Controlo da Evolução de Sistemas Legados

Miguel Goulão
Fernando Brito e Abreu
INESC

António Silva Monteiro
Alberto Bigotte de Almeida
DAMAG

Resumo

No âmbito de uma iniciativa de reorganização do processo de desenvolvimento de software na Marinha Portuguesa, desenvolveu-se um sistema de rastreio de acções de evolução implementado sobre a intranet da organização, com vista a melhor controlar a evolução de sistemas legados. A arquitectura desse sistema e o modo como os dados relativos à evolução de um conjunto de sistemas legados são utilizados para melhor compreender o processo serão abordados. Uma análise detalhada sobre a definição de um modelo de estimação de esforço com base na complexidade envolvida na execução de acções de manutenção será aqui apresentada. A evolução das soluções de desenho utilizadas há uma década na organização, face às adoptadas correntemente será também objecto de estudo.

Palavras Chave: métricas de software, rastreio de acções de evolução, modelos de estimação

1. Introdução

A comunidade científica reconhece desde há muito a necessidade de aplicar métodos quantitativos ao controlo da actividade de evolução de software. São conhecidos os efeitos nefastos de abordagens baseadas apenas na experiência dos gestores: dificuldades na previsão do esforço, custo e prazos adequados para a realização de determinadas acções de evolução. Consequências tais como a entrega tardia do software aos utilizadores finais, degradação da sua qualidade e necessidade de reforçar a equipa de desenvolvimento para controlar desvios face ao previsto inicialmente, são comuns. Todas estas situações acarretam prejuizos importantes para as organizações, o que as conduz à motivação de desenvolver métodos de trabalho que minimizem estes problemas.

Um estudo recente acerca da avaliação da qualidade do software na Europa[Punter98], quer ao nível do processo de desenvolvimento, quer ao nível do produto em si, revelava que uma percentagem elevada (superior a 75%) das organizações produtoras de software estavam envolvidas em iniciativas de avaliação da qualidade do software e do seu processo de desenvolvimento, procurando encontrar formas de melhorar este último. A amostra foi constituida com base em dois inquéritos realizados em 1997 aos quais responderam um total de 55 organizações, cerca de 1/6 das organizações contactadas. Um dos dados retirados deste estudo era a necessidade que as organizações sentiam em ganhar confiança no software e seu processo de desenvolvimento. Enquanto consumidoras, para poderem escolher as melhores alternativas entre produtos concorrentes. Enquanto produtoras, para melhor gerirem os riscos inerentes aos projectos e para ganhar aceitação entre potenciais clientes.

Ao nível de organizações estatais portuguesas, são poucos os casos conhecidos em que se estão a desenvolver projectos de melhoria do processo de software suportados por iniciativas de recolha de dados quantitativos sobre o mesmo.

A Direcção de Análise de Métodos de Apoio à Gestão (DAMAG), organização que é responsável pelo desenvolvimento e manutenção da infra-estrutura dos sistemas de informação na Marinha Portuguesa, está envolvida numa iniciativa de evolução do seu processo de software. Este projecto é realizado com a colaboração do Grupo de Engenharia de Software do Instituto de Engenharia de Sistemas e Computadores (INESC).

Nesta fase de estudo e desenvolvimento, foram seleccionados 4 sistemas de informação de pessoal desenvolvidos internamente pela DAMAG utilizando COBOL e SQL-DS num sistema proprietário, dos quais se apresentam aqui alguns dados, na Tabela 1.

Sistema de Informação	Nº de módulos	Ano de entrada em exploração	Fase actual
Informação do Pessoal	650	1989	Exploração
Formação do Pessoal	880	1997	Exploração
Vencimentos	927	1999 (previsto)	Testes
Orçamento	612	1999 (previsto)	Testes

Tabela 1 – Sistemas analisados

A abordagem a esta evolução do processo assenta, em primeiro lugar, na criação de uma cultura diferente na organização. É dado particular relevo à necessidade de registar, de forma fidedigna e suficientemente detalhada, as acções de evolução dos sistemas e o respectivo esforço envolvido na sua realização. É neste contexto que surge o SofTrack (Software Defect Report and System Tracking), um sistema desenvolvido neste projecto a fim de suportar o rastreio de acções de evolução. Um requisito para o SofTrack era a sua disponibilização através da Intranet da Marinha Portuguesa, dada a dispersão

geográfica da organização. Pretendia-se torná-lo facilmente acessível a todos os intervenientes no processo, desde a equipa de desenvolvimento até aos utilizadores finais.

