SMARTCLEAN: UMA FERRAMENTA PARA A LIMPEZA INCREMENTAL DE DADOS

Paulo Jorge Oliveira, Maria de Fátima Rodrigues
Departamento de Engenharia Informática /
GECAD – Grupo de Investigação em Engenharia do Conhecimento e Apoio à Decisão
Instituto Superior de Engenharia – Instituto Politécnico do Porto
{pjo,mfc}@isep.ipp.pt

Pedro Rangel Henriques
Departamento de Informática / gEPL – grupo de Especificação e Processamento de Linguagens
Universidade do Minho
prh@di.uminho.pt

Resumo: Neste artigo apresenta-se a ferramenta *SmartClean*, destinada à detecção e correcção de problemas de qualidade dos dados. Comparativamente às ferramentas actualmente existentes, o *SmartClean* possui a mais-valia de não obrigar a que a sequência de execução das operações seja especificada pelo utilizador. Para tal, foi concebida uma sequência segundo a qual os problemas são manipulados (*i.e.*, detectados e corrigidos). A existência da sequência suporta ainda a execução incremental das operações. No artigo, a arquitectura subjacente à ferramenta é exposta, sendo detalhados os seus componentes. A validade da ferramenta e, consequentemente, da arquitectura é comprovada através da apresentação do caso de estudo efectuado. Apesar do *SmartClean* possuir potencialidades de limpeza de dados noutros níveis (*e.g.*, relação), no artigo apenas são descritas as relativas ao nível do valor individual do atributo.

Palavras chave: Limpeza de Dados, Detecção, Correcção, Problemas de Qualidade dos Dados, Arquitectura, Ferramenta

1. INTRODUÇÃO

Apenas recentemente as entidades públicas e privadas começaram a aperceber-se do real valor dos dados. Como consequência, a sua exploração passou a assumir um papel cada vez mais importante. Todas as ferramentas que realizam exploração de dados (*e.g.*, análise multi-dimensional) requerem um elevado grau de qualidade dos dados. Se os dados forem de má qualidade, isso reflecte-se nos resultados (princípio "lixo entra, lixo sai"). Por este motivo, é muito importante "limpar" os dados, *i.e.*, detectar e corrigir os seus problemas de qualidade.

Comercialmente, há diversas ferramentas de limpeza de dados (*e.g.*, [1]; [2]; [3]). Fruto da investigação, a comunidade académica também desenvolveu algumas ferramentas com o mesmo fim, sendo as mais relevantes: [4]; [5]; e, [6]. Apesar das diferenças existentes entre as diversas ferramentas, em todas compete ao utilizador especificar: (*i*) as Operações de Detecção (OD) e Operações de Correcção (OC) a efectuar; e, (*ii*) a ordem pela qual são executadas.

Neste artigo apresenta-se a ferramenta de limpeza de dados *SmartClean*. A originalidade desta ferramenta reside no facto do utilizador não necessitar de especificar a ordem de execução das operações. As operações são executadas de acordo com uma sequência preestabelecida. Esta sequência, além de reflectir as dependências de detecção e correcção entre os diversos tipos de Problemas de Qualidade dos Dados (PQD), também permite a sua manipulação incremental. Ainda que o *SmartClean* suporte a manipulação de PQD nos diferentes níveis de granularidade (*i.e.*, tuplo (*e.g.*, violação de restrição de integridade); relação (*e.g.*, violação de dependência funcional); e, multi-relação (*e.g.*, violação de integridade referencial)), devido a restrições de espaço, neste artigo apenas é apresentado o seu modo de operação ao nível do valor individual do atributo.

O artigo encontra-se organizado da seguinte forma. Na Secção 2 são enumerados os PQD que ocorrem ao nível do valor individual do atributo. Na Secção 3 expõe-se em detalhe a arquitectura do *SmartClean*. Na Secção 4 é apresentado o caso de estudo realizado com o intuito de testar o *SmartClean* e validar a arquitectura proposta. Por fim, na Secção 5, é apresentada a conclusão.

