Estimação do esforço de desenvolvimento de um sistema de software com *Use Case Points*: Análise de um Caso de Aplicação

CAPSI'2012

André Sousa [1], Ricardo J. Machado [2], Pedro Ribeiro [3].

1) Departamento de Sistemas de Informação, Universidade do Minho, Guimarães, Portugal
pg20115@alunos.uminho.pt

2) Centro ALGORITMI, Universidade do Minho, Guimarães, Portugal
rmac@dsi.uminho.pt

3) Centro ALGORITMI, Universidade do Minho, Guimarães, Portugal
pmgar@dsi.uminho.pt

Resumo

Os modelos de Casos de Uso são frequentemente utilizados em análise orientada a objectos para capturar o âmbito funcional de um sistema de software. Existem vários métodos para estimar o esforço de um projecto de desenvolvimento de software que são baseados em atributos dos modelos de Casos de Uso. Este trabalho descreve a aplicação do método *Use Case Points* para estimar o esforço de um projecto de desenvolvimento de software para um protótipo desenvolvido em contexto educacional com um cliente real a partir dos modelos de Casos de Uso que capturaram os requisitos do sistema. Apesar de a estimativa obtida apenas poder ser comprovada (calibrada) em contexto de execução de projecto, o método é um instrumento útil para suportar o processo de estimação do esforço de desenvolvimento. Conclui-se também que a qualidade da escrita dos Casos de Uso é um factor essencial na precisão da estimativa.

Palavras chave: Modelos de Casos de Uso, *Use Case Points*, Estimação, Esforço

1. Introdução

Os modelos de casos de uso são artefactos que capturam os requisitos funcionais de um sistema de software. Ao capturarem o âmbito funcional podem ser utilizados para obter uma medição do tamanho funcional do sistema a desenvolver. Assim sendo, obtendo uma medida do tamanho funcional do software, se medirmos alguns factores técnicos e ambientais que afectam o projecto, é também possível derivar uma estimativa do esforço de desenvolvimento.

Resumidamente, o processo de estimação de um projecto de desenvolvimento de software envolve os seguintes passos: (1) Estimar o tamanho e complexidade funcionais do sistema de software. Para tal, existem vários métodos que resultam em métricas que fornecem estimativas do tamanho e complexidade do sistema de software entre os quais se destacam os métodos

Function Points[1], *Lines of Code*[2] e *Use Case Points*; (2) Estimar o esforço em horas (i.e., transformar as estimativas relativas a características do sistema a projectar em estimativas de esforço para o concretizar); (3) Estimar a duração do projecto (i.e., transformar a estimativa de esforço numa estimativa relativa ao tempo necessário para concretizar no terreno o esforço estimado); (4) Estimar o custo financeiro do projecto (i.e., transformar os esforço e tempo estimados numa estimativa do custo em que incorre o projecto se for executado com base nos pressupostos associados às primeiras estimativas referidas).

O estudo realizado envolveu a aplicação do método *Use Case Points* de forma a obter uma estimativa do esforço de desenvolvimento de um sistema de software com o intuito de suportar três processos de negócio de uma Instituição Privada de Solidariedade Social O restante artigo está organizado da seguinte forma. A secção 2 descreve o método dos *Use Case Points*. A secção 3 descreve o estudo realizado e os resultados obtidos. A secção 4 apresenta as conclusões retiradas do estudo e os objectivos para a 2º fase do trabalho de investigação.