O SofTrack é também usado no acompanhamento da evolução do software, através da análise do código fonte ao longo de diferentes versões. Esta análise inclui a extracção de métricas de complexidade do software e a produção automatizada de documentação técnica actualizada, em formato HTML.

A análise das métricas de complexidade extraídas permite uma comparação entre as práticas de programação utilizadas em diferentes sistemas, apesar das tecnologias de base serem as mesmas. A combinação entre a variação das métricas de complexidade e o esforço registado para as acções que as originam é a base para a proposta de um modelo de estimação do esforço com base na complexidade das alterações.

Os relatórios produzidos com base nos dados recolhidos com o SofTrack constituem assim uma fonte de informação valiosa, no planeamento de acções de evolução. Fornecem informação quantitativa que visa reduzir a subjectividade com que o planeamento seria tradicionalmente efectuado, ajudam a manter o processo controlado e contribuem para a detecção de oportunidades de evolução no processo de software.
Este documento está organizado do seguinte modo:

Na secção 2, será descrita brevemente a arquitectura do Softrack. A terceira secção é dedicada à construção, validação e análise crítica de um modelo de estimação de esforço com base na complexidade das acções de evolução. São apresentadas limitações do modelo, bem como acções possíveis para as erradicar. Segue-se um estudo comparativo dos sistemas analisados, na quarta secção, com base nas métricas de complexidade recolhidas, em que se identificam mudanças na forma como os sistemas foram desenvolvidos, apesar de as tecnologias usadas serem semelhantes.

Finalmente, apresentam-se algumas conclusões sobre o trabalho efectuado e perspectivam-se as linhas de evolução do projecto.

2. Arquitectura do SofTrack

Todo o processo de recolha de dados assenta num sistema de rastreio de acções de evolução desenvolvido no âmbito deste projecto, o SofTrack. Este sistema, descrito em [Monteiro99], pretende dar uma resposta à necessidade de implementar um mecanismo de rastreio de acções de evolução num ambiente geograficamente distribuido.

O SofTrack está implementado sobre a Intranet da Marinha Portuguesa. Assenta num servidor Web com o sistema operativo Windows NT 4.0 e o Internet Information Server 4.0 (IIS). As potencialidades do IIS são extendidas através de um componente que permite o suporte a *servlets*[1] em Java (ServeltExec for Windows).

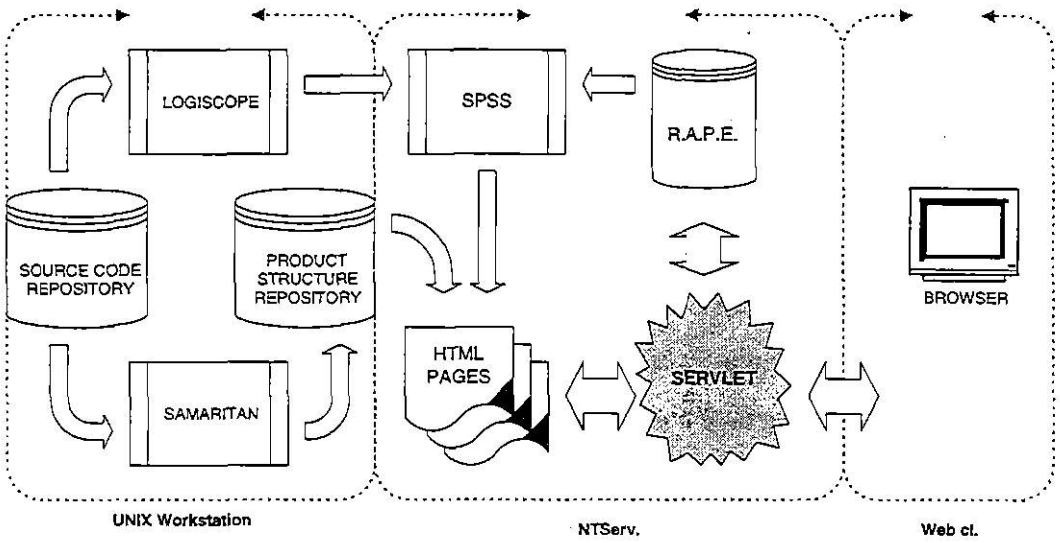


Figura 1 - Arquitectura do SofTrack

Os ficheiros de código-fonte, residentes no SCR (*Source Code Repository*), são submetidos às ferramentas Logiscope [Verilog93] e Samaritan instaladas numa *Workstation Unix.*

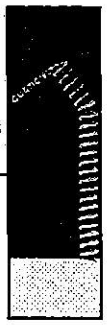
[1] *Servlet* – Aplicação independente da plataforma, escrita de acordo com a *Java Servlet API* que permite extender um servidor HTTP, implementando serviços.

