

Avaliação de Sistemas Legados

Cristiane Soares Ramos, Káthia Marçal De Oliveira, Nicolas Anquetil
Universidade Católica de Brasília
{kathia, anquetil}@ucb.br

Abstract. *The process of the legacy system maintenance is inevitable. One way for organizations to decrease their costs has been to outsource maintenance to specialized companies. The challenge for these contracted companies is to rapidly evaluate the legacy systems they will maintain, in order to better price their services. This paper presents a model of metrics to evaluate the complexity of legacy systems. This model was applied on several real COBOL systems.*

Resumo. *A manutenção de sistemas legados é algo inevitável. Uma das formas que as organizações têm encontrado para reduzir seus custos tem sido a contratação de empresas especializadas em manutenção de software. Um grande desafio para as empresas contratadas é avaliar e conhecer os sistemas que elas irão manter em um curto espaço de tempo de forma a melhor definir seus contratos. Desta forma, é muito importante que elas adquiram rapidamente o máximo de informações possível sobre os sistemas legados. Esse trabalho apresenta um modelo de métricas para avaliar a complexidade de sistemas legados. O modelo proposto foi aplicado em um conjunto de sistemas reais.*

1 Introdução

A aceitação de um projeto por uma organização mantenedora envolve alguns riscos, que têm recebido pouca atenção [POLO, PIATTINI e RUIZ, 1998]. A falta de conhecimento prévio sobre o sistema a ser mantido faz com que as organizações mantenedoras podem assumir compromissos difíceis de serem cumpridos, e que comprometerão a lucratividade do serviço e/ou a confiança do cliente. Ainda assim, a maioria dos estudos, na área de engenharia de software, tem se preocupado com as vantagens e desvantagens da terceirização sob o ponto de vista da empresa contratante [POLO, PIATTINI e RUIZ, 1998] e não da contratada (empresa mantenedora).

Por exemplo, não existe proposta destinada à avaliação de sistemas legados antes de iniciar uma manutenção de forma auxiliar as organizações mantenedoras a melhor definir seus contratos de manutenção. A própria norma ISO9126 (2001) define características e métricas para avaliar manutenção com dados coletados durante a manutenção como uma forma de monitorar a evolução do sistema.

Neste artigo apresentamos um modelo de métricas, que foi definido utilizando a abordagem *Goal-Question-Metric* [BASILI e ROMBACH, 1998] para apoiar o melhor entendimento de um sistema legado no que se refere a seu código e sua documentação, de forma a fornecer informações para apoiar a definição de contratos de manutenção.

Nas próximas seções são introduzidos brevemente conceitos e características referentes à manutenção (seção 2) e a métricas de software (seção 3). Na seção 4 será apresentado o modelo de métricas para avaliação de sistemas legados através da aplicação da abordagem GQM. A seção 5 mostra os resultados obtidos através da aplicação prática dessas métricas em sistemas em Cobol. Finalmente, na seção 6 são apresentadas as conclusões e lições aprendidas com a realização deste trabalho.

2 Manutenção de software

A manutenção é definida como a totalidade de atividades requeridas para prover suporte de custo efetivo para um sistema de software. As atividades são realizadas tanto durante o estágio de pré-entrega quanto no estágio de pós-entrega [PIGOSKI, 1996]. A manutenção é uma atividade inevitável e pode vir a ser a mais longa fase no ciclo de vida do sistema, isso pode ser justificado através das leis de evolução do software [LEHMAN, PERRY e RAMIL, 1998] que mostram que os sistemas devem ser continuamente adaptados senão eles se tornarão progressivamente insatisfatórios; que o sistema aumenta sua complexidade conforme ele evolui e que o conteúdo funcional de um sistema deve crescer continuamente para manter a satisfação do usuário durante a sua vida útil.

Alguns fatores que dificultam e encarecem a manutenção de software [PIGOSKI, 1996] são: a dificuldade de entender o programa “de outra pessoa”; a inexistência de documentação e o fato do software não ser projetado para sofrer mudanças.

3 Métricas de software

Segundo Fenton e Pfleeger (1997) medição de software é o processo através do qual números e símbolos são atribuídos do mundo real de forma a tornar possível caracterizar cada entidade através de regras claramente definidas. A medição se dá através da aplicação de métricas de processos, produtos e serviços. Segundo a norma ISO 9126 (2001), métrica é a composição de métodos para medição e escalas de medição. Essas escalas de medição são formas de mapeamento para que, por meio da manipulação de dados numéricos, seja possível entender o comportamento das entidades.

Para que a medição seja efetiva, torna-se necessário focá-la nos objetivos a serem alcançados com tais medições [BASILI e ROMBACH, 1998]. Pfleeger (2000) destaca que é necessário deixar claras as necessidades para poder melhor conhecer os objetivos das medições. O uso de modelos pode guiar a definição de aplicação de medições a serem realizadas [PFLEEGER, 2000]. Um modelo comumente utilizado é o GQM – *Goal Question Metrics* [BASILI e ROMBACH, 1998].

A abordagem do método GQM (Goal-Question-Metric) é baseada na premissa de que uma organização para medir deve primeiro especificar os seus próprios objetivos e os objetivos de seus projetos, então deve-se traçar os objetivos para os dados que os definem operacionalmente, e finalmente prover um framework para interpretação dos dados, respeitando os objetivos estabelecidos anteriormente [BASILI e ROMBACH, 1998]. A principal idéia do GQM é que medição deve ser orientada a objetivos. É uma metodologia *top-down* em que se inicia com a definição de um objetivo de medição explícito que é

refinado em questões e cada questão, por sua vez, é refinada em métricas que quando medidas provêm informações para responder as questões. Respondendo as questões somos capazes de analisar se os objetivos foram alcançados.