2. Método *Use Case Points*

A primeira descrição do método foi publicada por Gustav Karner [Karner 1993] com o objectivo de criar um modelo que permitisse estimar os recursos necessários para desenvolver um sistema de software no âmbito do *Objectory Process* [Jacobson *et al*. 1992]. No entanto, após os trabalhos originais, o autor não fez mais nenhuma publicação, pelo que o método foi evoluindo sem o envolvimento do autor original [Anda et al. 2001], [Nageswaran 2001]. O método consiste em calcular uma métrica denominada *Use Case Points* que fornece uma aproximação do tamanho e complexidade de um projecto de desenvolvimento de um sistema de software. Ao número de *Use Case Points* (UCPs) obtido é aplicado um factor ou índice de conversão que resulta numa estimativa do esforço em horas que é necessário para desenvolver o software, sendo que este factor pode ser calculado com base em projectos anteriores. Este factor expressa a produtividade da equipa em horas por cada UCP, podendo ser utilizado para calibrar futuras estimativas. Os *Use Case Points* estão associados à complexidade de natureza funcional, técnica e ambiental do projecto, pelo que o número de *Use Case Points* é caracterizado por uma equação em função dos seguintes aspectos:

- O número e a complexidade dos actores no sistema;
- O número e a complexidade dos casos de uso no sistema;
- Vários requisitos não funcionais com impacto no projecto (e.g. portabilidade, segurança, etc.);
- O impacto do ambiente onde o projecto vai ser desenvolvido.

Aplicando o método, começa-se por calcular a complexidade dos actores e dos casos de uso do sistema, resultando as variáveis *Unadjusted Actor Weight* (UAW) e *Unadjusted Use Case Weight* (UUCW), respectivamente. Combinando o seu peso, obtem-se uma medida desajustada do tamanho e da complexidade funcionais do sistema denominada *Unadjusted Use Case Points* (UUCP). O passo seguinte corresponde ao ajuste do UUCP com um conjunto de factores técnicos e ambientais, denominados, respectivamente, pelas variáveis *Technical Complexity Factor* (TCF) e *Environmental Factor* (EF), que combinadas com a variável UUCP nos produzem o número efectivo de *Use Case Points* do projecto. De seguida, apresenta-se, detalhadamente, como se calculam cada uma das variáveis.

[1] http://www.ifpug.org/
[2] http://www.aivosto.com/project/help/pm-loc.html

2.1. *Unadjusted Actor Weight* (UAW)

Um actor é uma entidade que interage com o sistema para solicitar serviços (funcionalidades implementadas no sistema). Um actor pode ser uma pessoa, uma máquina, uma aplicação, etc. Assim sendo, para capturar estas diferenças, cada actor é classificado como simples, mediano ou complexo e a cada tipo de actor está associado um peso como apresenta a tabela 1. Apesar de se poder considerar que a implementação de uma interface gráfica apresenta maior complexidade do que um ecrã em linha de comandos ou um sistema externo, não existem trabalhos publicados que justifiquem este aspecto [Alain 2010].

Categoria	Exemplo	Peso
Simples	Um sistema externo através de uma aplicação	1
Mediano	Uma pessoa que interage com um sistema através de uma linha de comandos	2
Complexo	Uma pessoa que interage com o sistema através de uma interface gráfica	3

Tabela 1 - Complexidade dos Actores

A complexidade dos actores no sistema é calculada pelo somatório dos produtos entre os actores de cada categoria com os seus pesos associados. A tabela 2 mostra como se calcula a UAW para um dado sistema.

Categoria	Peso	Total de Actores	Produto
Simples	1	5	5
Mediano	2	9	18
Complexo	3	4	12
Total			35

Tabela 2 - Cálculo do *Unadjusted Actor Weight (UAW)* para um dado sistema (exemplo)

2.2. Unadjusted Use Case Weight (UUCW)

Um caso de uso pode ser percepcionado como uma "história" sobre como os utilizadores interagem com o sistema para atingirem os seus objectivos. Existem várias formas de descrever um caso de uso. No entanto, para que a medição do número de *Use Case Points* do projecto não seja afectada, todos os casos de uso do sistema devem ser descritos ao mesmo nível de detalhe. O formato orientado aos objectivos do utilizador descrito em [Cockburn 2001] é adequado para aplicar o método. Da mesma forma que os actores são classificados segundo a sua complexidade, os casos de uso também são classificados como simples, medianos ou complexos, mediante o número de transacções que contenham no cenário principal de sucesso e nos fluxos alternativos, conforme apresenta a tabela 3. É importante realçar que uma transação pode não ser um passo nos fluxos dos casos de uso. Uma transacção é uma ação desencadeada pelo utilizador à qual o sistema reage e apresenta os resultados do seu processamento. Note-se o seguinte exemplo de um fluxo num caso de uso:

1. O utilizador seleciona opções A e B da lista Z.
2. O sistema valida os dados e apresenta os resultados.

O passo 1 e 2 não podem ser considerados como transacções individualmente, apenas a sua combinação é que perfaz uma transacção.