2. PROBLEMAS DE QUALIDADE DOS DADOS AO NÍVEL DO ATRIBUTO

Nesta secção apresentam-se os PQD que podem ocorrer ao nível do atributo, com base na

taxionomia proposta em [7]. A ferramenta *SmartClean* apresentada neste artigo suporta a detecção e correcção de todos estes problemas.

- Valor em falta – Ausência de valor num atributo de preenchimento obrigatório.
- Violação de sintaxe – O valor não respeita a sintaxe estabelecida para o atributo.
- Erro ortográfico – O valor contém um erro ortográfico acidental ou não.
- Violação de domínio – O valor não faz parte do conjunto de valores válidos do atributo. Num atributo do tipo enumerado textual, uma violação de domínio pode corresponder a um:
  - Valor sobrecarregado – O valor do atributo contém informação além do pretendido.
  - Valor incompleto – O valor do atributo contém informação aquém do pretendido.

A existência destes problemas afecta negativamente os resultados produzidos por qualquer ferramenta de exploração de dados.

3. ARQUITECTURA DO *SMARTCLEAN*

Na Figura 1 apresenta-se a arquitectura da ferramenta *SmartClean*, desenvolvida para a detecção e correcção dos diversos tipos de PQD.

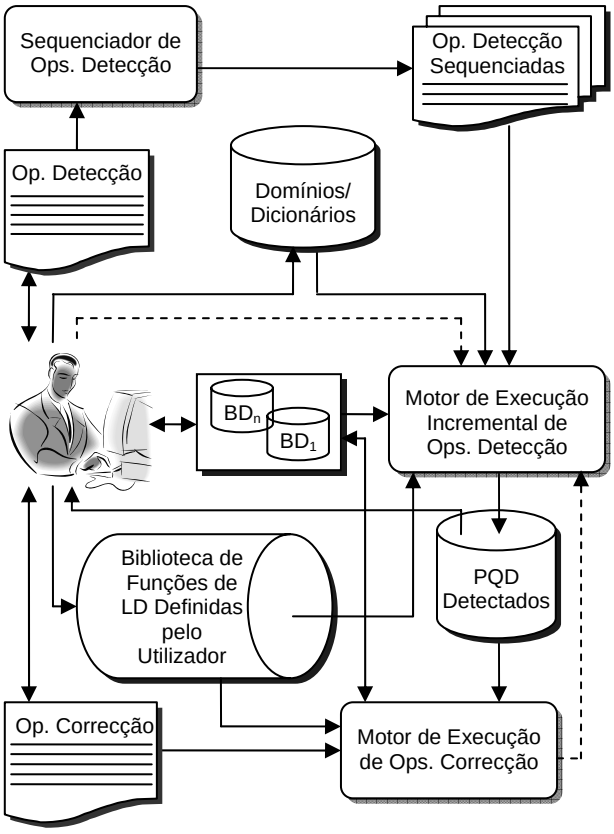

Fig. 1. Arquitectura do *SmartClean*

Uma descrição dos diversos componentes que compõem a arquitectura e da forma como estes interactuam entre si é, a seguir, fornecida.

O *utilizador* (*i.e.*, quem detém o conhecimento necessário para efectuar limpeza de dados) começa por especificar as OD que pretende executar. As operações são especificadas tendo por base uma linguagem declarativa desenvolvida para o efeito, inspirada na linguagem SQL. A seguir, as operações são analisadas a nível sintáctico e caso sejam detectados erros, são reportados ao utilizador para que este os corrija.

Quando não existirem problemas sintácticos, as operações são sujeitas ao *Sequenciador de OD*. Este constitui um dos principais componentes da arquitectura, pelo que será apresentado em detalhe na subsecção seguinte. Por agora, é suficiente dizer que é responsável por estabelecer a ordem pela qual as operações são executadas.

Estando sequenciadas, as operações são submetidas ao *Motor de Execução Incremental de OD*. À semelhança do anterior, pela sua importância na arquitectura, este componente também será alvo de exposição detalhada numa das subsecções seguintes. Como o próprio nome o indica, este componente é responsável pela execução de cada OD. Naturalmente, as operações incidem sobre a Base ou Bases de Dados (BD) alvo de limpeza. Apesar das OD disponíveis de base para execução, o utilizador pode especificar operações que envolvam a invocação de funções definidas por si.