4 Definição de métricas para avaliação de sistemas legados

Utilizamos a abordagem GQM para definir um conjunto de métricas que permitisse avaliar um sistema legado em um espaço curto de tempo, ou seja, em um tempo aceitável que se permita à empresa poder definir seus contratos com o cliente. Para uma empresa mantenedora o custo de uma manutenção pode vir do grande número de modificações solicitadas periodicamente, da urgência das manutenções, da dificuldade intrínseca da própria atividade e da dificuldade de entender e/ou modificar o sistema. Nosso foco ao definir o GQM foi nesse último aspecto. Ou seja, como auxiliar a empresa a conhecer o sistema que vai manter.

Inicialmente, definimos a equipe GQM, que foram os autores desse trabalho e duas pessoas especialistas em manutenção (um gerente e um mantenedor). Além disso, explicitamos nosso objetivo de negócio, ou seja, permitir à empresa mantenedora avaliar a complexidade de um sistema legado de forma a apoiar na definição de seus contratos. Um importante requisito para o plano de medição foi dar preferência às métricas automatizáveis quando possível. A razão para isso é que uma empresa mantenedora geralmente não dispõe de muito tempo para analisar o sistema antes de definir o contrato.

A partir desse objetivo de negócio a equipe GQM definiu dois objetivos de medição: avaliar a documentação e o código do sistema. Para cada objetivo definimos questões de avaliação e métricas para avaliá-las. A seguir são apresentados os dois objetivos de medição, assim como suas questões e métricas correspondentes.

4.1 Avaliação da documentação do sistema

O primeiro objetivo de medição, formalizado com a estrutura proposta em [SOLIGEN e BERGHOUT, 1999], é:

Analisar a: documentação do sistema

Com o propósito de: avaliar

Com respeito a: completude, facilidade de entendimento e consistência

Do ponto de vista de: analista de sistemas e do mantenedor

No contexto da: organização mantenedora.

Três questões foram derivadas para atender esse objetivo: (1) Qual o nível de documentação do sistema?, (2) A documentação pode ser entendida pela equipe mantenedora?, e (3) Qual a consistência da documentação?.

Para definir as métricas que respondessem essas questões tivemos que considerar o tipo de documentação que seria avaliada nesses sistemas. Definimos, então, que por nosso interesse se referir a sistemas legados, deveríamos considerar a documentação utilizada nesses sistemas que, geralmente, considerando a idade dos sistemas legados, é proveniente da análise estruturada. Consideramos então avaliar: o diagrama de contexto, diagrama de fluxo de dados nível 0, o modelo de dados (lógico/físico) e a especificação de requisitos.

Cada métrica foi definida usando um tipo de escala específico. Algumas métricas foram definidas a partir da discussão entre os membros da equipe GQM e outras foram selecionadas da literatura. Para auxiliar na interpretação das medições coletadas para as métricas, propomos quando necessário uma escala ordinal que permita a melhor avaliação. A Tabela 1 apresenta as métricas definidas para as questões 1. De forma semelhante foram definidas 4 métricas para a questão 2 e 4 métricas para a questão 3.

Tabela 1 – Métricas do Obj. 1 – Questão 1: Qual nível de documentação do sistema?

Métrica	Fórmula	Interpretação
M1.1.1 Porcentagem de elementos do diagrama de contexto que estão documentados	$X = \{(A1 + B1) / (A + B)\} * 100$, onde: A = Total de entidades Externas B = Total de fluxos de dados A1 = Total de entidades externas documentadas B1 = Total de fluxos de dados documentados	00% a 25% - documentação insuficiente 26% a 50% - pouco documentado 51% a 80% - moderadamente documentado. 81% a 100% - bem documentado.
M1.1.2 Porcentagem de elementos do DFD que estão documentados * de preferência utilizar o DFD nível 1	$X = \{(A1 + B1 + C1 + D1) / (A + B + C + D)\} * 100$, onde: A = Total de entidades externas B = Total de fluxos de dados C = Total de processos D = Total de depósitos de dados A1 = Total de entidades externas documentadas B1 = Total de fluxos de dados documentados C1 = Total de processos documentados D1 = Total de depósitos de dados documentados	00% a 25% - documentação insuficiente 26% a 50% - pouco documentado 51% a 80% - moderadamente documentado. 81% a 100% - bem documentado.
M1.1.3 Porcentagem de elementos do modelo de dados que estão documentados * de preferência utilizar o modelo físico	$X = \{(A1 + B1 + C1) / (A + B + C)\} * 100$, onde: A = Total de tabelas B = Total de colunas C = Total de domínios A1 = Total de tabelas documentadas B1 = Total de colunas documentadas C1 = Total de domínios documentados	00% a 25% - documentação insuficiente 26% a 50% - pouco documentado 51% a 80% - moderadamente documentado. 81% a 100% - bem documentado.
M1.1.4 Indicador do nível de detalhamento dos requisitos	$X = A$, onde: A, (0) Insuficiente (1) Pouco detalhado (2) Detalhado (3) Bem detalhado	(0) Insuficiente (1) Pouco detalhado (2) Detalhado (3) Bem detalhado
M1.1.5 Porcentagem de requisitos funcionais registrados na especificação de requisitos que possuem descrição	$X = (A/B) * 100$, onde: A = Total de requisitos registrados no relatório de requisitos funcionais do software, que possuem descrição. B = Total de requisitos registrados no relatório de requisitos funcionais.	00% a 25% - Descrição insuficiente 26% a 50% - baixo grau de descrição. 51% a 80% - descrição moderada 81% a 100% - alto grau de descrição
M1.1.6 Porcentagem de descrição de requisitos funcionais que possuem descrição das regras de negócio	$X = A/B * 100$, onde: A = Total de requisitos funcionais registrados na especificação de requisitos que possuem descrição das regras de negócio; B = Total de requisitos funcionais registrados na especificação de requisitos.	00% a 25% - insuficiente 26% a 50% - baixo grau de descrição. 51% a 80% - descrição moderada 81% a 100% - alto grau de descrição

