• U C •
FCTUC FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

DEPARTAMENTO DE ENGENHARIA MECÂNICA

Desenvolvimento de uma plataforma interactiva para a utilização do DD3IMP

Dissertação apresentada para a obtenção do grau de Mestre em Engenharia Mecânica na Especialidade de Sistemas de Produção

Autor

Ricardo Miguel Santos Heleno

Orientadores

Marta Cristina Cardoso Oliveira

José Luís de Carvalho Martins Alves

Júri

Presidente	**Professor Doutor Cristóvão Silva** **Professor Auxiliar da Universidade de Coimbra**
Vogais	**Professor Doutor Luís Filipe Martins Menezes** **Professor Catedrático da Universidade de Coimbra**
	Mestre Diogo Mariano Simões Neto **Aluno de Doutoramento da Universidade de Coimbra**
	Professora Doutora Marta Cristina Cardoso de Oliveira **Professor Auxiliar da Universidade de Coimbra**
	Professor Doutor José Luís de Carvalho Martins Alves **Professor Auxiliar da Universidade do Minho**

Colaboração Institucional

Universidade do Minho
Escola de Engenharia
Centro de Tecnologias Mecânicas e de Materiais

Coimbra, Setembro, 2012

“It is not knowledge, but the act of learning, not possession but the act of getting there, which grants the greatest enjoyment.”

Johann Carl Friedrich Gauss, numa carta para Farkas Bolyai, 1808.

Aos meus Pais, ao meu Padrinho e à minha Namorada.

Agradecimentos

O trabalho que aqui se apresenta só foi possível graças à colaboração e apoio de algumas pessoas, às quais não posso deixar de prestar o meu reconhecimento.

À Professora Doutora Marta Cristina Cardoso de Oliveira, uma das pessoas de maior boa vontade que tive a oportunidade de conhecer até hoje, pela disponibilidade constante na orientação deste trabalho.

Ao Professor Doutor José Luís de Carvalho Martins Alves, pela orientação, pelos seus conselhos críticos que permitiram levar este trabalho a bom termo.

Ao Mestre Diogo Mariano Simões Neto, pelas críticas e sugestões, pela disponibilização para partilhar o seu conhecimento.

Aos meus Pais e o meu Padrinho, pelo apoio incondicional sem o qual não teria sido possível chegar aqui. A eles, a minha imensa gratidão.

À minha Namorada, pelo apoio, pelo encorajamento, pela paciência e em especial pela sua presença.

O presente trabalho foi realizado no âmbito do projeto designado por "*N2C: Tratamento contínuo do contacto com atrito com recurso à descritização com superfícies Nagata*" com a referência PTDC/EME-TME/103350/2008, co-financiado pela Fundação para a Ciência e Tecnologia através do projecto PTDC/EME-TME/103350/2008 e pelo FEDER, através do Programa Operacional Factores de Competitividade do QREN com referência COMPETE: FCOMP-01-0124-FEDER-010301.

Resumo

Este trabalho descreve a plataforma DD3LT que foi desenvolvida para apoiar a aprendizagem e o ensino de métodos de produção virtual de conformação de chapas metálicas, alertando para os problemas associados ao uso incorrecto dos códigos de análise por elementos finitos. Esta plataforma é suportada pelo programa DD3IMP, um *solver* de elementos finitos académico que tem vindo a ser continuamente desenvolvido e optimizado para a simulação de estampagem de chapas metálicas. A plataforma DD3LT integra uma aplicação interactiva que visa ajudar na tarefa de pré e pós processamento do modelo, e uma base de dados extensiva de problemas de estampagem de chapas metálicas. Esta plataforma tem como objectivo colmatar necessidades didácticas do ensino e da aprendizagem dos métodos de produção virtual associados às tecnologias de estampagem de chapas metálicas. A plataforma DD3LT permite o acesso fácil a uma panóplia de problemas de estampagem de chapas metálicas através de um interface de selecção interactivo, possibilitando alterar o problema seleccionado de uma forma intuitiva, num interface amigável. A selecção do problema baseia-se na selecção topológica do conjunto de ferramentas envolvidas no processo de estampagem pretendido. Após esta selecção são exibidas várias opções de problemas de estampagem já realizados com essas mesmas ferramentas, diferindo na discretização e no modelo constitutivo do material do corpo deformável. Uma vez isolado o problema pretendido, é possível alterar a discretização do corpo deformável, as leis constitutivas dos materiais, os parâmetros de processo, os parâmetros de simulação, as condições de contacto, em suma, todos os parâmetros do modelo (com a excepção da geometria das ferramentas), permitindo desta forma a sua redefinição. A plataforma DD3LT permite executar a simulação no DD3IMP, e acompanhar o decorrer da simulação. Por fim, a plataforma DD3LT disponibiliza algumas das variáveis do processo simulado (e.g. as forças exercidas pelas ferramentas em cada incremento), possibilitando uma primeira análise do problema simulado.