Da execução de cada OD resulta a identificação dos PQD existentes, sendo estes armazenados num repositório. O utilizador acede a este repositório para analisar os PQD identificados (que problemas e em que locais).

As correcções passíveis de serem efectuadas automaticamente aos PQD são efectuadas de acordo com as operações definidas pelo utilizador. As OC também são especificadas tendo por base uma linguagem declarativa desenvolvida para o efeito, inspirada em SQL. Após a especificação, estas são analisadas a nível de sintaxe e caso sejam detectados erros, são reportadas ao utilizador para que os corrija.

Assim que não existam problemas sintácticos, as operações são submetidas ao *Motor de Execução de OC*. Em virtude da sua importância, este componente também será alvo de atenção especial, materializada numa das subsecções seguintes. Por agora, é suficiente dizer que este permite a execução das OC especificadas.

Na execução de uma operação, acede-se ao repositório onde se encontram os PQD detectados, uma vez que apenas estes são alvo de correcção. Da execução da operação resultam

actualizações à BD, com o objectivo de solucionar os problemas identificados. Este componente suporta, de raiz, a execução de um conjunto de operações que permitem a correcção dos PQD ao nível do valor individual do atributo.

Após a execução das OC é despoletado novamente a execução das OD, nos mesmos pontos onde tinham sido detectados os PQD. Desta forma, reinicia-se uma nova iteração de detecção e, caso sejam detectados novos PQD, também de correcção. Quando não forem detectados qualquer PQD, este processo iterativo de detecção – correcção é dado como concluído.

A execução das OD ou OC pode envolver a invocação de funções implementadas pelo próprio utilizador. A finalidade da biblioteca de funções de limpeza de dados presente na arquitectura é, precisamente, a de permitir o armazenamento destas funções. Dependendo do domínio, podem ser necessárias funções específicas que suportem a detecção e correcção de certos PQD. Uma vez incluída na biblioteca, uma função pode ser usada na situação actual de limpeza de dados, mas também em qualquer outra situação futura.

A OD de erro ortográfico envolve o acesso a um dicionário de termos, armazenado sob a forma de tabela. A execução da OD de violação de domínio pode envolver o acesso ao domínio de valores válidos do atributo, também armazenado sob a forma de tabela. A finalidade do repositório presente arquitectura intitulado de *Domínios*/ *Dicionários* é a de suportar o seu armazenamento.

Além dos fluxos de dados, no diagrama da arquitectura apresentado na Figura 1 encontram-se representados fluxos de controlo (a tracejado). Um dos fluxos tem início no *Motor de Execução das OC* e fim no *Motor de Execução Incremental das OD*. Este fluxo representa o desencadear automático da execução das OD após a execução das OC. O outro fluxo tem origem no *Utilizador* e termina no *Motor de Execução Incremental das OD*. Este representa a capacidade que o utilizador tem de despoletar a execução incremental das OD. Uma vez que o utilizador pode efectuar correcções directamente na própria BD, este também tem a possibilidade de despoletar a re-execução das OD. Assim, o utilizador certifica-se que os problemas foram, de facto, solucionados.

Nas três subsecções seguintes descrevem-se, com o necessário detalhe que merecem, os três principais componentes da arquitectura.

3.1 Sequenciador de Operações de Detecção

Neste trabalho defende-se a detecção e, de imediato, a correcção dos PQD, segundo uma determinada sequência. A existência de um PQD pode impedir a detecção de um ou mais problemas (*e.g.*: a existência de uma violação de sintaxe pode impedir a detecção de uma violação de domínio). É necessário solucionar o problema detectado para que seja possível identificar outros problemas. Por outro lado, a existência de um problema pode resultar na detecção de um ou mais problemas (*e.g.*: a existência de um erro ortográfico pode conduzir à detecção de uma violação de domínio). Assim, um PQD pode reflectir-se, em cascata, num conjunto de outros problemas. Ao corrigir-se o primeiro, todos os outros supostos problemas são também solucionados. Face à interdependência entre os PQD, só se deve avançar para a detecção de um problema quando todos os problemas de que este depende, tiverem sido detectados e corrigidos. Caso não se adopte esta abordagem gradual, podem não ser detectados certos problemas ou, de forma redundante, serem identificados outros. O *Sequenciador de OD* é responsável por colocar as operações pela sequência de execução que se preconiza, de acordo com o que se apresenta na Figura 2.

