define os *status* de cada defeito na ferramenta *Testlink*.
importance	Médio (1), alto (2), baixo (3).	Importância da execução do caso de teste.

Da consulta realizada pela SQL, foram retornados na consulta para fazer o estudos 246 registros e 10 atributos. A Tabela 2.2 apresenta o formato apenas dos 15 primeiros registros retornados.

Tabela 2.2 - Tabela com formato dos dados analisados (15 primeiros registros)

priority	severity	repro.	st_mantis	resolution	fixed_ver	target_ver	tester	st_testlink	import.
30	50	100	90	20	2	1	7	p	2
40	60	10	80	20	2	1	7	p	2
30	60	70	90	20	2	1	7	p	3
30	50	70	90	20	2	1	7	p	3
30	50	70	90	20	2	1	7	p	3
30	50	70	90	20	2	1	7	p	3
30	50	70	90	20	2	1	7	p	3
30	60	70	90	20	2	1	7	p	3
30	50	70	90	20	2	1	7	p	3
30	60	70	90	20	2	1	7	p	1
40	60	10	90	20	2	1	7	p	3
40	60	10	90	20	2	1	7	p	3
40	60	10	90	20	2	1	7	p	3

Depois de selecionados o conjunto de dados a ser trabalhado, estes foram convertidos para o formato *.arff*, o qual é reconhecido pela ferramenta WEKA utilizada no processo de mineração. A próxima seção descreve brevemente o algoritmo Apriori, o qual foi utilizado para realização da mineração nos dados.

2.2 Mineração de dados

A descoberta de conhecimentos em bancos de dados *knowledge discovery in databases* (KDD) é caracterizado como sendo um processo interativo e iterativo, composto por várias etapas interligadas (FAYYAD et al., 1996). Essas etapas vão deste a definição do domínio, seleção, preparação e transformação dos dados, até a etapa de mineração dos dados, onde padrões podem ser encontrados e analisados para

tornarem-se conhecimento útil.

Mineração de dados reúne uma série de técnicas, capazes de descobrir novas informações em grandes bancos de dados. Umas das técnicas mais conhecidas é a mineração por Regras de Associação, que tem como objetivo a identificação de padrões de comportamento ao conjunto de dados que frequentemente ocorrem de forma conjunta na base de dados e formar regras a partir destes conjuntos. As regras de associação, quando aplicadas a um conjunto de dados, possibilitam encontrar regras do tipo do X–>Y, ou seja, transações do banco de dados que contêm X tendem a conter Y, e parâmetros de confiança sobre a regra são fornecidos (WITTEN, 2005).

Um dos algoritmos utilizados que realiza esta técnica é o algoritmo Apriori. O algoritmo Apriori pode trabalhar com um número grande de atributos, gerando várias combinações entre eles. Esse algoritmo gera um conjunto de itens frequentes a cada uma de suas passagens. São definidos parâmetros que determinam quais associações são interessantes ou não.

Neste trabalho foi utilizado a técnica de Regra de Associação, juntamente com o algoritmo Apriori. Para geração das regras de associação com este algoritmo, foi utilizada como auxilio a ferramenta WEKA. O Capítulo 3 apresenta os principais resultados obtidos.

3 RESULTADOS

Nesta seção são apresentados alguns resultados obtidos a partir da ferramenta WEKA. Podem-se observar pela Figura 3.1, os elementos de pré-processamento, informações dos atributos, instâncias e um histograma dos dados utilizados para as analises.

