SDDI 2006

António Nestor Ribeiro

Integração de Técnicas Formais e Informais
na Especificação e Desenvolvimento de Sistemas de Grande Escala

Autor: António Manuel Nestor Ribeiro (anr@di.uminho.pt)
Tema: *Integração de Técnicas Formais e Informais na Especificação e Desenvolvimento de Sistemas de Grande Escala*
Orientador: Fernando Mário Junqueira Martins, Departamento de Informática, Universidade do Minho (fmm@di.uminho.pt)

Resumo Alargado

O UML como linguagem de modelação tem vindo a crescer em importância e utilização, assumindo-se como uma ferramenta importante na análise, modelação e especificação de sistemas de software. Apesar do seu interesse intrínseco e das sucessivas alterações e evoluções que tem experimentado, não endereça completamente alguns aspectos importantes na concepção de sistemas de software complexos. A própria definição do UML caracteriza-o como uma linguagem de modelação que aborda diversas vertentes da modelação e análise, recorrendo a notações diferentes para retratar os aspectos dinâmicos e estruturais de um sistema software.

Uma das ferramentas que o UML disponibiliza é o diagrama de Use Cases que assume importância na medida em que o processo de modelação e especificação de software deve começar por ele. No entanto aos diagramas de Use Case falta a capacidade de demonstrarem uma fundação semântica formalmente bem definida. Tal reflecte-se principalmente em especificações complexas envolvendo noções de concorrência, tempo e múltipla cardinalidade dos actores existentes. Por outro lado não é claro do ponto de vista dos diagramas de Use Case quais as condicionantes que permitem, ou não, a execução de um determinado caso.

Esta dissertação tem como principal objectivo o estabelecimento de, ao nível do diagrama de Use Cases, uma extensão ao actual modelo dos Use Cases, por forma a incorporar no cenário a capacidade de raciocínio com semântica bem-definida e com capacidade de prova e prototipagem. Com a inclusão deste nível de raciocínio será possível tornar a fase de recolha de requisitos mais iterativa e mais bem fundada.

Existem alternativas diversas que tem procurado colmatar a falha identificada. Podemos dividir essas alternativas em três grandes domínios: uma abordagem tipicamente assente em métodos formais, uma abordagem de construção e definição semântica do meta-modelo do UML e uma abordagem que privilegia a construção de novos conceitos diagramáticos e estereótipos a acrescentar aos já existentes.

A primeira abordagem é uma abordagem baseada na utilização de métodos e notações formais para descrever numa linguagem adequada os conceitos que são expressos nos diagramas UML. Esta abordagem permite trabalho a nível da linguagem utilizada e respectiva prototipagem e prova, mas tem como principal inconveniente o facto de produzir um modelo paralelo ao modelo UML. Além deste facto, não é linear provar que o modelo formal criado seja rigorosamente representativo do diagrama e que todas as assumpções existentes no diagrama tenham sido correctamente passadas para o modelo formal.

Por forma a conseguir retirar ambiguidade a algumas construções em UML, existem trabalhos cujo esforço se centra em fornecer uma semântica formal para os construtores gráficos de modelação existentes. Esta abordagem é feita essencialmente ao nível do meta-modelo do UML e procura verificar se todos os conceitos e construtores existentes nos diversos diagramas tem uma semântica perfeitamente estabelecida, por forma a capturar a sua capacidade de representação. Esta abordagem vai no sentido de permitir analisar a informação nos diagramas resultantes de um processo de modelação e construir um modelo mental e de código perfeitamente válido e não-ambíguo.

Por fim, uma terceira abordagem que é utilizada em trabalhos semelhantes propõe para a falta de semântica nos construtores dos Use Cases a extensão por incorporação de novos elementos diagramáticos e novos padrões de representação. Esta estratégia tem sido utilizada com algum relevo na modelação de sistemas multi-agente e em sistemas em que é fundamental a descrição de protocolos de comunicação entre entidades. Esta extensão assume assim a forma de utilização conjunta de diversos diagramas embora não se preocupe com a semântica implícita nessas construções.