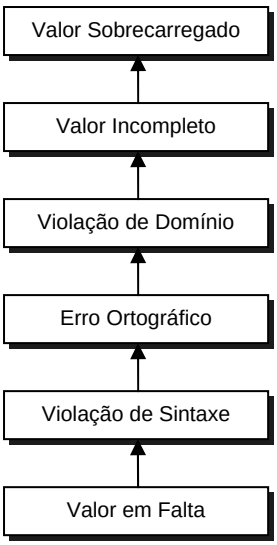

Fig. 2. Sequência de Execução das OD

A primeira operação a ser executada é a de detecção de valores em falta. A existência de um valor em falta num atributo de preenchimento obrigatório, obriga a que este problema seja identificado e imediatamente solucionado. Após ter sido fornecido o valor, é possível executar as restantes operações da sequência de OD. Desta forma, verifica-se se o valor fornecido possui qualquer outro problema de qualidade.

A OD de violação de sintaxe é a seguinte da sequência. Uma violação de sintaxe pode causar a detecção redundante de um erro ortográfico ou de uma violação de domínio.

A próxima operação a ser executada envolve a detecção de erros ortográficos. A existência de um erro ortográfico origina a detecção redundante de uma violação de domínio.

A OD de violação de domínio é a seguinte da sequência. Estando solucionados os PQD que podem resultar na detecção de violações de domínio, que não são mais do que um mero reflexo desses problemas, estão reunidas as condições para a execução desta operação.

Além da OD de violação de domínio, também podem ser especificadas pelo utilizador OD de valor incompleto ou de valor sobrecarregado. Os valores incompletos ou sobrecarregados representam casos específicos de violação de domínio. Estas operações ainda que normalmente acompanhem a OD de violação de domínio, podem ser efectuadas mesmo que esta não exista. Quando existe a OD de violação de domínio, qualquer uma das duas operações (de valor sobrecarregado ou incompleto) tira partido dos resultados já produzidos. Apenas constitui um possível valor incompleto ou sobrecarregado, os que foram identificados como violações ao domínio. A não violação do domínio do atributo garante, desde logo, que o valor não é incompleto nem está sobrecarregado. Quando o valor não viola o domínio, nem sequer se justifica a execução destas OD. Por este motivo, as OD de valor incompleto e valor sobrecarregado surgem na sequência após a de violação de domínio.

A operação seguinte da sequência é a de detecção de valor incompleto. A ordem entre esta operação e a OD de valor sobrecarregado é irrelevante. No caso, optou-se por colocar em primeiro a de detecção de valor incompleto, ficando como última operação da sequência a detecção de valor sobrecarregado. A execução sequencial destas duas operações é importante, uma vez que os valores identificados como estando incompletos, já não são verificados pela OD de valor sobrecarregado. Um valor não pode estar incompleto e sobrecarregado.

3.2 Motor de Execução Incremental de Operações de Detecção

A execução das OD obedece às sequências de dependências estabelecidas. As OD referentes a cada atributo originam uma sequência de execução. A existência destas sequências resulta da independência de execução das operações que as compõem. Cada sequência pode ser composta por uma ou várias OD. A execução de cada sequência de OD é efectuada em paralelo.