O Logiscope é uma ferramenta que permite a extracção da informação acerca da estrutura global da aplicação (grafos de chamada), da estrutura lógica dos seus componentes (grafos de controlo), bem como a obtenção de métricas de complexidade do produto, incluindo a complexidade textual [Halstead77] e estrutural [McCabe76]. O Samaritan é uma ferramenta que permite a geração de documentação actualizada acerca dos sistemas em análise, com base nos ficheiros de código-fonte. A documentação gerada é organizada no PSR (*Product Structure Repository*) e apresentada em páginas HTML, permitindo uma fácil navegação entre elas. A informação disponibilisada contempla a estrutura e implementação do sistema, incluindo informação acerca das operações de acesso às base de dados pelos programas e interfaces.

Os dados associados ao processo são recolhidos através do RAPE (Registo e Análise de Pedidos de Evolução), o sub-sistema que suporta o registo dos pedidos de evolução dos sistemas de informação [Goulão98]. O SofTrack permite ao utilizador seleccionar os pedidos, verificar o seu estado de atendimento, e submeter novos pedidos de evolução. Adicionalmente, para a equipa de manutenção, é dado acesso à informação associada à evolução e seguimento do pedido, até ao seu encerramento.

A ferramenta SPSS é usada na análise estatística das métricas do processo e de produto. É disponibilizada informação de controlo dos projectos, através de relatórios periódicos gerados automaticamente em páginas HTML.

A arquitectura cliente-servidor do SofTrack é implementada recorrendo a um *servlet*, responsável pela gestão da interacção entre o utilizador e o SofTrack. Este é responsável pela implementação da política de segurança do SofTrack, associando previlégios de utilização aos grupos de páginas HTML a que o utilizador deverá ter acesso.



3. Modelo de estimação de esforço

3.1. Introdução

Intuitivamente, é expectável que a complexidade de uma determinada acção de evolução se reflicta no esforço necessário para a executar. Nesta secção será descrita a abordagem metodológica seguida no âmbito deste projecto para a verificação desta relação.

A partir dos dados recolhidos através do RAPE e das métricas de produto, pretende-se construir um modelo de estimação do esforço com base na variação dos valores das métricas de complexidade para as respectivas acções de evolução. Assim, o modelo a construir por regressão múltipla terá como variável dependente o esforço efectivamente despendido na acção de evolução e como estimadores os factores a extrair da variação das métricas de complexidade do software.

Para o presente estudo, assumem particular relevância os dados referentes ao esforço efectivo despendido em determinada acção de manutenção e a identificação dos módulos alterados, bem como a versão do software correspondente. Todos estes dados são armazenados no RAPE.

3.2. Construção do Modelo

Se a escolha da variável dependente é trivial (afinal, pretende-se estimar precisamente o esforço), já a selecção de um conjunto de estimadores adequado envolve uma reflexão mais cuidada.

As métricas de complexidade recolhidas reflectem diversos aspectos da complexidade do software. No entanto, verifica-se que esses aspectos não são completamente independentes entre si, isto é, algumas das métricas de complexidade apresentam uma

correlação significativa e elevada entre si. Por outras palavras, existe redundância na informação que nos é fornecida pelo conjunto de métricas recolhido.

Um segundo problema no conjunto de métricas é o do seu elevado número. Avaliar a complexidade de uma alteração com base em algumas dezenas de indicadores é uma tarefa semelhante à tomada de decisões com base em múltiplos critérios. Tipicamente, de acordo com alguns critérios, um módulo A é mais complexo do que um módulo B, acontecendo precisamente o contrário se considerarmos outro grupo de critérios. É necessário estabelecer uma forma de pesar os diversos critérios em jogo.

Métrica de complexidade	F1	F2	F3
Complexidade do programa	0.992	0.037	-0.028
Aninhamentos	0.990	-0.058	0.009
Chamadas directas	0.989	0.050	-0.054
Nós de Saida	0.988	-0.055	-0.013
Nós de Entrada	0.987	-0.053	-0.050
Máximo de arestas ligadas a um nó	0.982	0.047	-0.061
Conteúdo inteligente	0.976	0.128	-0.013
Complexidade ciclomática	0.946	0.313	0.018
Nível de programação	0.945	-0.106	-0.064
Dimensão do programa	0.929	0.342	-0.026
Máximo de Nós	0.924	-0.017	0.108
Máximo de Nós Sequenciais	0.919	0.376	0.000
Máximo de Instruções	0.916	0.093	0.056
Nós	0.907	0.409	0.021
Operadores	0.896	0.397	-0.003
Quebras na sequência	0.895	0.439	-0.015