A alternativa proposta nesta dissertação apresenta como principais características as seguintes linhas de acção: uma abordagem totalmente baseada em construtores UML estabelecidos, utilização de construtores formais escritos na linguagem de suporte OCL com o intuito de suportar formalmente a descrição dos Use Cases, incorporação ao nível do diagrama de Use Cases de aspectos comportamentais por forma a criar um único domínio semântico e uma plataforma de prototipagem de alto nível de Use

Cases possibilitando a validação dos diagramas o mais cedo possível no processo de modelação.

Estas contribuições permitem criar um nível de racíocinio mais preciso e endereçar problemas típicos de modelação de sistemas software complexos. Apresentam como vantagem o facto de ser métodos que se albergam na plataforma UML definida, não criando nenhuma rotura por acréscimo de novos componentes nem pelo facto de se desenvolverem modelos paralelos noutras linguagens ou paradigmas. O objectivo desta dissertação consiste em acrescentar metodologicamente ao UML a capacidade de oferecer suporte para as falhas identificadas atráves da inclusão de métodos de construção da modelação e técnicas formais bem conhecidas e suportadas. Essa inclusão possibilitará também a prototipagem dos modelos descritos em UML numa arquitectura configurável de objectos activos.

Objectivos

O objectivo mais lato da dissertação consiste em providenciar à linguagem de modelação UML - e ao processo de construção de modelações - a capacidade de desenvolver modelos de sistemas de software intrinsicamente complexos que sejam formalmente válidos e que permitam o raciocínio e prototipagem ao nível da camada de Use Cases. O trabalho efectuado procurou que essa formulação fosse feita recorrendo às técnicas disponibilizadas pela *framework* do UML por forma a que seja facilmente utilizada nas diversas ferramentas que suportam o processo de modelação. Numa perspectiva mais fina os objectivos podem ser descritos da seguinte forma:

- incorporação de uma abordagem formal no processo de modelação em UML - através da sobre-especificação dos diagramas de Use Cases recorrendo a expressões escritas em OCL;
- aumento do nível de expressividade da modelação com Use Cases - complementando a informação diagramática com outra mais formal que permite, durante a execução desta fase da tarefa de modelação, pensar na orquestração das entidades e no fluxo de informação. Durante a fase de construção do diagrama de Use Cases à medida que se identificam os actores e as tarefas é possível enriquecer o modelo com expressões que permitem especificar o ciclo de vida das entidades e alterações ao seu estado interno;
- adicionar de expressividade relativa a aspecto complexos de modelação - acrescentando informação formal aos diagramas e combinando diagramas de comportamento para complementar as descrições efectuadas nos diagramas de Use Case;
- permitir raciocinar a nível formal sobre os traços de comportamento gerados a partir dos Use Cases - possibilita-se assim um nível de detalhe sobre o modelo geral, ao introduzir na fase de análise elementos necessários mais tarde numa fase de definição de arquitectura. O incremento de informação produzido a este nível permite também verificar os caminhos possíveis (traços de execução) para as diversas funcionalidades do sistema que está a ser modelado;
- elaboração de uma camada de prototipagem para a nova definição do diagrama de Use Cases - através de um nível de execução das entidades permite-se validar a nível dos Use Cases a modelação efectuada. Esta camada tem como objectivo a prototipagem de modelação de tarefas (por exemplo da camada interactiva).

Contribuições

Principais contribuições técnico-científicas

As contribuições resultantes desta dissertação encontram-se na criação de um nível de racíocinio mais preciso na modelação de sistemas software complexos. Pretende-se apresentar um conjunto de técnicas e métodos que proporcionem ao engenheiro de software a capacidade de utilização da framework UML recorrendo a construtores com semântica precisa e não ambígua. Para tal incorpora um nível de raciocício formal na descrição da fase de análise, nomeadamente ao nível do diagrama de Use Cases.

De forma macro apresenta-se como vantagem o facto de criar métodos e ferramentas que se albergam na linguagem UML definida, não criando nenhuma rotura por acréscimo de novos componentes nem pelo facto de se desenvolverem modelos paralelos noutras linguagens ou paradigmas. Recorre-se a linguagens

existentes no UML para especificar os diagramas que são tipicamente mais informais e conjugar vários diagramas por forma a capturar todos os requisitos do sistema software, que de outra forma não seria identificados na fase de análise e modelação.