Neste trabalho defende-se que a detecção de um PQD deve ser seguida da sua imediata correcção. Assim, a detecção de um determinado PQD faz com que a execução da sequência de OD seja interrompida nesse ponto. As restantes operações da sequência de detecção que envolvam esse valor ficam com a execução pendente até que o problema seja solucionado. No entanto, todas as outras OD da sequência cuja execução não dependa da correcção do PQD são executadas. Quando todas as OD tiverem sido executadas é que o utilizador toma conhecimento dos problemas entretanto identificados. Assim, minimiza-se o número de intervenções que o utilizador necessita de efectuar durante o processo de limpeza de dados. Após a correcção dos PQD, a execução da sequência de OD reinicia-se, nos mesmos locais onde tinha parado anteriormente, *i.e*., onde tinham sido detectados os problemas. As operações que tinham sido responsáveis pela detecção dos problemas voltam a ser executadas, com o intuito de garantir que estes foram efectivamente solucionados. As OD da sequência que tinham ficado pendentes vão agora ser executadas. Os valores já anteriormente analisados e nos quais não foram detectados PQD ficam excluídos do âmbito de execução das OD. As correcções efectuadas para solucionar os PQD detectados não são susceptíveis de causarem outros problemas nesses valores.

No caso de serem detectados outros PQD, o procedimento descrito volta a desenrolar-se. Até todos os problemas estarem solucionados, podem ser necessárias múltiplas iterações detecção – correcção. Em cada iteração, os PQD existentes vão sendo incrementalmente detectados e, de seguida, corrigidos, tendo como ponto de reinício da detecção os locais onde esta tinha sido interrompida na iteração anterior. Este processo de execução das OD que compõem a sequência encontra-se representado na Figura 3.

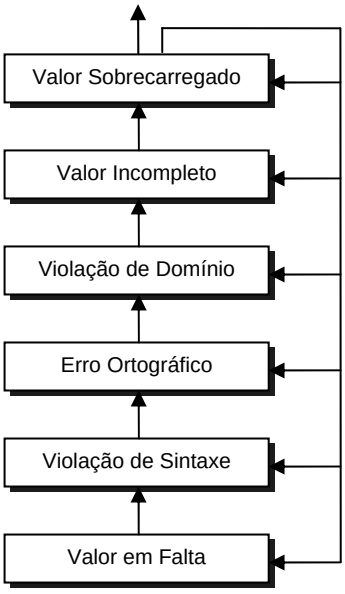

Fig. 3. Processo de Execução da Sequência de OD

A correcção de um PQD pode, de forma involuntária, resultar na introdução de outro problema que se encontre a montante na sequência de detecção. Como forma de garantir que tal não ocorreu, é ainda efectuada uma execução suplementar da sequência de OD, mas só sobre os valores que foram alvo de correcção.

Para melhor ilustrar todo o processo, considere-se o exemplo seguinte que se inicia com a apresentação da relação da Figura 4.

ID	$atrib_2$	CodigoPostal	…	$atrib_n$
1	xxx	4000-123	…	xxx
2	xxx	*null*	…	xxx
3	xxx	4415-206	…	xxx
4	xxx	4415	…	xxx
5	xxx	4445-235	…	xxx
6	xxx	1000-111	…	xxx
7	xxx	3770-255	…	xxx

Fig. 4. Relação para Exemplificação do Processo de Execução Incremental das OD

Suponha-se que o utilizador especificou três OD sobre o atributo *CodigoPostal*: valor em falta; violação de sintaxe; e, violação de domínio. A primeira operação a ser executada é a de detecção de valor em falta. Da execução desta operação resulta a identificação de um valor em falta no tuplo identificado com o *ID* 2. Como consequência, as restantes operações da sequência não são executadas neste valor até o PQD ser solucionado. Nos restantes valores do atributo não é detectado qualquer valor em falta. Nestes, a sequência de execução prossegue.