Categoria	Número de transacções	Peso
Simples	3 ou menos	5
Mediano	4 a 7	10
Complexo	Mais de 7	15

Tabela 3 - Complexidade dos Casos de Uso

Para calcular esta variável, contam-se todos os casos de uso em cada categoria e multiplicam-se pelos seus pesos. O somatório dos totais fornece o valor da variável UUCW conforme ilustrado na tabela 4 para um dado sistema (exemplo).

Categoria	Peso	Total de Casos de Uso	Produto
Simples	5	7	35
Mediano	10	6	60
Complexo	15	10	150
Total			245

Tabela 4 - Cálculo do Unadjusted *Use Case Weight (UUCW)* para um dado sistema (exemplo)

2.3. Unadjusted Use Case Points (UUCP)

Tendo determinado a complexidade dos actores e dos casos de uso do sistema pela combinação das respectivas variáveis, determina-se uma aproximação desajustada do número de *Use Case Points* no sistema pela variável *Unadjusted Use Case Points (UUCP)*. Considera-se que é uma aproximação desajustada, porque ainda não considera a influência do conjunto de factores técnicos e ambientais do projecto. O cálculo dos *UUCP* é concretizado através da seguinte equação:

$$UUCP = UAW + UUCW \quad (1)$$

Substituindo os valores das tabelas 2 e 4, obtém-se, para o exemplo, o seguinte número de *Use Case Points* (ainda "desajustados"):

$$UUCP = 35 + 245 = 280$$

2.4. Technical Complexity Factor (TCF)

A estimação do esforço para desenvolver um sistema está para além do seu tamanho funcional. Por exemplo, o esforço para desenvolver um sistema distribuído é maior do que o esforço para desenvolver um sistema não distribuído. O esforço para desenvolver um sistema concorrente é maior que o esforço para desenvolver um sistema utilizado para um único utilizador. Há, portanto, um conjunto de factores de natureza técnica (ou não-funcionais), apresentados na tabela 5, que é necessário avaliar para obter uma estimativa mais precisa do esforço do projecto.

Factor	Descrição	Peso	Influência (sistema exemplo)
T1	Sistema distribuído	2	0
T2	Performance	2	5
T3	Eficiência	1	5
T4	Complexidade de processamento	1	3
T5	Reutilização de código	1	0
T6	Instalação	0.5	5
T7	Usabilidade	0.5	5
T8	Portabilidade	2	3
T9	Manutenção	1	5
T10	Concorrência	1	3
T11	Segurança	1	5
T12	Acessibilidade de terceiros	1	3
T13	Formação	1	3

Tabela 5 - Factores técnicos com impacto no projecto

A influência de cada factor no projecto pode assumir valores entre 0 (irrelevante) a 5 (essencial). Somando os produtos entre o valor das influências e o peso dos factores associados resulta no *TFactor*, que é utilizado para calcular a variável *Technical Complexity Factor (TCF)* através da seguinte equação:

$$TCF = 0.6 + (0.01 \times TFactor) \quad (2)$$

Karner baseou-se nos fatores de ajustamento de *Function Points* criados por Albrecht [Albrecht 1979] para utilizar as constantes 0.6 e 0.01. Para o sistema imaginado o *TFactor é* igual a 48:

$$TCF = 0.6 + (0.01 \times 48) = 1.08 \quad (3)$$

2.5. Environmental Factor (EF)

A caracterização das equipas de desenvolvimento também é importante para estimar o esforço de desenvolvimento. O nível de motivação da equipa ou a familiaridade com um processo de desenvolvimento são exemplos de factores ambientais que têm impacto no cálculo dos *Use Case Points*. Tal como no caso dos factores técnicos, também aqui é necessário analisar a influência para o projecto dos factores ambientais apresentados na tabela 6.