Nível do programa	0.894	0.425	-0.002
Número de arestas	0.890	0.445	0.027
Operandos	0.889	0.448	0.000
Erros esperados	0.762	0.624	-0.009
Volume do Programa	0.742	0.655	-0.001
Esforço	-0.033	0.950	-0.019
Saltos incondicionais	-0.288	0.928	-0.032
Complexidade essencial	0.626	0.714	-0.065
Nós Pendentes	-0.012	-0.050	0.993

Tabela 2 – Componentes derivados por análise de componentes principais

O terceiro problema, que de certo modo deriva do segundo, é o de que diferentes métricas de complexidade utilizam diferentes escalas para os respectivos valores, o que obriga a uma operação de transformação dos mesmos, neste caso uma padronização, de modo a que todos possam ser utilizados na construção de um único modelo.

A fim de dar resposta a estas questões, decidiu-se utilizar uma técnica estatística denominada Análise Factorial em Componentes Principais sobre os valores padronizados das diversas métricas de produto recolhidas. Esta técnica permite reduzir o número de variáveis utilizadas sem uma perda significativa de informação [Reis93]. Os factores derivados a partir desta análise têm como característica fundamental uma baixa correlação entre si, minimizando simultaneamente a redundância dos critérios e o seu número. A **Tabela 2** destaca o modo como as métricas de complexidade se agrupam em 3 factores que serão usados no resto do documento como os critérios de avaliação da complexidade das alterações e referidos abreviadamente por F1 e F2. Foi utilizado o método Varimax [Kaiser58] que permite obter factores de modo a que a contribuição de cada métrica seja elevada para um deles e baixa para os restantes. Pretende-se com isto minimizar o grau de associação entre os factores, o que é conveniente para a utilização que lhes será dada adiante.

Uma análise mais cuidada aos agrupamentos de métricas em cada factor revela uma relativa heterogeneidade no que diz respeito aos aspectos da complexidade que cada métrica contempla. Tal resulta, essencialmente, da elevada correlação entre as métricas e a dimensão do módulo respectivo.

Postas as anteriores considerações, pode-se então apresentar o seguinte modelo, construído por regressão múltipla (uma descrição detalhada sobre a criação e validação destes modelos pode ser encontrada em [Arroja39] e [Bryman92]) e considerando 41 casos para a sua construção:

$Esforço_i = \beta_0 + \beta_1 * F1_i + \beta_2 * F2_i + \varepsilon_i$

Os coeficientes β_j foram estimados pelo método dos mínimos quadrados que nos garante estimadores com todas as propriedades desejadas, isto é, lineares, centrados, eficientes e consistentes (convergentes), de menor variância. β_0 representa o valor do esforço previsto pelo modelo quando os Factores F1 e F2 são nulos (note-se que estes factores estão padronizados, podendo assim assumir valores positivos e negativos). β_1 e β_2 são os coeficientes de estimação para as variáveis independentes. ε é o erro aleatório, também conhecido como termo estocástico, que explica o facto de uma determinada proporção da variação na variável não ser explicada no modelo. A exclusão de F3 do modelo resultou do facto de a sua contribuição para a explicação da variação do esforço ser aproximadamente nula, isto é, $\beta_3 \approx 0$.

Neste caso, o modelo pode ser instanciado com os parâmetros apresentados na Tabela 3.

Coeficiente	Estimativa	Erro Padrão	Estatística t (5%)	Significância
β_0	6.682	0.470	14.221	0.000
β_1	3.104	1.060	2.928	0.006
β_2	8.522	0.587	14.508	0.000

Tabela 3 – Parâmetros do modelo

$Esforço_i = 6.682 + 3.104 * F1_i + 8.522 * F2_i + \varepsilon_i$

Verifica-se que todos os coeficientes apresentados apresentam valores significativos e positivos. Isto reflecte que um aumento de qualquer dos factores de complexidade se traduz por um aumento no esforço estimado. O erro padrão (e) permite o estabelecimento de intervalos de confiança do tipo $]\beta_i - e_i; \beta_i + e_i[$, com um grau de certeza de 95%. O teste de significância estatística dos valores dos coeficientes de regressão individuais, através do cálculo dos valores da estatística t e o teste bicaudal de significância indicam uma baixa probabilidade de os coeficientes serem iguais a 0, na população.

Na Figura 2 o eixo horizontal do gráfico representa a complexidade da alteração e no vertical o respectivo esforço, em homens.hora. As etiquetas junto aos casos contêm o seu código de identificação, no RAPE.