Em resumo, neste trabalho apresenta-se uma metodologia de construção de modelos de sistemas software complexos. O trabalho foca as questões associadas à análise e modelação de sistemas de software, nomeadamente no que respeita à fase de análise, recolha de requisitos, validação dos requisitos e consequente modelação. As contribuições do trabalho são genericamente endereçados por três grandes linhas de acção: (i) o reforço da componente diagramática dos Use Cases por inclusão de formalização, (ii) a construção de uma abordagem unificada com recurso a várias vistas do mesmo sistema, como mecanismos de consolidação semântica da modelação e por fim (iii) uma preocupação em operacionalizar o processo através de recurso a prototipagem.

Apresenta-se de seguida a enumeração das principais contribuições para a área de I&D de Engenharia de Software, resultantes deste trabalho:

- Extensão ao diagrama dos Use Cases por enriquecimento de informação - permite associar à fase de análise mais desprovida de fundamentação, métodos e constutores que possibilitam que estes diagramas sejam mais fiáveis, correctos e com semântica mais bem definida. Possibilita-se que possam ser utilizados como ferramenta de análise para a descrição de problemas complexos, o que é reconhecidamente uma limitação da utilização actual dos diagramas de Use Case;

- Uso de métodos formais para complementar os modelos UML - permite acrescentar metodologicamente informação aos elementos gráficos constantes de um diagrama de Use Case por forma a descrever o fluxo de informação entre os diversos casos e os actores envolvidos. Combinação de Use Cases com a linguagem de restrições OCL por forma a descrever as pré e pós-condições, bem como as alterações ao estado interno, conjugado com os diagramas de comportamento gerados permite que se endereçe logo a fase de modelação;

- possibilidade de prova e raciocínio sobre os traços gerados a partir de um Use Case - acrescenta ao modelo a capacidade de raciocinar sobre as alterações de estado das entidades e formalmente provar que o que foi modelado corresponde ao que realmente se quer fazer. Acrescenta um nível de prova formal que não poderia ser feito no actual processo de utilização do UML, no qual apenas é possível testar o sistema depois de implementado;

- camada de prototipagem rápida para descrições de Use Case - permite verificar após a tarefa de construção do Use Case se o que foi especificado corresponde ao pretendido. Permite também comparar com a informação resultante da especificação usando notação formal no que respeita à preservação dos invariantes de estado e pré e pós-condições.

Enquadramento

Enquadramento Científico

A Engenharia de Software tem como objectivo agregar uma série de métodos e processos que permitem coordenar o efectivo desenvolvimento de sistemas de software. Esses métodos e processos incutem no desenvolvimento de software rigor e capacidade de pensar arquitecturalmente os problemas e respectivas soluções antes de passar à fase de desenvolvimento da solução. Nesse âmbito são endereçados muitos domínios entre os quais está a recolha de requisitos do utilizador - tanto funcionais como não-funcionais - e a captura dessa informação numa representação conhecida e com semântica bem definida.

Essa necessidade é tanto maior quanto maior seja a complexidade do sistema de software que se queira modelar. A recolha de requisitos do utilizador e a sua representação em diagramas de Use Case não pode padecer de problemas tipícos de sub-especificação de complexidade. É necessário que nesta fase se identifiquem todas as características do problema, sob pena de as ferramentas utilizadas depois no processo se basearem em informação errada e modelarem estruturalmente conceitos não completamente especificados. Acresce a isto, o facto de os diagramas de comportamento serem feitos numa base em que o elemento chave é uma entidade, por decomposição do problema segundo a estratégia normalmente seguida no paradigma dos objectos, o que dificulta a descoberta de sub-especificação a nível da recolha de requisitos.

Os objectivos macro da dissertação consistem em providenciar um modelo de raciocínio e de processo que permitam melhorar a utilização do UML na modelação e especificação de sistemas software complexos, ao acrescentar informação aos diagramas de Use Case e providenciar um nível de prototipagem dos mesmos. Estes contributos podem ser aproveitados em projectos de I&D em que a componente de Engenharia de Software e modelação de sistemas de software complexos (concorrentes, tempo-real, distribuídos, interacção e fluxo de informação não-trivial, sistemas interactivos, etc.) estejam presentes.