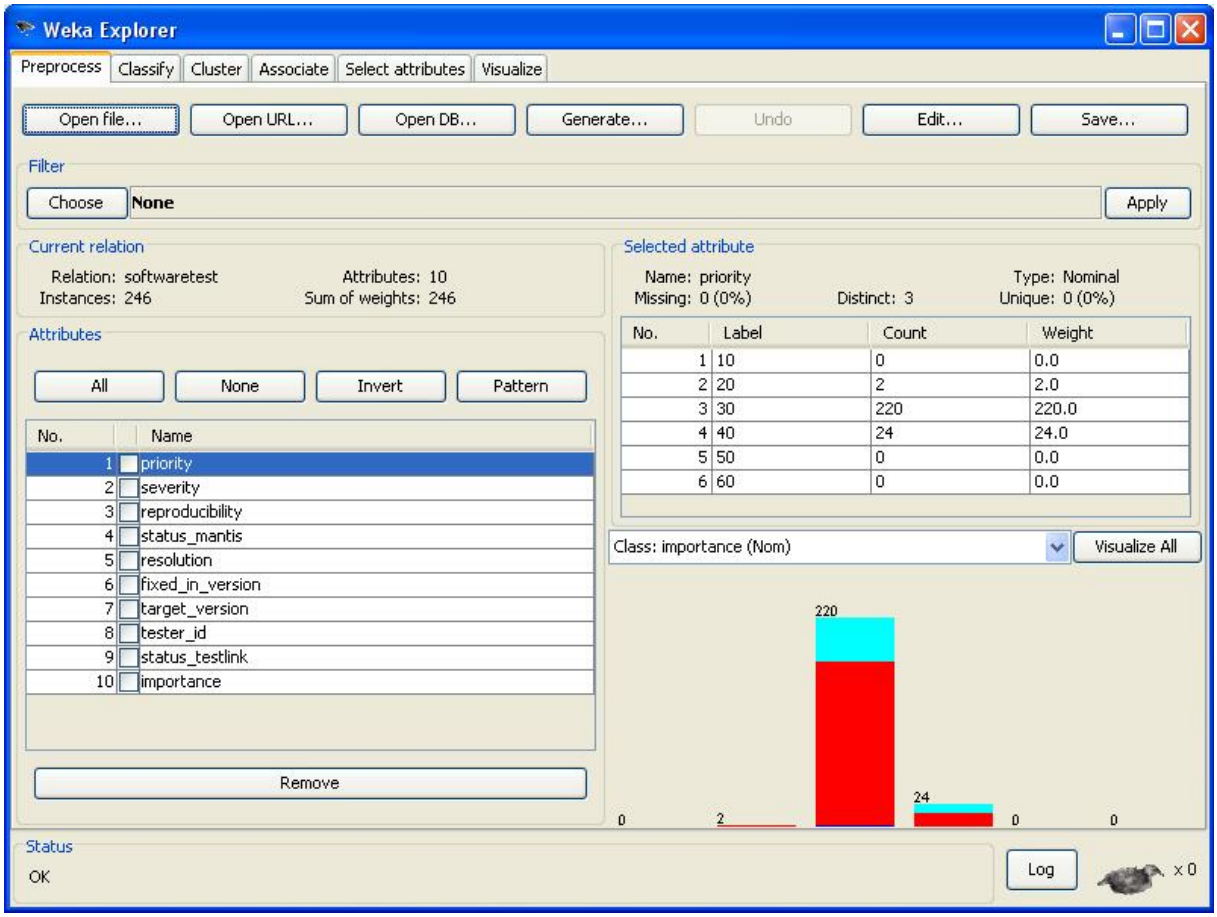

Figura 3.1 - Tela do Explorer da ferramenta WEKA.

Conforme mencionado nas seções anteriores, neste trabalho foi utilizado o algoritmo Apriori, buscando identificar regras de associações entre os atributos escolhidos para analise. Dos parâmetros utilizados no algoritmo tem-se:

Número de regras geradas: 10 regras
Confiança: média de 98%

Outros parâmetros são apresentados na Figura 3.2 abaixo.