Palavras-chave: Método dos Elementos Finitos, Conformação de chapas metálicas, DD3IMP, Produção virtual, Ensino.

Abstract

This work describes the DD3LT platform developed to support the learning and teaching of virtual try-out of sheet metal forming processes, based on the awareness of the problems associated with the careless use of finite element analysis codes. This platform is supported on the DD3IMP code, which is an in-house FE solver that has been continuously developed and optimized to simulate sheet metal forming processes. The DD3LT platform integrates an interactive application in order to help the model pre and post processing, as well as an extensive database of sheet metal forming benchmark problems. This platform aims to fill educational needs of the learning and teaching of virtual try-out of sheet metal forming processes. The platform DD3LT allows an easy access to an extensive database of sheet metal forming benchmark problems via an interactive selection interface, enabling the modification of the selected model through a friendly user interface. The selection of the benchmark problem is based on the topological selection of the tools set involved in the desired stamping process. After the selection, several options are displayed differing in the discretization and in the constitutive model of the material of the deformable body. Once the desired benchmark problem is isolated, one can change the discretization of the deformable body, the constitutive laws of materials, the process parameters, the simulation parameters, the contact conditions, in short, all model parameters (with the exception of the tool geometry), thus allowing its redefinition. The DD3LT platform is able to launch the DD3IMP simulation, and of follow its course. Finally, the platform DD3LT provides some simulated process variables (e.g. forces exerted by tools at each increment), enabling a first analysis of the problem simulated.

Keywords: Finite Element Method, Sheet metal forming, DD3IMP, Virtual try-out, Teaching.

Índice

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

ÍNDICE DE QUADROS

1. INTRODUÇÃO

Actualmente, o Engenheiro Mecânico dispõe de poderosas ferramentas numéricas de apoio à decisão. De facto, a concepção virtual com recurso à análise pelo método dos elementos finitos (MEF) permite desenvolver e realizar estudos detalhados de inúmeros problemas em diferentes áreas da engenharia. Neste contexto, a análise pelo MEF de problemas de engenharia tornou-se uma ferramenta indispensável, uma vez que permite a optimização de parâmetros que ditam o sucesso de processos tecnológicos.

O processo de conformação de chapas metálicas é um dos processos tecnológicos para os quais a concepção virtual através do MEF adquiriu uma importância relevante nos últimos anos, conduzindo também a desenvolvimentos tecnológicos. A determinação da janela de processo envolve a análise de diversos parâmetros tais como a geometria das ferramentas, parâmetros de materiais, etc. Os métodos de produção virtual reduzem o tempo e custo de concepção e fabrico das ferramentas de estampagem. Por este motivo, a indústria de estampagem de chapas metálicas apoia-se no MEF para realizar o processo de concepção e fabrico de ferramentas, em particular, no caso de processos complexos envolvendo multi-etapas.