Motivação

A motivação para o trabalho advém do facto de a linguagem de modelação commumente utilizada, o UML, não ter o mesmo nível de detalhe em todas as suas componentes. Tal disparidade de exaustividade na descrição das diferentes componentes de um sistema de software leva a problemas naturais de interpretação e de posterior gestão do processo de desenvolvimento. Acreditando que o UML possa ser utilizado para mais do que a geração de documentação e possa ser visto como uma ferramenta de modelação e mesmo incluída no processo de desenvolvimento, é necessário acrescentar mais informação e tornar mais robusta a linguagem e a sua utilização. A falta de disciplina na utilização de alguns diagramas leva a que possam surgir problemas de interpretação de um modelo, pelo que se torna interessante complementar a linguagem com mecanismos que a robusteçam e que diminuam a possibilidade de não captura total de todos as particularidades do sistema a especificar. A necessidade de evitar que ao nível dos Use Cases a compreensão dos diagramas seja mais aparente que real e que a interpretação de um mesmo diagrama por pessoas diferentes possa levar a modelações distintas e implementações diferentes é a principal motivação do trabalho.

Este trabalho pretende retirar um nível de raciocínio *fuzzy* às descrições criadas pela conjugação com expressões escritas em OCL e que estabelecem os invariantes a respeitar, bem como pela conjugação de outros tipos de diagramas na descrição dos Use Cases. Além desta motivação estruturante no que respeita às boas práticas de construção de sistemas software, acresce o facto de ser pouco evidente a tarefa de prototipagem dos Use Cases, sendo que eles representam já um nível de entendimento do problema que permite que possa ser validado pelo cliente. E tendo em consideração o incremento de informação que se coloca nos Use Cases é possível efectuar a sua prototipagem, o que é importante na concepção dos sistemas de software (e mesmo indispensável por exemplo em sistemas interactivos).

Objectivos Detalhados

Resposta na anterior secção relativa a objectivos.

Trabalhos Alternativos

Tal como referido anteriormente existem com o mesmo objectivo final várias abordagens distintas no caminho que trilham para chegar ao mesmo objectivo: o enriquecimento do UML com uma semântica mais bem definida. Essas abordagens podem ser caracterizadas em três famílias: uma (i) abordagem de construção de métodos e linguagens formais, uma (ii) abordagem orientada à fundamentação e raciocínio ao nível do meta-modelo UML e por último (iii) uma abordagem em que são criados novos construtores e diagramas.

Enumeram-se de seguida alguns dos trabalhos existentes nestas três vertentes de forma muito sintética, apenas para conhecimento do espaço de intervenção.

Em [21] é utilizada a notação formal Z para providenciar uma representação do modelo dos Use Cases por forma a aplicá-la a vários exemplos de casos de uso. Existe um focus na operacionalização ao permitir a utilização de Executable Z para testar os modelos criados. Neste trabalho endereçam-se problemas decorrentes da introdução de concorrência nos Use Cases e as sequências de acções são agregadas em diálogos, permitindo depois testes sobre os traços do diálogo. Em [26] é abordada uma estratégia semelhante com o uso de Object-Z, na medida em que a partir dos diagramas de Use Case são gerados modelos na linguagem formal e é sobre esses modelos que o raciocínio e execução é efectuada. No trabalho de [15], a mesma raíz do problema é abordada de forma semelhante, sendo que o foco é dado aos diagramas de actividade que podem ser criados a partir dos diagramas de Use Case. Neste trabalho a contribuição é muito mais focada na componente de *workflow* que é de difícil representação ao nível dos Use Cases. É utilizado o model checking para a verificação de requisitos funcionais em especificações

de workflow. A temática do trabalho reside na definição de uma semântica formal para os diagramas de actividade que são escritos em complemento aos de Use Case para descrever as tarefas de workflow, sendo que questões como paralelismo e concorrência são naturalmente endereçadas, não conseguindo ser expressas a nível dos diagramas de Use Case.