Para saber até que ponto o colectivo das variáveis independentes explicam a variável dependente, pode-se calcular o coeficiente de determinação (R^2). Este modelo apresenta um R^2 de cerca de 85.1%, sendo o seu valor ajustado ($R^2_{Ajustado}$) de 84.3%. O valor ajustado dá uma ideia mais próxima da variação explicada do esforço, por entrar em conta com o número de variáveis independentes (o valor de R^2 tende a ser inflaccionado pelo número de variáveis independentes). Tomando o valor ajustado, temos que apenas 15.7% (100% - 84.3%) da variação do esforço não é explicada pelas variáveis da equação (F1 e F2).

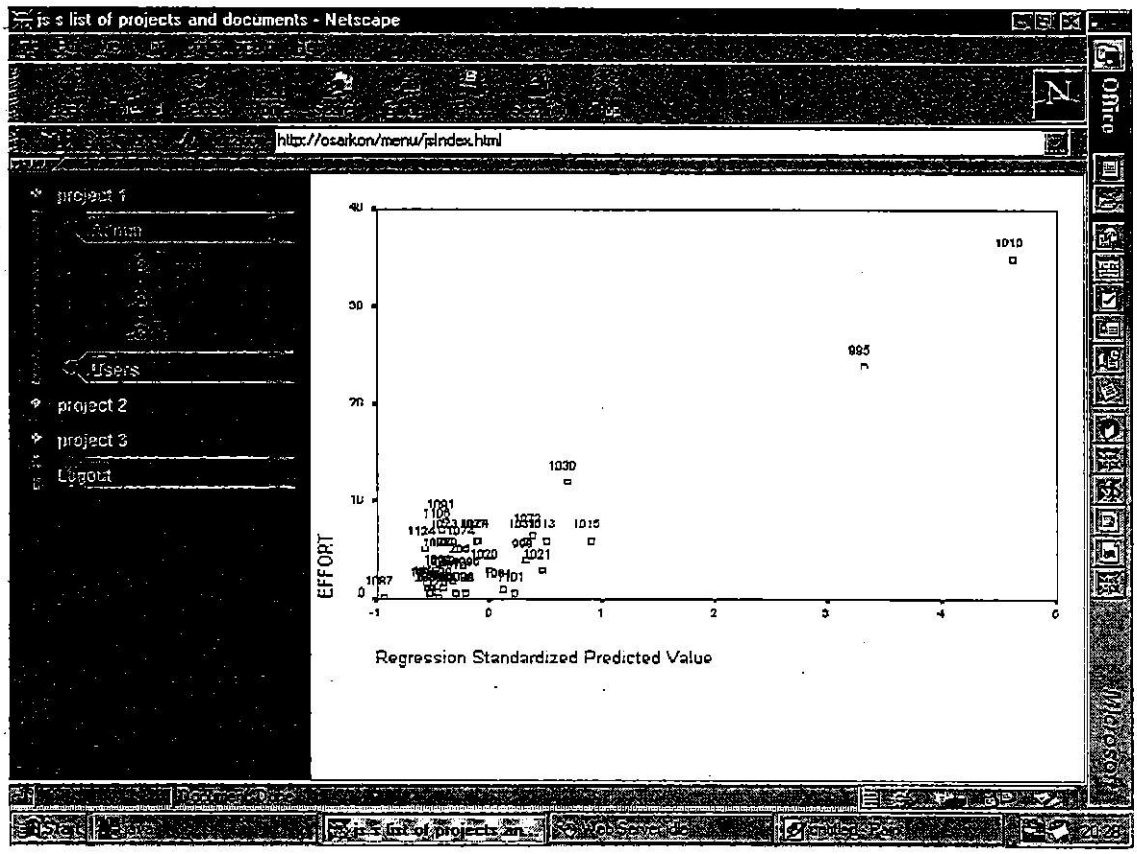


Figura 2 - Esforço previsto vs esforço registado

É extremamente improvável que a correlação múltipla (R), ou seja, a correlação entre o esforço e as variáveis independentes (F1 e F2) em conjunto seja 0, como comprova o teste F. Com um valor de significância de 0.05, o valor do teste F (108.561) apresentado pelo modelo é muito superior ao valor crítico de $F_{(2,38)} = 3.23$. Por outras palavras, está assegurada a significância estatística da equação como um todo.