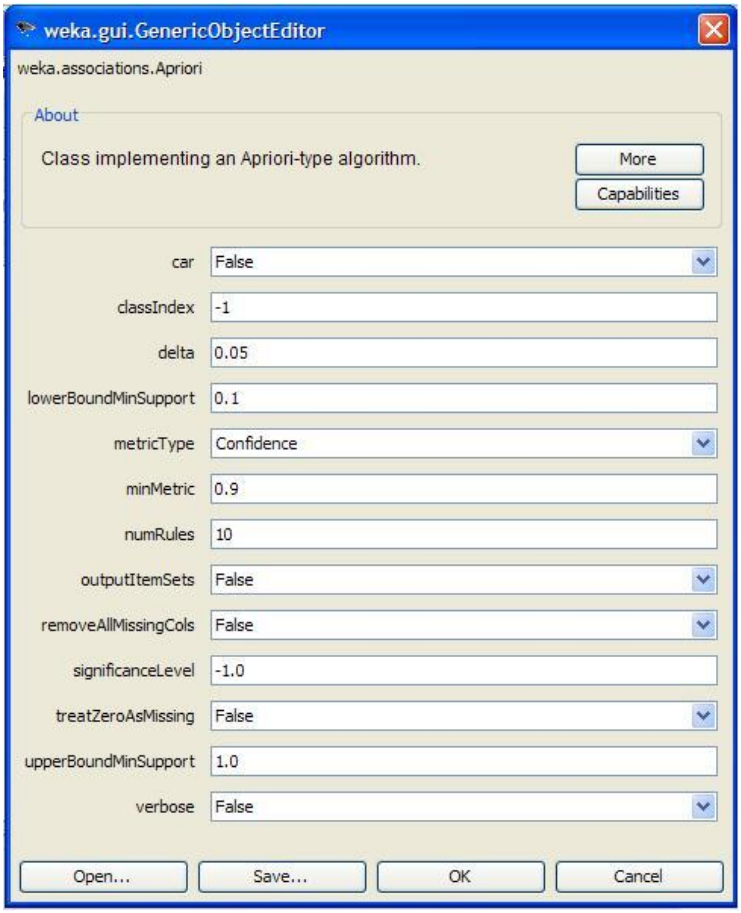

Figura 3.2 - Parâmentros utilizados

A listagem abaixo apresenta os resultados para as 10 melhores regras de associação geradas a partir da base de dados utilizada.

```
Apriori
=======
Minimum support: 0.55 (135 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 9

Generated sets of large itemsets:
```

```
Size of set of large itemsets L(1): 8
Size of set of large itemsets L(2): 19
Size of set of large itemsets L(3): 13

Best rules found:

 1. severity=50 176 ==> priority=30 176 conf:(1)
 2. status_mantis=80 160 ==> resolution=20 160  conf:(1)
 3. severity=50 importance=2 156 ==> priority=30 156 conf:(1)
 4. severity=50 tester_id=7 148 ==> priority=30 148 conf:(1)
 5. status_mantis=80 importance=2 147 ==> resolution=20 147 conf:(1)
 6. priority=30 status_mantis=80 145 ==> resolution=20 145 conf:(1)
 7. severity=50 resolution=20 144 ==> priority=30 144 conf:(1)
 8. severity=50 reproducibility=10 136 ==> priority=30 136 conf:(1)
 9. resolution=20 importance=2 157 ==> status_mantis=80 147 conf:(0.94)
10. status_mantis=80 160 ==> importance=2 147 conf:(0.92)
```

Analisando as 10 regras pode-se chegar a algumas conclusões. A primeira regra, por exemplo, mostra que de 176 defeitos registrados, a severidade do defeito é 50 e a prioridade de correção é 30, o que realmente tem lógica, pois conforme a Tabela 2.1, apresentada no Capítulo 2, a severidade 50 significa ter uma gravidade *mínima* enquanto que prioridade de correção 30 significa ser *normal* em relação aos defeitos encontrados.

Pela regra 3, embora o cadastro destes casos de teste na ferramenta *Testlink*, tenham sido classificados com um nível alto, ao ser cadastrado o defeito gerado na ferramenta *Mantis*, a severidade e a prioridade para correção do defeito destes casos de teste são menores, sendo mínimo e normal respectivamente. Então, embora a execução destes casos de testes sejam de prioridade alta, o defeito que foi gerados por eles não foi tão grave.

Outro caso interessante é a regra 5 que mostra casos de teste que foram suspensos na ferramenta *Mantis* (*status_mantis=80*), e a resolução dos casos de testes que foram fechados (*resolution=20*), embora os casos de teste criados, que fizeram gerar tais defeitos, fossem de alta importância. Analisando as datas e os *status* dos defeitos, pela ferramenta, realmente houve uma fase no projeto em que diversos defeitos

foram suspensos, pois a *release* que foi passada para equipe de teste, foi passada sem ainda estar apta para realização de testes, ocasionado a geração de diversos defeitos. Devido ao grande número de defeitos gerados, os casos de teste para essa *release*, foram suspensos.