Uma modelação completa em termos de utilização do Z foi feita por [18], na qual são utilizados os diagramas de Classe e de Use Case e é feita a sua transformação em especificações em Z. A análise dos modelos gerados em Z e a própria transformação dos modelos permite detectar as inconsistências e ambiguidades na modelação feita a partir dos requisitos fornecidos.

A problemática da consistência de informação entre os vários níveis de diagramas que se podem retirar dos diagramas de Use Case é abordada também pelos trabalhos de [30] e [29]. Essa preocupação é também evidente no trabalho de [34] na preocupação que expressa na consistência do modelo e na parte comportamental na qual existe muito trabalho a ser feito. Nesta dissertação a consistência do modelo pretende ser assegurada desde a componente de diagramas de Use Case recorrendo a sobre-especificação dos mesmos e a complementariedade com diagramas de comportamento. Os trabalhos de [30] e [34] são a esse nível importantes referências para validar a consistência do modelo.

No que se refere à fundamentação a nível do meta-modelo do UML podem ser citados os trabalhos de [35] em que de forma consistente aborda a possibilidade de conjugar vários diagramas por forma a acrescentar expressividade e diminuir o risco de a modelação ficar afectada pelas insuficiências da notação. Refira-se também o trabalho em profundidade feito na resposta à especificação do UML 2.0, nomeadamente em relação ao OCL que pode ser encontrada em [8] e em documentação subjacente do mesmo grupo de trabalho [25].

Na vertente em que se acrescentam novos construtores diagramáticos e se altera o processo e a linguagem, existem vários trabalhos que abordam a problemática desta forma. Entre eles podem-se identificar os trabalhos da comunidade multi-agente [2, 28] que deram origem a uma linha de acção na especificação da versão 2 do UML, no âmbito da FIPA. Ainda nesta abordagem surgem extensões aos diagramas de Use Case por incorporação de outros conceitos semelhantes [1]. Neste trabalho procura-se diminuir a distância semântica entre os requisitos e os diagramas de Use Cases a eles associados, da restante modelação recorrendo a diagramas de comportamento.

Entre os trabalhos mais próximos do desta dissertação identificam-se alguns que abordaram o problema com a mesma intenção e que optaram por soluções semelhantes, mas não completamente concorrenciais. Alguns desses trabalhos abordam o problema para resolverem problemas de uma área específica, os sistemas interactivos, onde a ambiguidade de alguns diagramas de Use Case se fazem sentir. Os sistemas interactivos são inclusivé um bom caso de estudo na medida em que o nível de expressividade exigido é muito grande e a mesma modelação pode ser derivada de sistemas software distintos. Acresce a esta necessidade o facto de a prototipagem nos sistemas interactivos ser muito importante na validação dos requisitos. O trabalho de [27, 23] parte dos mesmos principíos que esta dissertação, ou seja, identifica a falta de suporte para a descrição de sistemas interactivos na linguagem UML e no processo a ela associado, propondo uma perfil para suprir estas lacunas. Esta falha é contornada pela incorporação de extensões baseadas num método de engenharia de software - o Wisdow. As extensões propostas permitem expressar a modelação de tarefas e possibilitam a descrição de padrões de interacção.

Trabalho semelhante é apresentado em [31, 9], ainda na área dos sistemas interactivos, em que se desenvolveu uma ferramenta - ARGOi - que importa modelos definidos em ArgoUML (que é baseada no UML padrão). Neste trabalho faz-se uma descrição exaustiva das lacunas do UML para a descrição de sistemas de software complexos, como sejam os sistemas interactivos, e incorporam-se técnicas baseadas em modelos de descrição de interfaces com o utilizador. Consegue-se assim uma ligação mais eficaz entre a interface com o utilizador com o sistema software em que ela assenta. O UMLi providencia uma notação diagramática aumentada para modelar interfaces com o utilizador e acrescenta também notação adicional aos diagramas de actividades por forma a descreverem comportamentos específicos dos sistemas interactivos. O trabalho fornece uma definição semântica bem definida com recurso ao uso da OCL e a ferramenta resultante é bastante interessante.