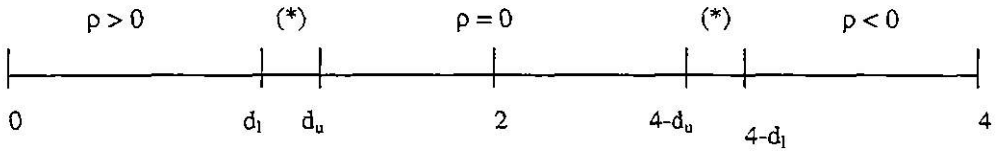
Para demonstrar a inexistência de autocorrelação entre os termos estocásticos recorreu-se ao teste de Durbin-Watson. Tomou-se como hipótese nula a inexistência de autocorrelação entre os termos estocásticos (H_0: $\rho = 0$) e por hipótese alternativa a existência de autocorrelação (H_1: $\rho \neq 0$).

Para 41 casos analisados (n) e 2 variáveis independentes (k), define-se a partir dos valores críticos (d_l – valor crítico inferior; d_u – valor crítico superior) retirados da tabela

da estatística de Durbin-Watson (5% de significância; $d_l = 1.2$; $d_u = 1.4$) o seguinte intervalo de aceitação da hipótese nula:

$]d_u, 4 - d_u] =]1.4, 2.6]$

O valor da estatística de Durbin-Watson (2.079) enquadra-se no intervalo de valores associados à aceitação da hipótese nula (ver Figura 3). Por consequência, é de admitir que a eficiência dos estimadores obtidos pelo método dos mínimos quadrados, dada a inexistência de autocorrelação entre eles.



(*) Teste inconclusivo

Figura 3 - Tomada de decisão no teste de Durbin-Watson

A variância dos termos estocásticos deverá ser constante, ou seja, não dependente das observações ou da variável dependente (homocedasticidade). A validação desta premissa foi, no âmbito deste trabalho, efectuada recorrendo ao teste de Goldfeld-Quandt. Este teste permitiu verificar a inexistência de variáveis causadoras de heterocedasticidade, garantindo assim a premissa da homocedasticidade.

A ausência de colinearidade foi também testada: várias aplicações com dados experimentais têm demonstrado que *conditions numbers* entre 20 e 30 são provavelmente indicadores de problemas de colinearidade [Johnston84]. O modelo proposto apresenta valores máximos de 1.843, não existindo portanto evidências de colinearidade.

3.3. Discussão de Resultados Obtidos

Com base no modelo apresentado, interessa agora verificar o grau de conformidade com os valores reais do esforço face às previsões produzidas. A Figura 4 apresenta o esforço efectivo numa determinada acção de evolução, quando comparado com o respectivo esforço estimado.

Verifica-se que para alguns dos casos, a margem de erro entre o esforço real e o estimado é relativamente grande. Certos factores que pesam no esforço não são reflectidos pelo modelo apresentado, isto é, a quantificação da complexidade das acções de evolução não pode ser considerada completa. Isto porque apenas dispomos de ferramentas para analisar a variação da complexidade do código fonte produzido em COBOL, o que nos permite analisar apenas parte das alterações. Note-se que estas podem envolver modificações ao nível dos acessos da base de dados, por exemplo, através do código SQL embutido no COBOL. Presentemente, ainda não foi desenvolvida uma forma de quantificar o esforço relacionado com esse aspecto da complexidade da alteração. Um problema semelhante ocorre no que diz respeito a modificações nos paineis, que estão também codificados noutra linguagem (ISPF), pelo que essas alterações também não estão a ser consideradas no modelo.

Não menos problemática para a definição de um modelo, é a eficiência das pessoas envolvidas na acção. Ainda que fosse possível uniformizar factores como a experiência dos programadores, o seu conhecimento dos sistemas que estão a manter, a eficácia (qualidade das alterações produzidas) e eficiência (celeridade com que as evoluções são desenvolvidas) dos mesmos varia de pessoa para pessoa. Em [Brooks75] e [Sackman68], são referidas variações de uma ordem de grandeza na produtividade de programadores experientes. O crescimento da amostra pela inclusão de novas acções de manutenção tenderá a diluir um pouco estas discrepâncias, contribuindo assim para a construção de um modelo mais fiável.