Isso mostra que a forma como estava sendo seguido o plano de teste para criação e execução dos casos de teste, e preenchimento dos defeitos gerados a partir destes casos, estão coerentes com o processo de teste aplicado.

Se aumentarmos o número de regras geradas para 100, por exemplo, podemos obter novas regras a serem analisadas. Dentre as 100 regras geradas, foram escolhidas algumas que apresentaram um comportamento mais interessante em relação às outras, para serem discutidas com mais detalhes. As 100 regras geradas encontram-se no Apêndice A.

Algumas regras interessantes:

```
Apriori
=======

Minimum support: 0.4 (98 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 12

Generated sets of large itemsets:
Size of set of large itemsets L(1): 9
Size of set of large itemsets L(2): 32
Size of set of large itemsets L(3): 48
Size of set of large itemsets L(4): 32
Size of set of large itemsets L(5): 11
Size of set of large itemsets L(6): 1

Best rules found:

...
11. severity=50 reproducibility=10 importance=2 129 ==>
priority=30 129 conf:(1)
```

```
...
33. severity=50 target_version=5 108 ==> priority=30 108 conf:(1)
34. priority=30 target_version=5 108 ==> severity=50 108 conf:(1)
35. priority=30 target_version=5 108 ==> importance=2 108 conf:(1)
36. severity=50 target_version=5 108 ==> importance=2 108 conf:(1)
37. severity=50 reproducibility=10 resolution=20 108 ==>
...
67. target_version=5 111 ==> priority=30 108 conf:(0.97)
68. target_version=5 111 ==> severity=50 108 conf:(0.97)
69. target_version=5 111 ==> priority=30 severity=50 108 conf:(0.97)
70. target_version=5 importance=2 111 ==> priority=30 108 conf:(0.97)
71. target_version=5 111 ==> priority=30 importance=2 108 conf:(0.97)
...
```

As 10 primeiras regras são bem parecidas com as já apresentadas anteriormente. A regra 11 também apresenta clareza na classificação, onde 129 registros, apresentam defeito que foram cadastrados que mostram ter severidade mínima, prioridade normal, é um tipo de defeito em que sua reprodutividade ocorre sempre.

Analisando as regras que apresentam informações sobre a versão em que os defeitos ocorreram (atributo *target_version*), diversas regras apresentaram fortemente classificações com *target_version* sendo igual a 5. Após um levantamento dos dados do projeto, em relação a *release* 5, verificou-se que esta é a mesma *release* que foi liberada para os testadores, sem estar a nível de teste. Podemos perceber isso nas regras 30 à 42 e de 67 à 75.

Das regras geradas percebeu-se a coerência entre as associações que foram apresentadas pelo algoritmo Apriori. O mesmo ocorre com os valores que foram dados a cada atributo, mostrando também ter coerência entre as informações e o processo de teste aplicado.

4 CONCLUSÃO

Este Capítulo apresenta as principais conclusões encontradas durante o estudo. Ao final, também são apresentadas algumas sugestões para realização de trabalhos futuros.

Este trabalho teve como objetivo principal identificar padrões de comportamento dos defeitos registrados em bases de dados através de ferramentas de gestão de defeitos, e analisar a conformidade das regras de associação geradas com o processo de teste OTM3 utilizado no projeto de desenvolvimento ICAMMH.

O processo de teste, como qualquer outro processo deve ser revisto continuamente, de forma a ampliar sua atuação e possibilitar aos profissionais uma maior visibilidade e organização dos seus trabalhos. Este trabalho mostrou alguns resultados preliminares de uma análise que pode ser realizada em um processo de teste. Os primeiros resultados obtidos mostraram ser coerentes com o processo aplicado. No entanto, a análise apresentada não é conclusiva, mas é um bom ponto de partida para estudos afins, com o intuito de se alcançar uma melhor conclusão sobre os resultados, e mostrar a importância de se saber a qualidade de um processo de teste e também propor melhorias.

4.1 Sugestões para trabalhos futuros

Para a continuidade deste trabalho, destacam-se algumas sugestões para trabalhos futuros, dentre elas:

- Analisar de forma mais detalhada os resultados obtidos;
- Fazer o mesmo estudo em diferentes projetos como estudo de caso e comparar os resultados alcançados; e
- Utilizar de novas técnicas de mineração de dados.