Um trabalho com objectivos finais concominantes é o apresentado em [14, 3] no qual o resultado da modelação recorrendo a diagramas de Use Case é analisado e considerado insuficiente. São estudados os defeitos e erros típicos na utilização de diagramas de Use Case e o facto de as capacidades de expressividade actuais destes influirem na qualidade do resultado obtido. Este trabalho foca-se no processo de descoberta de requisitos por forma a aplicar heurísticas no processo, por forma a aumentar a expressividade dos diagramas de Use Case obtidos.

Ainda dentro da problemática desta dissertação encontra-se o trabalho expresso em [20], pelo esforço que representa a construção de especificações UML precisas e não ambíguas em problemas típicos de concorrência. Este trabalho faz uso da OCL para provar que é possível com o recurso a uma linguagem de restrições preservar construtores típicos de concorrência e que é possível acrescentar aos modelos UML esta informação, combatendo a possível ambiguidade dos mesmos.

Bibliografia Principal

Referências

[1] Daniel Amyot and Gunter Mussbacher. On the extension of UML with use case maps concepts. In Andy Evans, Stuart Kent, and Bran Selic, editors, *UML 2000 - The Unified Modeling Language. Advancing the Standard. Third International Conference, York, UK, October 2000, Proceedings*, volume 1939, pages 16–31. Springer, 2000.

[2] Bernhard Bauer, Jörg P. Müller, and James Odell. *Agent-Oriented Software Engineering*, chapter Agent UML: A Formalism for Specifying Multiagent Interaction. Springer-Verlag, 2001.

[3] Beatriz Bernardez, Amadore Duran, and Marcela Genero. Empirical evaluation and review of a metrics-based approach for use case verification. *Journal of Research and Practice in Information Technology*, 36(4), 2004.

[4] Grady Booch, James Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language - User Guide.* Addison-Wesley, 1998.

[5] Tony Clark, Andy Evans, and Stuart Kent. Engineering modelling languages: A precise meta-modelling approach. Technical report, 2001.

[6] Alistair Cockburn. *Writing Effective Use Cases.* Addison-Wesley, 2001.

[7] 2U Consortium. Response to the uml 2.0 ocl rfp. Technical report, March 2002.

[8] 2U Consortium. Unanbiguous uml (2u) 2nd revised submission to uml 2 infrastructure rfp. Technical report, 2002.

[9] Paulo Pinheiro da silva. *Object Modelling of Interactive Systems: The UMLi approach.* PhD thesis, Department of Computer Science - University of Manchester, 2002.

[10] Alioune Diagne, Jean-Michel Ilié, and Daniel Moldt. Integration of Semi-Formal and Formal Approaches. In *ECOOP'97 Workshop on Models, Formalisms and Methods for Distributed Object-Oriented Concurrent Systems.* Springer-Verlag, 1997.

[11] Bruce Powel Douglass. *Real Time UML - Developing Efficient Objects for Embedded Systems.* Addison-Wesley, 1998.

[12] Desmond d'Souza and Alan Wills. *Objects, Components, and Frameworks with UML.* Addison-Wesley, 1999.

[13] Sophie Dupuy-Chessa and Lydie du Bousquet. Validation of uml models thanks to z and lustre. In *FME 2001.*

[14] A. Duran, B. Bernardéz, M. Genero, and M. Piattini. Empirically driven use case metamodel evolution. In "Thomas Baar, Alfred Strohmeier, Ana Moreira, and Stephen J. Mellor", editors, *UML 2004 - The Unified Modeling Language. Model Languages and Applications. 7th International Conference, Lisbon, Portugal, October 11-15, 2004, Proceedings*, volume 3273 of *LNCS*, pages 1–11. Springer, 2004.

[15] H. Eshuis. Semantics and verification of uml activity diagrams for workflow modelling, 2002.

[16] A. Evans, R. France, K. Lano, and B. Rumpe. The uml as a formal modelling notation. In *UML'98 - Beyond the notation.*

[17] Andy Evans, Stuart Kent, and Bran Selic, editors. *UML 2000 - The Unified Modeling Language, Advancing the Standard, Third International Conference, York, UK, October 2-6, 2000, Proceedings*, volume 1939 of *Lecture Notes in Computer Science*. Springer, 2000.