A operação seguinte a ser executada é a de detecção de violação de sintaxe. Esta operação considera todos os valores do atributo, à excepção do que se encontra no tuplo com o *ID* 2. Da execução desta operação resulta a detecção de uma violação de sintaxe no tuplo com o *ID* 4. A execução das restantes operações fica pendente para este valor até que o problema seja solucionado. Nos restantes valores do atributo não é detectada qualquer outra violação de sintaxe. Nestes valores passa-se à execução da OD seguinte, *i.e.*, detecção de violação de domínio. Esta operação considera todos os valores do atributo, à excepção dos que foram identificados como estando em falta (tuplo com o *ID* 2) ou violando a sintaxe (tuplo com *ID* 4). Da execução da operação, resulta a detecção de uma violação de domínio no valor do atributo do tuplo com o *ID* 6 (*i.e.*, 1000-111). Uma vez que esta é a última operação da sequência, conclui-se que todos os restantes valores (que constam dos tuplos com *ID* impar) não se encontram afectados pelos PQD em causa. Na próxima iteração das OD, estes valores não são considerados na execução destas, uma vez que já se confirmou que não estão afectados pelos problemas em questão. Ao utilizador são, então, reportados os problemas encontrados. Suponha-se que o utilizador procede à correcção (automática ou manual) de cada um destes problemas.

O reinício da execução das OD que compõem a sequência ocorre com a operação de valor em falta. A execução desta operação considera unicamente o valor do atributo *CodigoPostal* no tuplo com o *ID* igual a 2. Suponha-se que o problema foi de facto solucionado. A execução das operações da sequência prossegue com a detecção das violações de sintaxe. Esta operação considera, não só o valor do atributo no qual anteriormente foi identificada a violação de sintaxe (*i.e.*, tuplo com o *ID* 4), mas também o valor do atributo que já não está em falta (*i.e.*, tuplo com o *ID* 2). Suponha-se que ambos os valores não constituem violações de sintaxe. Por fim, é executada a OD de violação de domínio. Esta operação considera não só o valor do atributo no qual anteriormente foi detectada a violação de domínio (*i.e.*, tuplo com o *ID* 6), mas também os valores do atributo que não constituem violação de sintaxe (*i.e.*, tuplos com os *ID* 2 e 4). Assim, como não foram detectados problemas, é efectuada uma execução suplementar da sequência de OD para garantir que as correcções efectuadas não possam ter, inadvertidamente, introduzido outros problemas a montante. O âmbito de execução encontra-se restrito aos tuplos com ID par. Não sendo detectado qualquer PQD, este processo de detecção–correcção finda.

3.3 Motor de Execução de Operações de Correcção

Ao contrário do que acontece nas OD em que a sua execução é feita de acordo com uma sequência predefinida, tal não acontece nas OC. Confrontado com os PQD resultantes de uma iteração de execução das sequências de OD, o utilizador especifica um conjunto de OC que visam a sua resolução. A ordem pela qual estas operações são executadas é irrelevante, uma vez que não há dependências de execução. Assim, a sua execução é efectuada em paralelo. Uma excepção ocorre quando existe mais do que uma OC para o mesmo tipo de problema (*e.g.*: duas operações de correcção para o mesmo PQD de violação de sintaxe). Nestes casos, pode fazer sentido que as operações sejam executadas segundo uma ordem. Assim, quando o utilizador define mais que uma OC para o mesmo PQD, a sua execução respeita a ordem de especificação.

4. CASO DE ESTUDO

O caso de estudo eleito para testar o *SmartClean* e, consequentemente, demonstrar a validade da arquitectura de limpeza de dados subjacente é do domínio das dádivas de sangue.

A BD em questão encontra-se implementada em *MySQL*, sendo composta por 12 tabelas: 3 principais (*i.e.*, colheitas; dadores; e, análises) e 9 auxiliares (*e.g.*, sexo; profissões; estado civil). A tabela de maior dimensão (*i.e.*, colheitas) possui 246208 tuplos, enquanto que a de menor dimensão (*i.e.*, sexo) possui apenas 2 tuplos.

No conjunto das 12 tabelas foram realizadas 62 OD, cobrindo os 6 tipos diferentes de PQD. A título de exemplo, na Figura 5 apresenta-se a especificação de três dessas OD.

```
DETECT MISSING-VALUE
ON CodigoConclusao FROM Conclusoes OF BDD

DETECT SYNTAX-VIOLATION
ON CodigoConclusao FROM Conclusoes OF BDD
WHERE CodigoConclusao NOT LIKE '[A-Z][0-9][0-9][0-9]'

DETECT MISSPELLING-ERRORS
ON DesignacaoConclusao FROM Conclusoes OF BDD
USING DICTIONARY DicPortugues
USING METRIC Jaro-Winkler
```

Fig. 5. Exemplificação das OD

Em 13 das 62 OD, a execução culminou na identificação de PQD. Na Figura 6 apresentam-se os resultados da execução de uma OD de erros ortográficos.