4.2 Avaliação do código do sistema

O segundo objetivo de medição, formalizado com a estrutura proposta em [SOLIGEN e BERGHOUT, 1999], é:

Analisar o: sistema implementado

Com o propósito de: avaliar

Com respeito a: complexidade

Do ponto de vista do: analista de sistemas e do mantenedor

No contexto da: organização mantenedora.

Cinco questões foram derivadas para atender esse objetivo: (1) Qual o tamanho do sistema?; (2) Qual o nível de interação no sistema?; (3) Qual a complexidade do sistema implementado?; (4) Qual a complexidade de interface com o usuário?; e (5) Qual a complexidade de interface com outros sistemas?

Para definir métricas para essas questões consideramos que a complexidade de manutenção está relacionada à dificuldade de entender/manter um sistema considerando o número de linhas de código (total e por programa/módulo), número de tabelas que utiliza, número de telas a serem manipuladas (total e por programa), quantidade de tecnologias envolvidas, grau de acoplamento entre os programas e a própria complexidade do código. Para definir essas métricas utilizamos novamente a experiência da equipe GQM e a literatura relevante. Para essas métricas não foi possível definir limites de valores que nos ajudasse na interpretação. Esses valores devem ser definidos a medida em que os sistemas sejam avaliados considerando uma base histórica dos mesmos. No entanto, para auxiliar na interpretação, destacamos algumas observações da literatura sobre as métricas coletadas. A Tabela 2 apresenta as métricas da questão 2 desse objetivo. De forma semelhante foram definidas 11 métricas para a questão 1, 14 métricas para a questão 3 (uma reutilizada), 4 métricas para a questão 4 (uma reutilizada) e 4 métricas para a questão 5 (uma reutilizada).

Tabela 2 – Métricas do Obj. 2 – Questão: Qual o nível de interação no sistema?

Métrica	Fórmula	Observações para Interpretação
M2.2.1 Média de tabelas do próprio sistema lidas por programa	$X = A / M2.1.8$, onde: A = Número de tabelas próprias do sistema acessadas para leitura M2.1.8 = Número de programas do sistema	Quanto maior a quantidade de tabelas acessadas, maior será a complexidade. Baseada na característica geral da APF “Modificação facilitada”
M2.2.2 Média de tabelas do próprio sistema modificadas por programa	$X = A / M2.1.8$, onde: A = Número de tabelas próprias do sistema acessadas para inserção, alteração ou exclusão de dados. M2.1.8 = Número de programas do sistema	Quanto maior a quantidade de tabelas acessadas, maior será a complexidade. Baseada na característica geral da APF “Modificação facilitada”
M2.2.3 Fan-out médio por programa	$X = A / M2.1.8$, onde: A = Total de fan-out M2.1.8 = Número de programas do sistema	Quanto maior o valor do fan-out, menor será a manutenibilidade do sistema.
M2.2.4 Distribuição fan-out por programa.	Para cada programa, $X = A$, onde: A = Total de fan-out do programa.	Quanto maior o valor do fan-out, menor será a manutenibilidade do sistema. O fan-out não deveria ultrapassar o valor 7 (PAGE-JONES, 1988)

A definição completa das métricas pode ser encontrada em [RAMOS, OLIVEIRA, ANQUETIL, 2004a¹].

5 Aplicação Prática

Este trabalho foi validado na Politec Ltda, uma grande empresa de software que possui mais de 6.000 funcionários distribuídos em diversas filiais no Brasil, sendo que duas delas possuem uma área especializada em manutenção de software.

As métricas de documentação (objetivo de medição 1) foram coletadas através de formulários manuais (ver Figura 1 – exemplo parcial), enquanto que as métricas do sistema implementado (objetivo de medição 2) foram coletadas com a utilização de um programa construído especificamente para coleta de dados em código-fonte de sistema implementado com a linguagem de programação Cobol. O programa de coleta de dados foi desenvolvido com o gerador de compilador JavaCC¹.

FORMULÁRIO PARA COLETA DE DADOS -MÉTRICAS DE DOCUMENTAÇÃO
1- DIAGRAMA DE CONTEXTO
O diagrama de contexto está disponível para consulta? Sim/Não Se não, vá para o item 2
Qual a quantidade total de entidades externas representadas no diagrama de contexto?
Qual a quantidade total de fluxos de dados representados no diagrama de contexto?
Qual a quantidade total de entidades externas do diagrama de contexto que possuem descrição?
Qual a quantidade total de fluxos de dados do diagrama de contexto que possuem descrição?
Você considera que o diagrama de contexto pode ser:
(3) facilmente compreendido (2) compreendido (1) dificilmente compreendido (0) insuficiente

Figura 1 – Exemplo parcial do formulário para coleta de dados – métricas de documentação

Para a realização da coleta de dados, foram avaliados 10 sistemas que possuem pelo menos um módulo desenvolvido em Cobol (CICS ou Batch), a idade varia entre 5 e 10 anos e possuem mais de uma linguagem de programação envolvida na sua implementação. No entanto, apenas a parte Cobol dos sistemas foi avaliada neste trabalho. Além disso, as métricas que avaliam a complexidade de interface com o usuário não foram coletadas pelo fato de não ter sido disponibilizado o código das interfaces dos programas. Os resultados das medições podem ser consultados nas Tabelas 3 e 4.