[18] Robert France, Emanuel Grant, and Jean-Michel Bruel. Umltranz: An uml-based rigorous requirements modeling technique. Technical report, Colorado State University, 2000.

[19] Hassan Gomaa. *Designing Concurrent, Distributed, and Real-Time Applications with UML*. Addison-Wesley, 2000.

[20] Agustín Goñi and Yadran Eterovic. Building precise uml constructs to model concurrency using ocl. In "Thomas Baar, Alfred Strohmeier, Ana Moreira, and Stephen J. Mellor", editors, *UML 2004 - The Unified Modeling Language. Model Languages and Applications. 7th International Conference, Lisbon, Portugal, October 11-15, 2004, Proceedings*, pages 212–225. Springer-Verlag, 2004.

[21] Wolfgang Grieskamp and Markus Lepper. Using use cases in executable z. In *ICFEM*, pages 111–120, 2000.

[22] Ivar Jacobson. *Object-Oriented Software Engineering - a Use Case Driven Approach*. Addison-Wesley, 1992.

[23] Nuno Jardim Nunes and João Falcão e Cunha. Wisdom — A UML based architecture for interactive systems. *Lecture Notes in Computer Science*, 1946, 2001.

[24] Anneke Kleppe and Jos Warmer. Unification of static and dynamic semantics of uml. Technical report, 2001.

[25] Girish Maskeri, James Willans, Tony Clark, Andy Evans, Stuart Kent, and Paul Sammut. A pattern based approach to defining translations between languages. In *4 th Workshop on Rigorous Object Oriented Methods*.

[26] Ana Moreira and João Araújo. Generating object-z specifications from use cases. In Joaquim Filipe, editor, *Enterprise Information Systems*, pages 43–50. Kluwer Academic Publishers, 1999.

[27] Nuno Jardim Nunes and João Falcão e Cunha. Towards a uml profile for interaction design: the wisdom approach. In *UML*, pages 101–116, 2000.

[28] James Odell, H. Van Dyke Parunak, and Bernhard Bauer. Extending uml for agents. In *AOIS Workshop - AAAI*.

[29] Gunnar Övergaard and Karin Palmkvist. A formal approach to use cases and their relationships. In Jean Bézivin and Pierre-Alain Muller, editors, *The Unified Modeling Language, UML'98 - Beyond the Notation. First International Workshop, Mulhouse, France, June 1998*, pages 309–317. Springer-Verlag, 1998.

[30] R. Paige, J. Ostroff, and P. Brooke. Checking the consistency of collaboration and class diagrams using pvs, 2002.

[31] Paulo Pinheiro da Silva and Norman W. Paton. UMLi: The unified modeling language for interactive applications. In Andy Evans, Stuart Kent, and Bran Selic, editors, *UML 2000 - The Unified Modeling Language. Advancing the Standard. Third International Conference, York, UK, October 2000, Proceedings*, volume 1939, pages 117–132. Springer, 2000.

[32] James Rumbaugh, M. Blaha, W. Premerlani, and F. Eddy. *Object-Oriented Modelling and Design*. Prentice-Hall, 1991.

[33] James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1998.

[34] Ragnhild Van Der Straeten, Viviane Jonckers, and Tom Mens. Supporting model refactorings through behaviour inheritance consistencies. In Thomas Baar, Alfred Strohmeier, Ana Moreira, and Stephen J. Mellor, editors, *UML 2004 - The Unified Modeling Language. Model Languages and Applications. 7th International Conference, Lisbon, Portugal, October 11-15, 2004, Proceedings*, volume 3273 of *LNCS*, pages 304–319. Springer, 2004.

[35] Alain Wegmann and Guy Genilloud. The roles of "roles" in use case diagrams. In Andy Evans, Stuart Kent, and Bran Selic, editors, *UML 2000 - The Unified Modeling Language. Advancing the Standard. Third International Conference, York, UK, October 2000, Proceedings*, volume 1939, pages 210–224. Springer, 2000.