Factor	Descrição	Peso	Influência (Sistema exemplo)
E1	Familiarização com um processo de desenvolvimento	1.5	3
E2	Experiência em projectos semelhantes	0.5	3
E3	Experiência com programação orientada a objectos	1	5
E4	Maturidade em análise OO	0.5	3
E5	Motivação	1	3
E6	Estabilidade dos requisitos	2	5
E7	Elementos em tempo parcial	-1	0
E8	Experiência com tecnologias adoptadas	-1	3

Tabela 6 - Factores ambientais do projecto

A soma dos produtos entre os valores das influências e os pesos dos factores associados resulta no *EFactor* que será utilizado para calcular o *Environmental Factor (EF)* pela seguinte equação:

$$EF = 1.4 + (-0.03 \times EFactor) \quad (4)$$

Para o sistema imaginado o *EFactor é* igual a 22.5:

$$EF = 1.4 + (-0.03 \times 22.5) = 1.4 + (-0.68) \approx 0.72$$

2.6. *Use Case Points* (UCP)

Tendo calculado as variáveis anteriores, o último passo consiste em determinar o número efectivo de *Use Case Points* do sistema através da seguinte equação:

$$UCP = UUCP \times TCF \times EF \quad (5)$$

Substituindo os valores das variáveis anteriormente calculadas, o número efectivo de *Use Case Points*, para o exemplo dado é o seguinte:

$$UCP = 280 \times 1.08 \times 0.72 \approx 218$$

Segundo Karner, por cada *Use Case Point* correspondem 20 horas de trabalho. Assim sendo, para o sistema imaginado como exemplo, o esforço de desenvolvimento seria aproximadamente de 4360 horas.

3. Caso de Estudo

Um caso de estudo foi desenvolvido de modo a aplicar o método *Use Case Points* tendo por objectivo estimar o esforço de um projecto de desenvolvimento de um sistema de software. O projecto é caracterizado por duas fases principais: o ante-projecto e a execução. O caso de estudo considerado neste artigo incide unicamente sobre a primeira fase e envolve uma equipa de desenvolvimento constituída por 5 alunos do segundo ano da disciplina de Processos e Metodologias de Software da Licenciatura em Tecnologias e Sistemas de Informação da Universidade do Minho. O objectivo seria estimar o esforço de desenvolvimento para a segunda fase do projecto. O projecto com origem num cliente real teve como objectivo o desenvolvimento de um sistema de software que suportasse alguns dos processos de negócio de uma Instituição Privada de Solidariedade Social. No âmbito da disciplina, o *RUP* e o *CMMI-DEV v1.2 ML2* foram adoptados como processo de desenvolvimento e modelo de qualidade a seguir pela equipa. O objectivo da equipa seria planear a fase de *Inception* do projecto e realizar diversas actividades do *RUP* nomeadamente a modelação de requisitos, necessária para a aplicação do método.

Foram identificados treze processos de negócio entre os quais a equipa teria que escolher três, com base em necessidades de melhoria comunicadas pela provedora da organização. Assim sendo, o sistema para o qual a equipa teria que modelar os requisitos suportaria os seguintes processos de negócio: Admissão de Utentes, Recursos Humanos e Compras. Seria através dos modelos de casos de uso com os requisitos deste sistema que se derivaria uma estimativa do esforço de desenvolvimento. No que concerne à identificação dos Casos de Uso, considerando a imaturidade da equipa, foi dada uma linha de orientação baseada no conceito de processo elementar de negócio segundo [Larman 2005]. Um processo elementar de negócio consiste numa tarefa desempenhada por uma ou algumas pessoas (não mais que duas ou três) que adiciona valor mensurável ou observável ao negócio; isto é, que (i) contribui para a concretização de objectivos da organização ou no contexto em que a tarefa se enquadra (ex: processo de negócio); (ii) é realizada numa única sessão, ou seja, de uma só vez e não demora dias a realizar; (iii) os dados resultantes apresentam-se num estado coerente. Considerando que todos os elementos de grupo fariam modelação de requisitos do sistema, foi fornecido um *template* para as descrições dos casos de uso.