Como pode ser observado na Tabela 4 várias métricas são de distribuição, sendo analisadas a partir de gráficos como os apresentados nas Figuras 2 e 3 para o sistema S2. A Figura 2 mostra a distribuição da métrica 2.3.6 (Distribuição de linhas de comentário/programa). Esta métrica apresentou um comportamento comum nos 10 programas analisados em que a quantidade de linhas de comentário cresce pouco se comparado ao crescimento da quantidade de linhas de código dos programas. Isso significa que os programas maiores, que precisariam de mais documentação por que são potencialmente mais difíceis de entender, proporcionalmente são os que têm o menor percentual de linhas de comentário.

¹ Esta ferramenta pode ser encontrada no site: <https://javacc.dev.java.net/>.

IV SBQS - IV Concurso de Teses e Dissertações em Qualidade de Software

Tabela 3 – Resultado das métricas do objetivo de medição 1 – Questões 1, 2 e 3

		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Questão 1	M1.1.1 – Porcentagem de elementos do diagrama de contexto que estão documentados	0%	0%	-	-	0%	0%	0%	0%	-	-
	M1.1.2 - Porcentagem de elementos do DFD que estão	0%	0%	-	-	0%	0%	0%	0%	-	-
	M1.1.3 - Porcentagem de elementos do modelo de dados que estão documentados	-	-	-	-	-	-	-	0%	0%	-
	M1.1.4 - Indicador do nível de detalhamento dos	3	-	1	-	2	0	2	2	3	-
	M1.1.5 - Porcentagem de requisitos funcionais registrados na especificação de requisitos que possuem	100%	-	50%	-	100%	100%	100%	74%	100%	-
	M1.1.6 – Porcentagem de descrição de requisitos funcionais que possuem descrição das regras de negócio	100%	-	100%	-	100%	0%	98%	38%	100%	-
Questão 2	M1.2.1 - Indicador de facilidade de entendimento do diagrama de contexto	2	2	-	-	2	2	2	1	-	-
	M1.2.2 - Indicador de facilidade de entendimento do	2	2	-	-	3	2	2	1	-	-
	M1.2.3 - Indicador de facilidade de entendimento do dicionário do modelo de dados	-	-	-	-	-	-	-	-	-	-
	M1.2.4 - Indicador de facilidade de entendimento da especificação de requisitos funcionais.	1	-	2	-	3	0	2	2	3	-
Questão 3	M1.3.1 - Porcentagem de integrações com outros sistemas na especificação de requisitos que estão consistentes com o diagrama de contexto.	83%	-	0%	-	-	-	0%	-	0%	-
	M1.3.2 - Porcentagem de integrações com outros sistemas na especificação de requisitos que estão	83%	-	0%	-	-	-	0%	-	0%	-
	M1.3.3 - Porcentagem de elementos do modelo físico de dados que estão consistentes com o banco de dados	-	-	-	-	-	-	-	-	-	-
	M1.3.4 - Porcentagem de todas as entidades externas representadas no DFD que estão representadas no diagrama de contexto	100%	100%	-	-	33%	42%	100%	-	0%	-

Legenda: M1.2.1 a M1.2.4: 0- Insuficiente 1- Difícilmente compreendido 2- Compreendido 3- Facilmente compreendido
M1.1.4: 0 – Insuficiente 1 – Pouco detalhado 2 – Detalhado 3 – Bem detalhado

IV SBQS - IV Concurso de Teses e Dissertações em Qualidade de Software

Tabela 4 - Resultado das métricas do objetivo de medição 2 – Questões 1, 2, 3 e 5

		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Questão 1	M2.1.1 - Tamanho do sistema em LOC	110.817	135.686	49.646	77.650	62.276	100.236	14.528	99.517	153.395	2.781
	M2.1.2 - Média de LOC por programa	1.131	590	814	388	490	964	2.075	630	1.322	348
	M2.1.3 Distribuição de LOC por programa.	Gráfico de distribuição									
	M2.1.4 - Número de tabelas próprias do sistema	41	67	2	85	37	32	7	74	48	7
	M2.1.5 - Número de tabelas de outro sistema	12	1	0	1	10	5	0	1	0	0
	M2.1.6 - Número de LOC por módulo	1.131	590	814	388	490	964	2.075	630	1.322	348
	M2.1.7 Distribuição de LOC por Módulo.	Gráfico de distribuição									
	M2.1.8 - Número de programas do sistema	98	230	61	200	127	104	7	158	116	8
	M2.1.9 - Média de programa por módulo	98	230	61	200	127	104	7	158	116	8
	M2.1.10 Número total de telas	Não coletado									
M2.1.11 Distribuição de programas por tecnologia	Gráfico de distribuição										
Questão 2	M2.2.1 Média de tabelas próprias lidas por programa	3,26	1,77	0,06	2,34	1,45	1,47	3	3,65	3,68	0,62
	M2.2.2 Média de tabelas próprias modificadas/programa	0,83	0,75	0,03	0,56	0,41	0,26	0,14	1,05	1,01	0,87
	M2.2.3 Fan-out médio por programa	2,07	3,2	0,93	0	0,007	2,83	7	0	4,71	0
	M2.2.4 Distribuição fan-out por programa.	Gráfico de distribuição									
Questão 3	M2.3.1 Distribuição da complexidade ciclomática / prog.	Gráfico de distribuição									
	M2.3.2 Média da Complexidade ciclomática / programa.	32,74	24,26	57,95	15,74	23,78	47,52	45,28	40,80	39,06	8,25
	M2.3.3 Distribuição de desvios por programa	Gráfico de distribuição									
	M2.3.4 Média de desvios por programa.	0,33	0,10	0,95	0,0787	0,18	0,45	6,46	0,25	0,33	1,03
	M2.3.5 porcentagem de LOC comentadas do sistema	15,31%	14,57%	16,00%	14,41 %	16,61 %	21,58%	15,14%	16,26%	15,63%	25,85%
	M2.3.6 Distribuição de linhas de comentário / programa	Gráfico de distribuição									
	M2.3.7 Média de biblioteca acessada por programa	Gráfico de distribuição									
	M2.3.8 Média de IF's aninhados do programa	4,72	3,32	10,19	0,98	2,80	8,65	8	5,5	2,79	0,375
	M2.3.9 Número de linguagens de programação	Métricas coletadas só para parte Cobol									