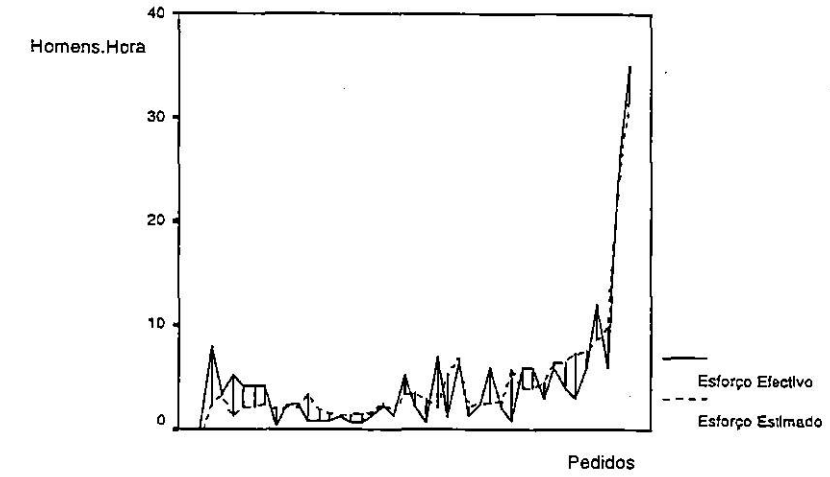


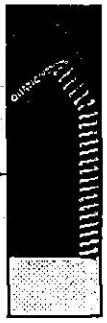
Figura 4 – Diferença entre o esforço efectivo e o esforço estimado

Por estas razões, para a construção deste modelo foram postos de lado alguns casos que apresentavam valores extremos, normalmente por as acções de evolução incidirem bastante sobre os aspectos do código fonte que não puderam ser incluídos no modelo.

4. Evolução no Desenho de Sistemas

Os sistemas analisados têm em comum o facto de terem sido desenvolvidos num ambiente que no essencial é semelhante, em particular no tocante às ferramentas de desenvolvimento utilizadas. A natureza dos sistemas também é comum, uma vez que todos são sistemas de informação sobre o pessoal, embora cada um incida sobre uma vertente particular da informação. Como factores de diferenciação, merece destaque o facto de terem sido desenvolvidos por equipas diferentes e ainda o facto de um deles ter sido desenvolvido quase uma década antes dos restantes. A organização tem uma política de padronização no modo como o desenvolvimento é efectuado, por forma a facilitar a integração de novos elementos, fazendo assim face à inevitável rotatividade do pessoal. Deste modo, é de admitir que o factor cronológico possa assumir particular relevância.

Para avaliar até que ponto as práticas de programação se alteraram com o passar dos anos, fruto da experiência acumulada com o desenvolvimento de anteriores sistemas, pode-se estudar a variação das métricas de complexidade entre um sistema actual e um construido cerca de uma década antes. Para tal, procedeu-se a uma análise de variância entre as métricas de complexidade extraídas de componentes do sistema desenvolvido em 1989 e as mesmas métricas, para os sistemas desenvolvidos desde 1996. Foram retirados desta amostra de módulos aqueles que apresentavam valores extremos para qualquer das métricas recolhidas. Para evitar uma enumeração exaustiva das semelhanças e diferenças encontradas, apresentam-se aqui apenas os dados relativos às maiores variações detectadas.



Métrica	F	Sig.
Saltos Incondicionais	1393.1	.000
Graus de Aninhamento	610.2	.000
Complexidade Essencial	412.4	.000
Quebras na sequência	103.5	.000

Tabela 4 Análise de variância de métricas de produto

Em qualquer dos casos, a análise de variância permite-nos afirmar que existe uma elevada probabilidade de associação entre o sistema considerado e a diferença de médias dos valores destas métricas.

Na Tabela 4, pode-se observar para cada métrica, o valor da estatística F e o nível de significância do teste. Sabendo que F representa a razão entre a variância estimada entre grupos (explicada) e a variância dentro dos próprios grupos (erro), sendo os grupos o sistema implementado há uma década e os mais recentes, isso significa que é pouco provável que as diferenças encontradas sejam devidas ao acaso, para as métricas aqui apresentadas. A amostra continha 2071 módulos.

Uma análise à variação da média destas métricas veio confirmar que as evoluções registadas vão de encontro a uma maior qualidade no desenho e facilidade de manutenção. A ideia dominante é a de que a complexidade dos módulos criados é menor nos sistemas mais recentes. Tal não implica que os sistemas em si sejam menos complexos, mas apenas uma redestribuição da complexidade entre os vários módulos. Uma leitura dos valores médios das métricas para o sistema mais antigo face às médias para os mais recentes permitiu registar como factor positivo o quase desaparecimento dos

saltos incondicionais e a diminuição em 22% da utilização de quebras na sequência (saídas anormais de estruturas do tipo ciclo), sendo que idealmente estes mecanismos deveriam ser evitados tanto quanto possível.