A estampagem de chapas metálicas é um processo no qual a geometria originalmente plana de uma chapa metálica é modificada para a forma desejada, através da aplicação de forças externas que provocam a deformação plástica do material. Este processo permite uma elevada cadência na produção de componentes de chapa metálicas de diferentes complexidades. Desta forma, devido às elevadas taxas de produção, uma das indústrias mais importantes na exploração das vantagens deste tipo de processo é a indústria automóvel, visando aumentar a competitividade. Nos últimos anos esta indústria tem vindo a ser continuamente estimulada por novas leis de segurança e ambientais, leis de conservação de energia e austeras exigências de desenvolvimento sustentável, conduzindo à necessidade de utilizar materiais com maior resistência específica. De facto, observa-se uma elevada exigência com vista à redução do peso dos veículos visando melhorar a eficiência do consumo de combustível, mantendo os requisitos de segurança através do melhoramento do desempenho em colisão. Neste contexto, os processos de estampagem

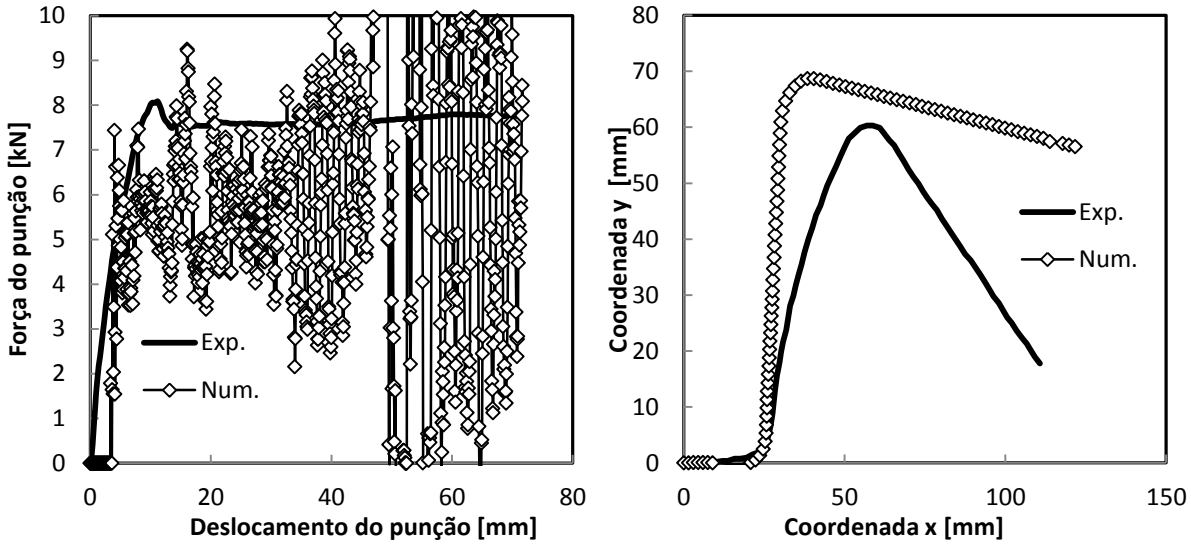
têm sido crescentemente utilizados em novos materiais, especialmente nesta indústria. De facto, novos materiais como aços de alta resistência e ligas de alumínio são cada vez mais procurados pela indústria automóvel, com vista à produção de componentes mais leves. Em particular, materiais avançados como os aços de alta resistência estão a ser a ser usados em mais de 60% das peças que constituem o corpo dos veículos modernos (Lingbeek, 2003). A complexidade crescente dos componentes obtidos por estampagem tem também contribuído para criar uma maior dependência da concepção virtual, em particular a simulação numérica de processos de conformação de chapa metálica com recurso ao MEF, estendendo-se o seu uso por toda a cadeia de produção (Roll e Rohleder, 2002). A simulação numérica permite a validação virtual de ferramentas de estampagem e de parâmetros de processo, conduzindo a uma diminuição relativa de tempo e de custos comparativamente aos testes experimentais. De facto, permite prever o escoamento do material, analisar o estado de tensão, deformação e distribuição de temperaturas, determinar as forças de conformação, identificar potenciais fontes de defeito e falha, aperfeiçoar a qualidade e a complexidade das peças e reduzir os custos de fabrico. Actualmente, num ambiente de produção integrado, a modelação e simulação de processos de fabrico são parte integrante do processo de concepção do produto (Tisza, 2004). Sumariamente, a simulação numérica pode ajudar na optimização de toda a cadeia produtiva, desde o material em bruto até ao produto final. Isto pode traduzir-se em enormes proveitos quer económicos, quer a nível de tempos de produção e desenvolvimentos tecnológicos, cruciais num mercado actual altamente competitivo.