IV SBQS - IV Concurso de Teses e Dissertações em Qualidade de Software

		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Questão 5	M2.3.10 Distribuição de programas / linguagem de prog.	Gráfico de distribuição									
	M2.2.3 - Fan-out médio por programa	2,07	3,2	0,93	0	0,007	2,83	7	0	4,71	0
	M2.3.11 Complexidade intermodular e intramodular	8,05	17,2	4,37	0	0,007	68,14	211,28	0	25,35	0
	M2.3.12 Distribuição da métrica de Halstead / programa	Gráfico de distribuição									
	M2.3.13 Distribuição do Índice de Manutenibilidade por programa	Gráfico de distribuição									
	M2.5.1 Número de tabelas lidas de outro sistema	12	1	0	1	6	5	0	1	0	0
	M2.5.2 Número de tabelas modificadas de outro sistema	6	0	0	0	5	0	0	0	0	0
	M2.5.3 Distribuição de tabelas acessadas de outros sistemas, por programa.	Gráfico de distribuição									

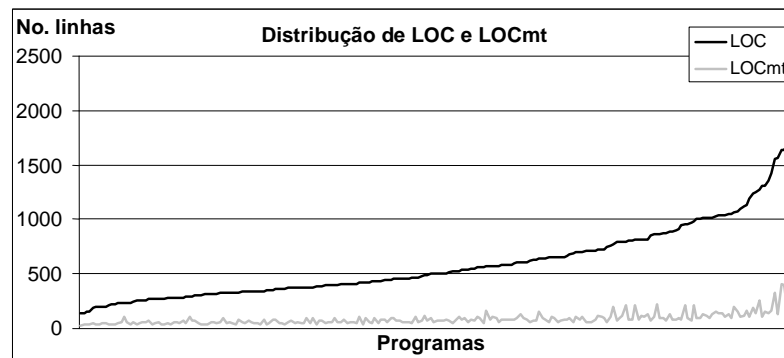


Figura 2 – Distribuição de linhas de código e linhas de comentário do S2

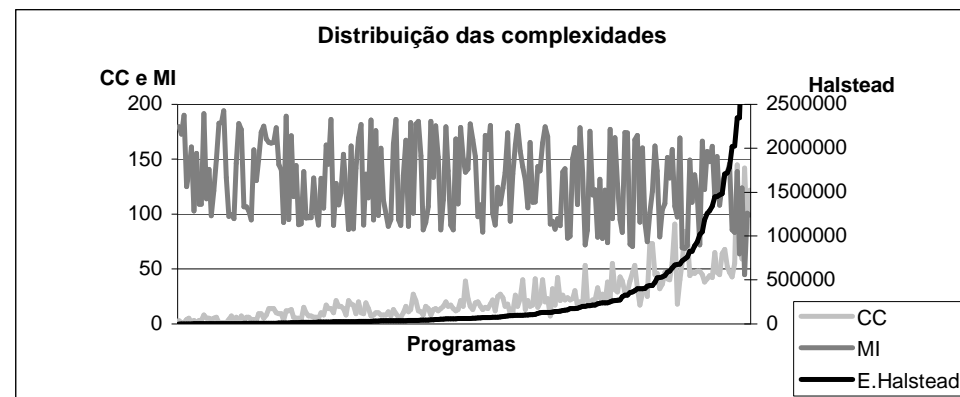


Figura 3 – Distribuição de complexidades do S2

A Figura 3 mostra a distribuição das métricas de Complexidade ciclomática [PEARSE e OMAN, 1995; FRAPIER et al, 1994], métrica de Esforço de Halstead [PEARSE e OMAN, 1995], e Índice de Manutenibilidade por programa [PEARSE e OMAN, 1995]. Vale lembrar que quanto maior os valores encontrados para as métricas de complexidade ciclomática e de esforço de Halstead, maior é a complexidade do sistema, enquanto que, quanto maior o valor do Índice de Manutenibilidade, maior será a facilidade de manutenção do sistema. [PEARSE e OMAN, 1995; FRAPIER et al, 1994]. Pode se dizer que o Índice de Manutenibilidade é uma métrica de “simplicidade” dos programas.

Analisando o gráfico da Figura 3 para o sistema S2 e os gráficos dos demais sistemas, pôde-se constatar que as três métricas apresentam coerência, isto é, quando os valores para complexidade ciclomática e de esforço de Halstead tendem a crescer, o valor do índice de manutenibilidade tende a diminuir. Seguindo a interpretação do Índice de Manutenibilidade de PEARSE e OMAN (1995), programas com índice maior a 85 são de fácil manutenção. Pôde-se constatar que a maioria dos programas está nesta condição. Contudo, vale lembrar que esse índice foi estabelecido com sistemas da HP e que para uma melhor precisão, a fórmula precisaria ser adaptada a cada organização. Isso apresentaria dificuldades particulares para uma mantenedora terceirizada que deve lidar com sistemas de vários clientes com culturas e características potencialmente muito diferentes.