REFERÊNCIAS BIBLIOGRÁFICAS

AGRAWAL, R.; IMIELINSKI, T.; SWAMI, A. Mining association rules between sets of itens in large database. **Conference Management of Data (SIGMOD-93)**, Washington DC, USA, p. 207–216, 1993. 7

BASTOS, A.; RIOS, E.; CRISTALLI, R.; MOREIRA, T. **Base de conhecimento em testes de software**. 2. ed. São Paulo: Ed. Martins, 2007. 8

FAYYAD, U.; GREGORY, P.; P.SMYTH, P. Knowledge discovery and data mining: Towards a unifying framework. **Proceeding of the Second International Conference on Knowledge Discovery and Data Mining**, Portland, Oregon, 1996. 13

LAMAS, E. **Uma estrutura de maturidade operacional para a gestão de teste de software aplicada a um projeto de monitoramento hidrológico**. Dissertação (Mestrado) — Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, 2010. 9

LAMAS, E.; FERREIRA, E.; NASCIMENTO, M. R.; DIAS, L. A. V.; SILVEIRA, F. F. Organizational testing management maturity model for a software product line. **Seventh International Conference on Information Technology, IEEE Computer Society**, p. 1026–1031, 2010. 6

MANTIS. 2010. Último acesso: 10/09/2010. Disponível em: <http://www.mantisbt.org/>. 6

MYERS, G. J. **The art of software testing**. 2. ed. Canada: John Wiley and Sons, 2004. 8

TESTLINK. 2010. Último acesso: 10/09/2010. Disponível em: <http://www.teamst.org/>. 6

WITTEN, E. F. I. H. **Data mining: practical machine learning tools and techniques**. 2. ed. San Francisco: Morgan Kaufmann, 2005. 7, 14

A APÊNDICE A - 100 REGRAS GERADAS USANDO O MÉTODO APRIORI

Apriori =======

Minimum support: 0.4 (98 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 12

Generated sets of large itemsets:
Size of set of large itemsets L(1): 9
Size of set of large itemsets L(2): 32
Size of set of large itemsets L(3): 48
Size of set of large itemsets L(4): 32
Size of set of large itemsets L(5): 11
Size of set of large itemsets L(6): 1

Best rules found:

1. severity=50 176 ==> priority=30 176 conf:(1)
2. status_mantis=80 160 ==> resolution=20 160 conf:(1)
3. severity=50 importance=2 156 ==> priority=30 156 conf:(1)
4. severity=50 tester_id=7 148 ==> priority=30 148 conf:(1)
5. status_mantis=80 importance=2 147 ==> resolution=20 147 conf:(1)
6. priority=30 status_mantis=80 145 ==> resolution=20 145 conf:(1)
7. severity=50 resolution=20 144 ==> priority=30 144 conf:(1)
8. severity=50 reproducibility=10 136 ==> priority=30 136 conf:(1)
9. reproducibility=10 status_mantis=80 133 ==> resolution=20 133 conf:(1)
10. priority=30 status_mantis=80 importance=2 132 ==> resolution=20 132 conf:(1)
11. severity=50 reproducibility=10 importance=2 129 ==> priority=30 129 conf:(1)
12. severity=50 tester_id=7 importance=2 128 ==> priority=30 128 conf:(1)
13. severity=50 status_mantis=80 126 ==> priority=30 126 conf:(1)
14. severity=50 status_mantis=80 126 ==> resolution=20 126 conf:(1)
15. status_mantis=80 tester_id=7 126 ==> resolution=20 126 conf:(1)
16. severity=50 status_mantis=80 resolution=20 126 ==> priority=30 126 conf:(1)
17. priority=30 severity=50 status_mantis=80 126 ==> resolution=20 126 conf:(1)
18. severity=50 status_mantis=80 126 ==> priority=30 resolution=20 126 conf:(1)
19. severity=50 resolution=20 importance=2 125 ==> priority=30 125 conf:(1)
20. reproducibility=10 status_mantis=80 importance=2 124 ==> resolution=20 124 conf:(1)
21. severity=50 status_mantis=80 importance=2 119 ==> priority=30 119 conf:(1)