Todos estes factores beneficiam da análise por elementos finitos que é extraordinariamente importante para a concepção e optimização do processo de estampagem de chapas metálicas. Tal abordagem de tentativa-erro virtual é consensualmente aceite como o principal factor que contribui para o enorme decréscimo no prazo de comercialização de novas peças estampadas, bem como para as notáveis reduções de custos, tempo e esforços no seu *design*, na concepção do processo e na sua produção. Actualmente, é possível encontrar no mercado diversos códigos comerciais especialmente desenvolvidos para a simulação numérica de processos de conformação largamente utilizados na indústria. O aumento da precisão dos resultados da simulação numérica e o aumento do poder computacional também contribuiu para o elevado interesse industrial no

desenvolvimento virtual de ferramentas de conformação, pois actualmente é possível analisar componentes e processos cada vez mais complexos.

Embora o potencial da utilização da simulação numérica na análise de processos tecnológicos seja óbvio, o uso descontrolado destas ferramentas é extremamente perigoso. O engenheiro CAE[1] deve estar consciente que todas as simulações numéricas são dependentes do modelo utilizado e, consequentemente, imperfeitas e de certa forma erradas. Por este motivo, a correcta interpretação dos resultados requer pessoal especializado com conhecimento detalhado do processo tecnológico bem como do código, tal como, dos métodos numéricos, dos parâmetros numéricos e algoritmos, da modelação do comportamento mecânico dos materiais, etc. De facto, existe um esforço contínuo em tornar o uso do MEF mais simples e interactivo. A indústria espera que a utilização deste tipo de programas seja tão simples que não exista a necessidade de contratar um especialista em elementos finitos. Para além disto, a ferramenta de simulação deve estar disponível onde é necessária, ou seja, deve ser passível de utilizar no gabinete de projecto e não só no departamento computacional. Desta forma, a facilidade de utilização e a amigabilidade são particularidades críticas para o sucesso do código de simulação na indústria (Tekkaya, 2000). Contudo, os riscos associados ao uso incorrecto desta poderosa ferramenta são elevados e são potenciados pela sua crescente versatilidade, como realçado pelos resultados dos *benchmarks* das conferências NUMISHEET. Por exemplo, no *BM 4 – Pre-strain on Spring-back of 2-D Draw Bending*, proposto pelo comité organizador dos *benchmarks* da conferência NUMISHEET'11, um dos participantes reportou os resultados apresentados na Figura 1.1. Esta figura compara a avaliação experimental da evolução da força do punção com o respectivo deslocamento e o perfil após a recuperação elástica, com os resultados numéricos previstos por um dos participantes (Chung *et al.*, 2011). O controlo incorrecto dos parâmetros numéricos de um código dinâmico explícito comercial conduziu a uma previsão incorrecta de ambos os resultados.

[1] CAE – Computer Aided Engineering



(a) (b)

Figura 1.1. Comparação entre resultados experimentais e numéricos para o BM4, NUMISHEET'11: (a) evolução da força do punção com o deslocamento; (b) perfil após retorno elástico (Chung *et al.*, 2011).