Após a coleta e interpretação dos dados foi realizada uma sessão de *feedback* com a participação de parte da equipe de GQM (pesquisadores envolvidos nessa dissertação) e dois profissionais de manutenção: o gerente da área em que os sistemas foram avaliados e um profissional com grande experiência em desenvolvimento e manutenção de sistemas COBOL (mais de 10 anos de experiência). Os resultados preliminares desse modelo de avaliação podem ser encontrados em [RAMOS, OLIVEIRA e ANQUETIL 2004a], ou [RAMOS, OLIVEIRA e ANQUETIL 2004b]. Maiores detalhes de como foi realizada a aplicação prática da pesquisa podem ser encontrados em [RAMOS 2004].

Durante a sessão de *feedback*, as informações geradas a partir da coleta de dados foram apresentadas para os profissionais de manutenção que analisaram e discutiram a situação de cada sistema e, com base em suas experiências e expectativas enquanto profissionais de uma organização mantenedora, eles classificaram a documentação e a complexidade do sistema implementado conforme as escala da tabela 5, respondendo assim os dois objetivos de medição. Para a classificação, inicialmente um membro da equipe GQM lia as informações das métricas explicando o significado dos resultados. Em seguida, os dois profissionais de manutenção opinavam e discutiam entre eles as características do sistema. Somente, no caso de não haver consenso os pesquisadores também opinavam dando justificativas. Ao final, se chegava a um consenso da classificação a ser definida.

Tabela 5 - Escalas de classificação da documentação e da complexidade do sistema

Documentação	Complexidade do Sistema
Baixa – baixa qualidade / insuficiente (*)	Baixa – complexidade baixa
Moderada – qualidade moderada	Moderada – complexidade média
Alta – qualidade alta	Alta – complexidade alta

(*) insuficiente indica que nenhuma documentação foi elaborada, com exceção do modelo de dados.

A documentação do sistema S1 obteve a classificação ALTA por ter todos os documentos esperados pelo modelo de avaliação e, principalmente, por ter um bom nível de detalhamento dos requisitos. Mesmo tendo sido indicado que os requisitos eram de difícil compreensão, foi considerado que essa dificuldade poderia ser em razão do domínio da aplicação ser desconhecido pelos avaliadores ou o negócio ser realmente de difícil entendimento. Com relação à complexidade, o sistema obteve a classificação MODERADA, pois apesar dos programas terem sido considerados pequenos, o sistema acessa um alto número de tabelas de outros sistemas.

O sistema S2 teve a documentação classificada como BAIXA, pois não tem a especificação de requisitos, que segundo os profissionais de manutenção, é um produto de documentação de extrema importância para a equipe que fará manutenção no sistema. A existência dos outros documentos (Diagrama de contexto e DFD) não elimina a necessidade de ter a especificação de requisitos. A complexidade do sistema foi classificada como MODERADA por ter muitos programas (já que para sistemas COBOL, normalmente, cada programa representa uma funcionalidade, o que implica portanto em muitas funcionalidades). Além disso, o sistema possui muitas tabelas, a média de complexidade ciclomática do sistema é “complexa” e a média de *fan-out* por programa apresentava um valor alto considerando o tamanho médio dos programas (590 linhas de código).

O sistema S3 possui apenas a especificação de requisitos, que, como citado anteriormente, é considerada como muito importante para a organização mantenedora. No entanto, este produto de documentação está pouco detalhado, por isso a documentação deste sistema foi classificada como BAIXA. A complexidade do sistema foi considerada também como BAIXA mesmo apresentando um alto valor médio de complexidade ciclomática, em virtude de possuir poucos programas e a média de tabelas acessadas para leitura/modificação ser baixa e não haver integração com outros sistemas legados.

O sistema S4 não apresentou nenhum documento para avaliação sendo, portanto, classificado como BAIXA (no caso, insuficiente). A complexidade é MODERADA por ter muitos programas e muitas tabelas mantidas pelo sistema.

A documentação do sistema S5 obteve a classificação ALTA por ter todos os documentos esperados e, principalmente, por ter um bom nível de detalhamento dos requisitos. Com relação à complexidade, o sistema implementado obteve a classificação MODERADA, por ter que acessar várias tabelas de outros sistemas legados. Além disso, 5 tabelas de outros sistemas são atualizadas pela aplicação. Os profissionais de manutenção consideram que atualizar tabelas de outro sistema torna o sistema mais difícil de se manter.

O sistema S6 teve a documentação classificada como MODERADA por apresentar o diagrama de contexto e o diagrama de fluxo de dados consistentes e moderadamente compreendidos, mesmo que o detalhamento da especificação de requisitos tenha se mostrado insuficiente. Nesse caso, o sistema apresentava uma especificação de programas que era difícil de ser entendida. Já a complexidade do sistema foi classificada como MODERADA com tendência para BAIXA. Moderada por causa da média de *fan-out* por programa e do alto número de GOTOS. E baixa porque, apesar das características anteriores, acessa outros sistemas apenas para leitura de tabelas.