22. priority=30 reproducibility=10 status_mantis=80 119 ==> resolution=20 119 conf:(1)
23. severity=50 status_mantis=80 importance=2 119 ==> resolution=20 119 conf:(1)
24. severity=50 status_mantis=80 resolution=20 importance=2 119 ==> priority=30 119 conf:(1)
25. priority=30 severity=50 status_mantis=80 importance=2 119 ==> resolution=20 119 conf:(1)
26. severity=50 status_mantis=80 importance=2 119 ==> priority=30 resolution=20 119 conf:(1)
27. severity=50 resolution=20 tester_id=7 118 ==> priority=30 118 conf:(1)
28. severity=50 reproducibility=10 tester_id=7 115 ==> priority=30 115 conf:(1)
29. status_mantis=80 tester_id=7 importance=2 113 ==> resolution=20 113 conf:(1)
30. priority=30 status_mantis=80 tester_id=7 112 ==> resolution=20 112 conf:(1)
31. target_version=5 111 ==> importance=2 111 conf:(1)
32. priority=30 reproducibility=10 status_mantis=80 importance=2 110 ==> resolution=20 110 conf:(1)
33. severity=50 target_version=5 108 ==> priority=30 108 conf:(1)
34. priority=30 target_version=5 108 ==> severity=50 108 conf:(1)
35. priority=30 target_version=5 108 ==> importance=2 108 conf:(1)
36. severity=50 target_version=5 108 ==> importance=2 108 conf:(1)
37. severity=50 reproducibility=10 resolution=20 108 ==> priority=30 108 conf:(1)
38. severity=50 target_version=5 importance=2 108 ==> priority=30 108 conf:(1)
39. priority=30 target_version=5 importance=2 108 ==> severity=50 108 conf:(1)
40. priority=30 severity=50 target_version=5 108 ==> importance=2 108 conf:(1)
41. severity=50 target_version=5 108 ==> priority=30 importance=2 108 conf:(1)
42. priority=30 target_version=5 108 ==> severity=50 importance=2 108 conf:(1)
43. severity=50 reproducibility=10 tester_id=7 importance=2 108 ==> priority=30 108 conf:(1)
44. reproducibility=10 status_mantis=80 tester_id=7 107 ==> resolution=20 107 conf:(1)
45. severity=50 status_testlink=p 106 ==> priority=30 106 conf:(1)
46. severity=50 reproducibility=10 status_mantis=80 102 ==> priority=30 102 conf:(1)
47. severity=50 reproducibility=10 status_mantis=80 102 ==> resolution=20 102 conf:(1)
48. severity=50 reproducibility=10 status_mantis=80 resolution=20 102 ==> priority=30 102 conf:(1)
49. priority=30 severity=50 reproducibility=10 status_mantis=80 102 ==> resolution=20 102 conf:(1)
50. severity=50 reproducibility=10 status_mantis=80 102 ==> priority=30 resolution=20 102 conf:(1)
51. severity=50 reproducibility=10 resolution=20 importance=2 102 ==> priority=30 102 conf:(1)
52. status_mantis=80 status_testlink=p 101 ==> resolution=20 101 conf:(1)

53. severity=50 status_mantis=80 tester_id=7 100 ==> priority=30 100 conf:(1)

54. severity=50 status_mantis=80 tester_id=7 100 ==> resolution=20 100 conf:(1)

55. severity=50 status_mantis=80 resolution=20 tester_id=7 100 ==> priority=30 100 conf:(1)

56. priority=30 severity=50 status_mantis=80 tester_id=7 100 ==> resolution=20 100 conf:(1)

57. severity=50 status_mantis=80 tester_id=7 100 ==> priority=30 resolution=20 100 conf:(1)

58. reproducibility=10 target_version=5 99 ==> importance=2 99 conf:(1)

59. severity=50 resolution=20 tester_id=7 importance=2 99 ==> priority=30 99 conf:(1)

60. priority=30 status_mantis=80 tester_id=7 importance=2 99 ==> resolution=20 99 conf:(1)

61. severity=50 reproducibility=10 status_mantis=80 importance=2 98 ==> riority=30 98 conf:(1)

62. severity=50 reproducibility=10 status_mantis=80 importance=2 98 ==> resolution=20 98 conf:(1)