Perante a necessidade de alertar os utilizadores para os problemas relacionados com o uso desmedido de códigos de análise pelo MEF, para a estampagem de chapas metálicas, este trabalho descreve o desenvolvimento da plataforma DD3LT, com vista a apoiar o ensino e a aprendizagem de métodos de produção virtual de estampagem de chapas metálicas. Esta plataforma é suportada pelo código DD3IMP, que é um programa académico, continuamente desenvolvido e optimizado para simular processos de conformação plástica de chapas metálicas (Menezes e Teodosiu, 2000; Oliveira *et al.*, 2003; Oliveira *et al.*, 2008). A plataforma integra uma base de dados extensiva de *benchmarks* de problemas de estampagem de chapas metálicas e uma aplicação interactiva para pré e pós processamento dos modelos. Na secção seguinte é feita uma descrição das principais funcionalidades do programa DD3IMP.

1.1. DD3IMP

O código de elementos finitos DD3IMP (acrónimo de Deep-Drawing 3D IMPlicit code) tem vindo a ser especialmente desenvolvido para simular processos de conformação de chapa metálica. No modelo mecânico são consideradas grandes deformações e rotações elastoplásticas, e assume que, relativamente à unidade, as

deformações elásticas são suficientemente pequenas para serem desprezadas. Neste contexto, o comportamento elástico é considerado isotrópico. O comportamento plástico é descrito através de modelos constitutivos fenomenológicos baseados na definição de: (i) uma lei de escoamento associada; (ii) um critério de cedência plástica e (iii) uma lei de encruamento. Existem vários critérios de cedência plástica implementados no DD3IMP, considerando isotropia (von Mises, 1913; Drucker, 1949; Hosford, 1972) e ortotropia (Hill, 1948; Bartlat *et al.*, 1991; Karafilis e Boyce, 1993; Cazacu e Barlat, 2001; Drucker +L (Bartlat *et al.*, 1991; Karafilis e Boyce, 1993; Cazacu e Barlat, 2001) e Cazacu *et al.* 2006). O encruamento isotrópico pode ser descrito através das leis de Swift (1947) ou de Voce (1948), que podem ser combinadas com as leis de encruamento cinemático de Prager (1955) e Lemaître e Chaboche (1985). O comportamento do encruamento pode também ser descrito através dos modelos microestruturais completo ou simplificado de Teodosiu e Hu (1998).

A formulação lagrangiana actualizada implementada, baseia-se no princípio das velocidades virtuais proposto por McMeeking e Rice (1975). É utilizada uma abordagem explícita para calcular uma primeira solução aproximada dos deslocamentos nodais, do estado de tensão e das forças de contacto com atrito. É implementada uma estratégia r_{min} na imposição de várias restrições ao tamanho do incremento temporal de forma a melhorar a convergência (Yamada *et al.*, 1968). A solução de previsão é iterativamente corrigida usando um algoritmo de Newton-Raphson, terminando quando é alcançado um estado de equilíbrio satisfatório para o corpo deformável. É então possível actualizar a configuração da chapa metálica e todas a variáveis de estado, sendo estes dados passados para o início do incremento seguinte. Este procedimento é repetido até ao fim do processo (Menezes e Teodosiu, 2000).

Nos processos de conformação metálica as condições de fronteira são ditadas pelo contacto estabelecido entre a chapa metálica e as ferramentas. Tais condições são continuamente alteradas durante o processo de conformação, aumentando a importância da necessidade de avaliar correctamente a superfície de contacto e o tipo de contacto estabelecido entre cada ponto do corpo deformável e as ferramentas. O DD3IMP adopta um algoritmo de detecção de contacto do tipo *master-slave*, uma vez que se considera que as ferramentas se comportam como corpos rígidos. O atrito entre as ferramentas e o corpo deformável é modelado através da lei clássica de Coulomb. A gestão do problema de

contacto com atrito é realizada com o auxílio do método do lagrangiano aumentado (Menezes e Teodosiu, 2000; Oliveira *et al.*, 2003; Oliveira *et al.*, 2008). Tal como mencionado anteriormente, o algoritmo implícito de Newton-Raphson é utilizado para resolver, num único ciclo iterativo, todas as não linearidades associadas ao contacto com atrito e ao comportamento elastoplástico do corpo deformável. A Tabela 1.1 apresenta um resumo do algoritmo principal do DD3IMP.