A documentação do sistema S7 obteve a classificação ALTA por ter todos os documentos esperados pelo modelo de avaliação e, principalmente, por ter um bom nível de detalhamento dos requisitos. Com relação à complexidade, o sistema obteve a classificação ALTA, por ter um valor alto para a média de *fan-out* e de GOTOs. Outro fato destacado foi a média de 5 tabelas por programa das quais 2 são mantidas. Finalmente, a complexidade ciclomática é considerada ainda “intestável” (ou seja, > 50). Os profissionais de manutenção chegaram a afirmar que este sistema seria um bom candidato a ser refeito.

O sistema S8 teve a documentação classificada como MODERADA, pois apesar de possuir todos os produtos de documentação necessários, o diagrama de contexto e o diagrama de fluxo de dados são de difícil compreensão e os requisitos moderadamente compreendidos. Já a complexidade do sistema foi classificada como MODERADA pois, apesar de ter apenas uma tabela de integração com outro sistema e ter *fan-out* nulo, a média de GOTOs por programa (que são considerados pequenos) é relevante (3 GOTOs em média) e possui muitas tabelas (com uma média de 5 acessos a tabelas por programa).

O sistema S9 obteve classificação MODERADA para a documentação, pois apesar de possuir apenas a especificação de requisitos, estes se apresentaram bem detalhados e facilmente compreendidos. A complexidade do sistema foi classificada como MODERADA mesmo não fazendo acesso à nenhuma tabela de outro sistema. No entanto, acessa em média 6 tabelas por programa e tem *fan-out* médio alto (6).

Assim como o sistema S4, pelo fato de não apresentar nenhum documento para avaliação, a documentação do sistema S10 foi classificada como BAIXA (insuficiente). No entanto, a complexidade do sistema foi classificada como BAIXA por ter um baixo número de programas, média de complexidade ciclomática simples e poucos acessos a tabelas.

No final da sessão de *feedback*, foi realizada uma análise da classificação da complexidade e da documentação do sistema implementado, verificando como essas informações apoiariam na definição de um contrato. Foi definida, então uma classificação, seguindo a mesma escala, para definição de impacto (Baixo, Médio ou Alto) em contratos de manutenção. A Tabela 6 indica essa classificação a partir da análise das informações da documentação e da complexidade do sistema.

Tabela 6 – Escala para avaliação do impacto da documentação e da complexidade do sistema em um possível contrato de manutenção.

Sistema	Alta complexidade	Moderada complexidade	Baixa complexidade
Documentação			
Alta qualidade	Moderado	Moderado	Baixo
Moderada qualidade	Alto	Moderado	Moderado
Baixa qualidade	Alto	Alto	Moderado

Observe na Tabela 6 que um sistema com documentação de alta qualidade, mas moderada/alta complexidade levaria a um impacto moderado no contrato. No entanto, observe também que apenas pelo fato da documentação cair em qualidade (ou seja, ser de moderada qualidade), para um sistema implementado de alta complexidade, o impacto no

contrato passaria a ser alto. Outro aspecto interessante, é que quando a documentação é de baixa qualidade, exceto para o caso em que o sistema também é de baixa complexidade, o impacto no contrato seria alto, pois os profissionais de manutenção consideram que seria trabalhoso mantê-lo enquanto a documentação é elaborada/atualizada. Assim, em futuras manutenções o impacto no contrato poderá ser menor pois a documentação já será de melhor qualidade. Foi destacado pelos profissionais de manutenção que outros aspectos, além da avaliação do sistema, são considerados quando da definição de um contrato, como por exemplo oportunidade de mercado, dentre outros.

Com base na Tabela 6 os sistemas foram classificados conforme apresentado na tabela 7. Por exemplo, o sistema S1 que tem alta qualidade de documentação e moderada complexidade, teve o impacto classificado como moderado.

Tabela 7 – Classificação do impacto nos contratos dos sistemas avaliados

Sistema	Documentação	Sistema Implementado	Impacto
S1	Alta qualidade	Moderada Complexidade	Moderado
S2	Baixa qualidade	Moderada Complexidade	Alto
S3	Baixa qualidade	Baixa Complexidade	Baixo
S4	Insuficiente qualidade	Moderada Complexidade	Alto
S5	Alta qualidade	Moderada Complexidade	Moderado
S6	Moderada qualidade	Moderada à Baixa Complexidade	Moderado
S7	Alta qualidade	Alta Complexidade	Moderado
S8	Moderada qualidade	Moderada Complexidade	Moderado
S9	Moderada qualidade	Moderada Complexidade	Moderado
S10	Insuficiente qualidade	Baixa Complexidade	Moderado

6 Conclusões e Lições Aprendidas

Conforme destacado nas seções anteriores, esse trabalho definiu um modelo de métricas para avaliar sistemas legados que pode apoiar uma empresa mantenedora na definição dos seus contratos. Pode-se destacar que essa avaliação requereu baixo esforço, considerando que se gastou em média 60 minutos para avaliação de cada um dos oito sistemas (visto que dois deles não foram avaliados por falta de documentação), ou seja, em aproximadamente 1 hora teve-se uma idéia sobre a qualidade da documentação do sistema.

Com a ferramenta implementada para a coleta de métricas de código, pôde-se ainda ter um conjunto de métricas que deram uma boa visão sobre a complexidade do sistema implementado. Os profissionais de manutenção que participaram do trabalho, na análise final sobre os sistemas demonstraram claramente que essas informações seriam de grande valor na definição de seus contratos, o que atende ao objetivo de do GQM.