63. reproducibility=10 status_mantis=80 tester_id=7 importance=2 98 ==> resolution=20 98 conf:(1)

64. severity=50 reproducibility=10 status_mantis=80 resolution=20 importance=2 98 ==> priority=30 98 conf:(1)

65. priority=30 severity=50 reproducibility=10 status_mantis=80 importance=2 98 ==> resolution=20 98 conf:(1)

66. severity=50 reproducibility=10 status_mantis=80 importance=2 98 ==> priority=30 resolution=20 98 conf:(1)

67. target_version=5 111 ==> priority=30 108 conf:(0.97)

68. target_version=5 111 ==> severity=50 108 conf:(0.97)

69. target_version=5 111 ==> priority=30 severity=50 108 conf:(0.97)

70. target_version=5 importance=2 111 ==> priority=30 108 conf:(0.97)

71. target_version=5 111 ==> priority=30 importance=2 108 conf:(0.97)

72. target_version=5 importance=2 111 ==> severity=50 108 conf:(0.97)

73. target_version=5 111 ==> severity=50 importance=2 108 conf:(0.97)

74. target_version=5 importance=2 111 ==> priority=30 severity=50 108 conf:(0.97)

75. target_version=5 111 ==> priority=30 severity=50 importance=2 108 conf:(0.97)

76. severity=50 reproducibility=10 status_mantis=80 102 ==> importance=2 98 conf:(0.96)

77. priority=30 severity=50 reproducibility=10 status_mantis=80 102 ==> importance=2 98 conf:(0.96)

78. severity=50 reproducibility=10 status_mantis=80 102 ==> priority=30 importance=2 98 conf:(0.96)

79. severity=50 reproducibility=10 resolution=20 importance=2 102 ==> status_mantis=80 98 conf:(0.96)

80. severity=50 reproducibility=10 status_mantis=80 resolution=20 102 ==> importance=2 98 conf:(0.96)

81. severity=50 reproducibility=10 status_mantis=80 102 ==> resolution=20 importance=2 98 conf:(0.96)

82. priority=30 severity=50 reproducibility=10 resolution=20 importance=2 102 ==> status_-mantis=80 98 conf:(0.96)

83. priority=30 severity=50 reproducibility=10 status_mantis=80 resolution=20 102 ==> importance=2 98 conf:(0.96)

84. severity=50 reproducibility=10 resolution=20 importance=2 102 ==> priority=30 status_-mantis=80 98 conf:(0.96)

85. severity=50 reproducibility=10 status_mantis=80 resolution=20 102 ==> priority=30 importance=2 98 conf:(0.96)

86. priority=30 severity=50 reproducibility=10 status_mantis=80 102 ==> resolution=20 importance=2 98 conf:(0.96)

87. severity=50 reproducibility=10 status_mantis=80 102 ==> priority=30 resolution=20 importance=2 98 conf:(0.96)

88. severity=50 resolution=20 importance=2 125 ==> status_mantis=80 119 conf:(0.95)

89. priority=30 severity=50 resolution=20 importance=2 125 ==> status_mantis=80 119 conf:(0.95)

90. severity=50 resolution=20 importance=2 125 ==> priority=30 status_mantis=80 119 conf:(0.95)

91. severity=50 reproducibility=10 136 ==> importance=2 129 conf:(0.95)

92. priority=30 severity=50 reproducibility=10 136 ==> importance=2 129 conf:(0.95)

93. severity=50 reproducibility=10 136 ==> priority=30 importance=2 129 conf:(0.95)

94. severity=50 status_mantis=80 126 ==> importance=2 119 conf:(0.94)

95. priority=30 severity=50 status_mantis=80 126 ==> importance=2 119 conf:(0.94)

96. severity=50 status_mantis=80 126 ==> priority=30 importance=2 119 conf:(0.94)

97. severity=50 status_mantis=80 resolution=20 126 ==> importance=2 119 conf:(0.94)

98. severity=50 status_mantis=80 126 ==> resolution=20 importance=2 119 conf:(0.94)

99. priority=30 severity=50 status_mantis=80 resolution=20 126 ==> importance=2 119 conf:(0.94)

100. severity=50 status_mantis=80 resolution=20 126 ==> priority=30 importance=2 119 conf:(0.94)