Avaliação

Em comparação com os trabalhos alternativos e concorrenciais apresentados em secção anterior a grande diferença entre esta dissertação e esses trabalhos prende-se com a abordagem seguida e com os objectivos a que se propuseram. Como referido em secção anterior, este trabalho procura estabelecer dentro da framework UML os mecanismos de raciocínio que permitem acrescentar a capacidade de aumentar a expressividade dos diagramas de Use Case. Neste trabalho existiu a preocupação em dotar o processo proposto de uma camada de operacionalização como mecanismo de enriquecimento da componente dos Use Cases e permitir que uma parte substancial da análise e descoberta de requisitos se faça a este nível.

Os trabalhos apresentados em secção anterior reconhecem o mesmo problema, mas abordam-no de forma diferente. Os trabalhos alternativos mais consistentes são aqueles em que existe uma formalização dos Use Case recorrendo a métodos formais estabelecidos. A abordagem seguida nesta dissertação não foi essa, pelo facto de criar dois níveis de interpretação do problema, o existente em UML e o criado na notação formal. Dessa forma o esforço de gestão de duas representações em paralelo aumenta, tornando mais complexo o refinamento por iteração da modelação. Acresce também o facto de não ser sempre completamente evidente que a formalização produzida corresponda ao que se pretende de facto modelar, por duas ordens de razão: discrepância entre o modelo correcto e a sua formalização, ou má construção do modelo formal por ambiguidade na modelação em UML.

Os trabalhos que envolvem a fundamentação ao nível do meta-modelo do UML, embora importantes não permitem resolver os problemas a que nos propomos nesta dissertação, pelo facto de a identificação do problema estar na fraca capacidade digramática de alguns modelos. É claro que uma semântica correcta e precisa ajuda a compreender melhor os modelos desenhados, mas tal não é por si só suficiente.

O trabalho de [9] parte dos mesmos princípios que esta dissertação, mas é particularmente dirigido às tarefas de construção de interfaces com o utilizador, daí ter introduzido novos construtores por forma a sistematizar tarefas identificadas, e introduz novos diagramas que sintetizam as tarefas de construção da camada interactiva. O trabalho de [9] é realizado com o intuito de simplificar a escritas de modelos em UML ao acrescentar construtores mais complexos, mas que simplifiquem o trabalho de quem modela a componente da interface com o utilizador. O trabalho desta dissertação assume uma posição de maior abrangência do que o trabalho de [9], na medida em que não se destina exclusivamente à construção de interfaces com o utilizador, embora também tenha como objectivo a modelação de sistemas interactivos.

Devido também a esta diferença de abordagem os resultados desta dissertação não são tão eficazes na modelação de sistemas interactivos, tanto pela diferença de abordagem quer pela criação de uma ferramenta assistida para a modelação. Atingiu-se um grau de automatismo na modelação da camada interactiva que esta dissertação não apresenta, também pelo facto de não pretender oferecer uma ferramenta específica e com construtores novos, mas um método para incremento da utilidade dos Use Cases e um mecanismo de operacionalização. Esta dissertação tem como objectivo fazer a integração das técnicas formais nas técnicas informais daí dar mais importância à integração aumentada da OCL na descrição dos Use Cases e na composição de diagramas como mecanismo de aumento de semântica.

Como comentário geral ao trabalho realizado pode-se referir a abrangência da utilização do método proposto, sendo que uma outra metodologia de abordagem do trabalho poderia ter sido utilizada - recorrendo a uma filosofia mais bottom-up e com uma generalização posterior. Teria sido possível começar o trabalho dividindo o espectro em sistemas software tipo (sistemas interactivos, sistemas concorrentes, etc.) e em face dos resultados de cada trabalho reunir os contributos e efectuar um exercício de generalização. A divisão por famílias de sistemas software é também a melhor estratégia para a disseminação dos resultados.

A estrutura da tese é também dividida de forma táctica em função dos problemas que se pretendem resolver, por forma a ser mais facilmente compreensível qual a estratégia e quais as vantagens do método proposto. Abordam-se assim as pricipais lacunas do suporte actual e explicam-se e contextualizam-se as alternativas propostas.