No entanto, com a aplicação das métricas e o acompanhamento da análise final dos sistemas, pôde-se listar algumas limitações desse trabalho:

- O percentual de comentários em programa, apesar de ser interessante conhecer, não se mostrou útil na avaliação final. Isso se deve ao fato de que comentários são difíceis de serem avaliados por ferramentas automatizadas. Os profissionais de manutenção destacaram, que apesar das pessoas manterem os sistemas, geralmente, elas não mantêm os comentários, e, além disso, algumas vezes, os comentários são apenas linhas de código comentado. A partir de uma discussão da equipe GQM com os profissionais de manutenção na análise final dos sistemas, foi sugerido, que se procurasse separar o percentual de linhas de comentário referentes ao prólogo do programa, as linhas de comentário que são de código comentado e as linhas que são efetivamente de comentário. Essa separação pode ter um significado melhor do que o simples percentual de linhas de comentário.
- O uso de tabelas foi essencial para a classificação da complexidade do sistema implementado. No entanto, considerando sistemas legados, foi percebido através da avaliação do sistema S3, que só tinha 2 tabelas, que seria importante também avaliar os arquivos acessados e não apenas as tabelas. Esse sistema, por exemplo, tinha 61 programas e somente 2 tabelas, o que pode indicar que tinha muitos arquivos.
- A consistência entre especificação de requisitos e diagrama de contexto e DFD foi feita considerando apenas se as entidades externas declaradas na especificação estavam presentes nos diagramas. Essa foi uma decisão tomada pelo fato de estar considerando que a avaliação da documentação deveria ser realizada em um tempo aceitável para apoiar a definição de um contrato. No entanto, poderia ser avaliado o esforço necessário para também se verificar se as funcionalidades descritas na especificação de requisitos estão declaradas como processos nos DFD. Essa avaliação não é trivial porque os nomes definidos nos processos são sempre muito simples, e pelo que foi constatado nas avaliações dos sistemas, embora existam os diagramas nem sempre existem as descrições (dicionário de dados) desses diagramas.
- A métrica de índice de complexidade proposta por Pearse e Oman (1995), foi incluída no modelo por ser uma métrica de literatura que indica uma informação importante para apoiar a definição de um contrato, que seria um índice que indica o grau de manutenibilidade (facilidade de manter) um sistema. No entanto, o resultado dessa métrica não foi muito efetivo e realmente de difícil avaliação. Dessa forma, deve-se avaliar a possibilidade de excluir essa métrica do modelo final.
- Para várias métricas de código foi utilizada a distribuição e a média. A distribuição foi muito útil quando se comparou métricas entre si. No entanto, a média para a análise final foi também bastante útil.
- Finalmente, a avaliação da especificação de requisitos é muito subjetiva, dependendo do avaliador. Nesse caso, a organização deve definir uma equipe que avalie a documentação segundo critérios e padrões pré-estabelecidos e segundo um treinamento que permita uma avaliação uniforme.

Agradecimentos

Agradecemos ao Centro de Sustentação de Sistemas da Politec, pelo apoio na realização prática deste trabalho. Este trabalho foi suportado pelo CNPq.

Referências Bibliográficas

- BASIL, V. e ROMBACH, H. **“Goal question metric paradigm”**, Encyclopedia of software engineering – 2, 1994.
- FENTON, N.E, PFLEEGER, L., **“Software metrics a rigorous and practical approach”**, PWS Publishing Company, 1997
- FRAPIER, Marc., MATWIN, Stan., MILI, Ali. **“Software metrics for predicting maintainability** - Software metrics study: Technical Memorandum 2”, Canadian Space Agency, 1994
- ISO 9126 **“Software engineering – Product quality – Part 1”**, 2001
- KHOSHGOFTAAR, T.M., ALLEN, E.B., HALSTEAD, R., e TRIO G.P., **“Detection of fault-prone software modules during a spiral life cycle”**, International Conference on Software Maintenance (ICSM '96), pp. 69-76, Nov. 04-08, 1996, Monterey, CA, 1996.
- LEHMAN, M.M., PERRY, D.E, RAMIL, J.F, **“Implications of evolution metrics on software maintenance”**, IEEE Transaction on Software Engineering, 1998.
- PAGE-JONES, MEILIR., **“Projeto estruturado de sistemas”**., McGraw-Hill Ltda., 1988.
- PEARSE, TROY E OMAN, PAULO. **“Maintainability measurement on industrial source code maintenance activities”**, IEEE Transaction on Software Engineering, 1995.
- PFLEEGER, S., **“Use realistics, effective software measurement”** cap. 8, in: **“Constructing Superior Software”**, Eds. CLEMENTS, Paul C., BASS Len, BELADY Les, *et al*, Software Quality Institute, 2000.
- PIGOSKI, T.M., **“Practical Software Maintenance”** John Wiley & Sons, Inc., 1996
- POLO, M., PIATTINI, M., RUIZ, F., **“Using code metrics to predict maintenance of legacy programs: a case study”**, IEEE Transaction on Software Engineering, 1998.
- RAMOS, C.S., OLIVEIRA, K.M., ANQUETIL, N. (2004a), **“Conhecendo Sistemas Legados através de Métricas de Software”**, III Simpósio Brasileiro de Qualidade de Software, 2004
- RAMOS, C.S., OLIVEIRA, K.M., ANQUETIL, N. (2004b), **“Legacy Software Evaluation Model for Outsourced Maintainer”**, In: 8th European Conference on Software Maintenance and Reengineering, Tampere, Finlândia. IEEE Computer Society, 2004, p.48–57
- RAMOS, C.S., **“Avaliação de sistemas legados”**, Mestrado em Gestão do Conhecimento e Tecnologia da Informação, Universidade Católica de Brasília, agosto 2004.
- SOLIGEN, R. e BERGHOUT, E., **“The goal/question/metric method – A practical guide for quality improvement of software development”**. Great Britain: Cambridge, McGraw-Hill, 1999.