A tabela 7 apresenta todos os actores e casos de uso identificados e associados aos módulos do sistema que suportariam os processos de negócio da organização. Assumiu-se que os actores identificados interagem com o sistema através de uma interface gráfica pelo que são todos categorizados como complexos. O número de transacções por cada caso de uso é também apresentado de modo a visualizar a sua complexidade.

Caso de Uso	Processo	Transacções	Complexidade	Actores
1. Registar pedido de admissão	Admissão de Utentes	10	Complexo	Func.
2. Actualizar pedido de admissão	Admissão de Utentes	7	Complexo	Func.
3. Validar pedido de admissão	Admissão de Utentes	6	Mediano	Gestor Ad.
4. Validar condição médica	Admissão de Utentes	3	Simples	Médico
5. Assinar contrato	Admissão de Utentes	3	Simples	Func.
6. Criar Mapa de Férias	Recursos Humanos	11	Complexo	Resp. RH
7. Planear Acções de formação	Recursos Humanos	9	Complexo	Gest. RH
8. Avaliar competências dos funcionários	Recursos Humanos	9	Complexo	Resp. RH
9. Gerir mapa de Vagas	Recursos Humanos	10	Complexo	Gest. RH
10. Receber candidaturas	Recursos Humanos	10	Complexo	Gest. RH
11. Realizar contratação	Recursos Humanos	10	Complexo	Gest. RH
12. Atribuir funções	Recursos Humanos	6	Mediano	Gest. RH
13. Emitir folha de pagamento	Recursos Humanos	7	Complexo	Resp. RH
14. Classificar Produto	Compras	3	Simples	Func. Arm
15. Comprar Produto	Compras	9	Complexo	Func. Arm
16. Realizar Inventário	Compras	3	Simples	Func. Arm
17. Validar pedido de compra	Compras	4	Mediano	Gestor Compras
18. Visualizar stock	Compras	3	Simples	Func. Arm

Tabela 7 - Casos de Uso e actores identificados

A avaliação dos factores ambientais e técnicos no projecto é apresentada nas tabelas 8 e 9.

Factor	Descrição	Peso	Importância	Total
E1	Familiarização com um processo de desenvolvimento	1,5	5	7,5
E2	Experiência com a aplicação	0,5	0	0
E3	Experiência com programação orientada a objectos	1	5	5
E4	Capacidades do analista	0,5	5	2,5
E5	Motivação	1	5	5
E6	Alterações nos requisitos	2	3	6
E7	Elementos em tempo parcial	-1	0	0
E8	Complexidade das ling. de programação	-1	3	-3
EFactor				23

Tabela 8 - Importância dos factores ambientais no projecto

Os factores considerados como cruciais para o sucesso do projecto foram avaliados com o peso máximo. Os factores que apresentam algum impacto no projecto (mas sem excesso) foram todos

ponderados com o valor 3 e os restantes factores considerados irrelevantes (ou que não se adequam ao contexto do projecto) foram ponderados com o valor 0.