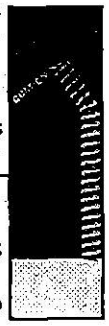
5. Conclusões e trabalho futuro

Apresentou-se nesta comunicação uma experiência que tem vindo a decorrer na Marinha Portuguesa, em que a utilização de métodos quantitativos está a contribuir para uma melhor compreensão do processo de desenvolvimento. As indicações fornecidas pelas métricas de software recolhidas potenciam a adopção de medidas informadas no sentido da melhoria do processo de software.

Descreveu-se brevemente o SofTrack, sistema que tem vindo a ser desenvolvido e utilizado no âmbito deste projecto para dar suporte computacional a esta iniciativa. A própria concepção e desenvolvimento do SofTrack apresentam um desafio em si, dada a diversidade de tecnologias a harmonizar.

A utilização do SofTrack permitiu uma mudança cultural na organização, através da implementação de meios formais para o registo de acções de evolução, um corte importante com a inexistência de um formalismo que permitisse de forma eficiente recuperar informação relativa à evolução dos sistemas, no tocante ao que foi feito, quando, porquê, por quem e com quanto esforço, na evolução dos sistemas. Esta modificação tem como reflexo a possibilidade de cruzar a informação recolhida com o RAPE com as métricas de complexidade de produto, a fim de encontrar a relação entre a complexidade de uma acção de evolução e o esforço nela envolvido.

Essa relação foi explorada no modelo de estimação do esforço aqui apresentado, modelo esse que continuará a ser ajustado à medida que forem sendo registadas novas acções de evolução, de modo a tornar-se cada vez mais fiável. O modelo deverá



posteriormente ser enriquecido com informação acerca da complexidade das alterações efectuadas ao nível da utilização do SQL embebido no Cobol, estando também previsto um estudo sobre a atribuição de pesos diferentes para diferentes programadores, com base na sua produtividade em acções anteriores, para reflectir a discrepância proveniente, por exemplo, da experiência de cada um tanto como programador como em relação ao sistema em causa.

Finalmente, apresentou-se ainda uma utilização das métricas de complexidade do produto para uma melhor compreensão sobre a evolução das soluções de desenho utilizadas num sistema desenvolvido à cerca de uma década, em relação às que estão a ser usadas nos sistemas correntemente em desenvolvimento. Esta análise revelou um aumento do know-how dentro da organização, traduzido por uma utilização mais reduzida de soluções que contribuem para uma má estruturação do código, como sejam os saltos incondicionais.

Bibliografia

[Arroja93] Arroja, P., "Métodos Econométricos", ISEGI, 1993.

[Brooks75] Brooks, F., "The Mythical Man-Month – Essays on Software Engineering", Addison-Wesley Publishing Company, 1975.

[Bryman92] Bryman, A., Cramer. D., "Análise de Dados em Ciências Sociais", Celta Editora, 1992.

[Goulão98] Goulão, M., Monteiro, A., Martins, J., Bigotte de Almeida, A., Brito e Abreu, F., Sousa, P., "A Software Evolution Experiment", ESCOM-ENCRESS98, Roma, 1998.

[Halstead77] Halstead, M., "Elements of Software Science", Elsevier North-Holland, New York 1977.

[Johnston84] Johnston, J., "Econometric Methods", 3ª Edição, McGraw-Hill; Economics Series, New York, 1984.

[Kaiser58] Kaiser, H.F., "The Varimax Criterion for Analytic Rotation in Factor Analysis", Pshycometrica, 1958.

[McCabe76] McCabe, T., "A Complexity Measure", IEEE Transactions on Software Engineering, Vol. 2, N°4 pp 308-320, 1976.

[Monteiro99] Monteiro, A., Goulão, M., Abreu, F., Almeida, A., Sousa, P., "Software Defect Report and Tracking System in the Internet: Controlling the Evolution of Legacy Systems", submetido à 6th European Conference on Software Quality, Viena, 1999.

[Punter98] Punter, T., Lami, G., "Factors of Software Cost Evaluation – Results of Two European Surveys", ESCOM' 98, Roma, 1998.

[Reis93] Reis, Elisabeth, "Análise Factorial das Componentes Principais: Um Método de Reduzir Sem Perder Informação", Giesta ISCTE, 2ª Ed., 1993.

[Sackman68] Sackman, H., Erikson, J., Grant, E., "Exploratory Studies Comparing Online and Offline Programming Performance", CACM, 11, 1, 1968.

[Verilog93] Verilog, "Logiscope Viewer – Basic Concepts", Technical Documents Relating to the Logiscope Tool. Verilog SA., 150 rue Nicolas Vauquelin, BP 1310, 31106, TOULOUSE cedex, France, 1993.