Erro Ortográfico	Valores Similares	Grau Semelhança
psicosocial	psicossocial	0.9888889
psicosocial	psico-social	0.98611116
patologai	patologia	0.98518515
patologai	catalogai	0.80423284
psqiquiátrica	psiquiátrica	0.95726496
infeciosa	infecciosa	0.98333335
inotoxicação	intoxicação	0.95353544

Fig. 6. Resultados de OD de Erros Ortográficos

Para solucionar os PQD foram especificadas OC. A título de exemplo, na Figura 7 apresenta-se a especificação de uma dessas OC.

```
CORRECT SYNTAX-VIOLATION
ON CodProfissao FROM Profissoes OF BDD
BY TRANSFORMING (\d\d)-(\d\d)-(\d\d) INTO \1-\2.\3
BY TRANSFORMING (\d\d)-(\d\d)-(\d\d)(\d\d) INTO \1-\2.\4
```

Fig. 7. Exemplificação de OC

Após a realização das correcções, o reinício de execução das OD, no geral, já não redundou na identificação de qualquer PQD. Apenas em duas situações tal não sucedeu. Nessas situações, as OC especificadas não foram capaz de solucionar todos os problemas que haviam sido identificados. Numa das situações, o único problema que ainda subsistia acabou por ser solucionado manualmente. Na outra situação, a OC inicialmente especificada foi alvo de refinamento, de modo a passar a manipular adequadamente os PQD que ainda persistiam.

5. CONCLUSÃO

O estudo de caso efectuado permitiu confirmar que o *SmartClean* é útil e válido na limpeza de dados. Contrariamente ao que sucede em todas as outras ferramentas comerciais e académicas, o *SmartClean* não obriga a que o utilizador especifique a sequência de execução das operações de limpeza. Esta característica constitui o principal contributo da ferramenta.

Actualmente, o *SmartClean* não possui uma interface gráfica. A especificação das operações é efectuada em ficheiros de texto, sendo os resultados colocados em tabelas de uma BD. A inexistência da interface gráfica reflecte-se negativamente na usabilidade da ferramenta. Este será um dos aspectos objecto de trabalho futuro.

Referências

[1] Trillium Software (2009), *TS Quality Version 10.0 - Enterprise Services Edition*. Disponível em http://www.trilliumsoftware.com/site/content/resources/library/pdf_detail.asp?id=147&pdfRecorded=1, no dia 03/02/2009, às 22:15.

[2] DataFlux (2009), *Accelerate to Compliance, Data Governance and MDM*. Disponível em http://www.dataflux.com/Resources/filestream.asp?rid=146, no dia 03/02/2009, às 22:40.

[3] ETI (2009), *ETI Data Cleanser: Process Driven Data Cleansing and Matching for the Transparent Enterprise*. Disponível em http://www.eti.com/data_sheets/ds_ETI_DataCleanser.pdf, no dia 03/02/2009, às 23:20.

[4] Galhardas, H.; Florescu, D.; Shasha, D.; Simon, E. e Saita, C. (2001), *Declarative Data Cleaning: Language, Model and Algorithms*, in *Proceedings of the 27th Very Large Databases Conference*. p. 371-380.

[5] Lee, M.; Ling, T. e Low, W. (2000), *IntelliClean: A Knowledge-Based Intelligent Data Cleaner*, in Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 290-294.

[6] Raman, V. e Hellerstein, J. (2001), *Potter's Wheel: An Interactive Data Cleaning System*, in Proceedings of the 27th Very Large Databases Conference. p. 381-390.

[7] Oliveira, P.; Rodrigues, F. e Henriques, P. (2005), A *Formal Definition of Data Quality Problems*, in Proceedings of the 10th International Conference on Information Quality, MIT. p. 13-26.