Factor	Descrição	Peso	Importância	Total
T1	Sistema distribuído	2	0	0
T2	Performance	2	3	6
T3	Eficiência	1	3	3
T4	Complexidade de processamento	1	5	5
T5	Reutilização de código	1	3	3
T6	Instalação	0,5	3	1,5
T7	Usabilidade	0,5	5	2,5
T8	Portabilidade	2	3	6
T9	Manutenção	1	5	5
T10	Concorrência	1	3	3
T11	Segurança	1	5	5
T12	Acessibilidade de terceiros	1	3	3
T13	Formação	1	3	3
TFactor				46

Tabela 9 - Importância dos factores técnicos no projecto

A tabela 10 apresenta os resultados das variáveis calculadas e do número total de *Use Case Points*. Assim sendo, o número de *Use Case Points* no sistema multiplicados pelas 20 horas de esforço, fornecem a estimativa do esforço para desenvolver este sistema de aproximadamente 3 402 horas.

Métrica	Resultado
UAW	21
UUCW	205
UUCP	226
TCF	1,06
EF	0,71
UCP	≈ 170

Tabela 10 - Resultados das métricas

4. Conclusões

A partir dos modelos de casos de uso que capturaram os requisitos do sistema de software para suportar três processos de negócio de uma IPSS foi aplicado o método *Use Case Points* para fornecer uma estimativa do esforço do projecto. Apesar da veracidade dos resultados obtidos só poder ser comprovada em contexto de execução de projecto, é possível tecer algumas considerações sobre alguns factores que têm impacto na estimativa do esforço de desenvolvimento. O primeiro factor está relacionado com o momento onde se situa o projecto no contexto da unidade curricular de Processos e Metodologias de Software. Considerando que o projecto estava enquadrado na fase *Inception* do *Rational Unified Process*, apenas uma pequena percentagem de análise de requisitos foi realizada, pelo que a estimativa do esforço do projecto não poderá ser muito rigorosa. A imaturidade natural da equipa de trabalho no que toca à identificação e escrita de casos de uso é também dos factores que contribui para a imprecisão da estimativa. Notou-se uma grande preocupação em cumprir a notação dos diagramas de casos de uso em detrimento da sua especificação detalhada em texto. Assim sendo, este factor também teve impacto na estimativa considerando a natureza do próprio método, visto que um dos passos consiste em contabilizar o número de transacções nos fluxos principais de sucesso e fluxos alternativos. Apesar das informações fornecidas pela provedora da Instituição, a equipa sentiu uma grande falta de informação para realizarem a modelação de casos de uso pelo que esta actividade foi realizada com base em várias suposições sobre o funcionamento da organização.

Como trabalho futuro, um segundo estudo será realizado na segunda fase do projecto acima descrito, no âmbito da disciplina de Desenvolvimento de Aplicações Informáticas do 2º ano da Licenciatura em Tecnologias e Sistemas de Informação da Universidade do Minho. Pretende-se determinar a produtividade das equipas de desenvolvimento por *UCP* com a finalidade de calibrar estimativas futuras.

5. Referências

Abran, Alain, Software Metrics and Software Metrology, IEEE Computer Society & Wiley & Sons, 2010.

Albretch, A.J. (1979) "Measuring Application Development Productivity", Proceeding of Joint Share, Guide and IBM Application Development Symposium, October 1997, pp. 83-92.

Anda, B, Dreiem, H, Sjoberg, D, Jorgesen, M, Estimating Development Effort based on Use Cases – Experiences from Industry. In M. Gogolla, C. Kobryn (Eds.): UML 2001 – The Unified Modeling Language, Modeling Languages, Concepts, and Tools, 4th International Conference, Toronto, Canada, October 1-5, 2001, LNCS 2185, Springer-Verlag, pp. 487-502.

Cockburn, A, Writing Effective Use Cases, Reading, MA, Addison-Wesley, 2001.

Jacobson I., Christerson M., Jonsson P. and Övergaard G. (1992). Object-Oriented Engineering: Addison-Wesley.

Karner, G, Use Case Points: resource estimation for Objectory projects, Objective SF AB (copyright owned by Rational/IBM), 1993.

Larman, Craig, Applying UML and Patterns – An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3th Edition, Prentice Hall, 2005.

Nageswaran, Suresh, Test Effort Estimation Using Use Case Points, Quality Week 2001, San Francisco, California, USA, June 2001.