Tabela 1.1. Algoritmo principal do DD3IMP.

INÍCIO

- Ler e verificar os dados de entrada
- Inicializar o número do incremento $N = 1$
- Repetir
  - ***Previsão***
    - ✓ Impor o incremento de tentativa
    - ✓ Impor as condições de contacto com atrito
    - ✓ Calcular a matriz de rigidez tangente e o vector nodal de força
    - ✓ Resolver o sistema de equações para o incremento de tentativa
    - ✓ Calcular o campo de tensão e deformação
    - ✓ Calcular o valor de r_{min} para definir o incremento efectivo
    - ✓ Actualizar a posição da chapa e das ferramentas
    - ✓ Actualizar as variáveis de contacto
  - ***Correcção***
    - ✓ Actualizar as condições de contacto com atrito
    - ✓ Calcular os incrementos de deformação e rotação
    - ✓ Integrar a lei de comportamento do material
    - ✓ Resolver o sistema de equações
  - Validar uma eventual mudança de fase do processo
  - Actualizar o número do incremento $N = N + 1$

  Até ao fim do processo

FIM

As ferramentas de estampagem são modeladas usando superfícies paramétricas, do tipo Bézier e/ou Nagata (Neto *et al.*, 2012). O corpo deformável é discretizado com elementos finitos sólidos 3D. Ainda que este tipo de elementos seja penalizado neste tipo de aplicações pelos custos e eficiência computacional, os elementos sólidos têm imensas vantagens, entre as quais: (i) permitem a avaliação precisa das forças de contacto através de uma descrição precisa da evolução do contacto e da variação de espessura; (ii) o contacto simultâneo em ambos os lados da chapa é naturalmente resolvido sem nenhuma estratégia numérica particular. Por outro lado, estudos indicam que é necessário utilizar elementos sólidos para garantir a correcta previsão do retorno elástico, quando a relação entre o raio da ferramenta e a espessura do corpo deformável é inferior a 5-6 (Li *et al.*, 2002). Estes factos motivaram diversos estudos com vista ao desenvolvimento de elementos sólidos adequados à simulação de estampagem de chapas metálicas (Jiao e Li, 2000; Areias *et al.*, 2003; Wang e Wagoner, 2005; Reese, 2005). No DD3IMP, o tradicional elemento finito trilinear hexaédrico de oito nós, pode ser aplicado usando integração completa, integração reduzida ou associado a um esquema de integração reduzida selectiva (IRS) (Hughes, 1980). Embora o esquema de IRS possa exibir modos espúrios de energia zero em problemas dominante pela torsão, este tipo de elementos finitos permite a computação eficiente da evolução da espessura bem como dos gradientes de tensão ao longo da espessura (Menezes *et al.*, 1991; Alves e Menezes, 2001), dependendo do tipo de aplicação e do número de elementos ao longo da espessura e no plano da chapa. Existem outros tipos de elementos sólidos disponíveis na libraria de elementos finitos, incluindo o elemento *serendipity* de vinte nós e o elemento hexaédrico tri-quadrático de vinte e sete nós (Alves, 2003).

O DD3IMP permite o uso de três estratégias diferentes para simular a fase de retorno elástico. A primeira pode ser entendida simplesmente como a continuação do processo de estampagem, em que o movimento das ferramentas é invertido e a simulação é realizada até ao final do processo, em que se dá a perda total de contacto entre as ferramentas e a parte conformada. Esta estratégia está fisicamente de acordo com o processo real, pois permite que a alteração nas áreas de contacto entre a chapa e as ferramentas de estampagem seja monitorizada durante a fase de descarregamento. Contudo, este procedimento leva a um aumento considerável do tempo de computação e pode também conduzir a problemas de convergência, associados o carácter discreto do

contacto. A segunda estratégia possível consiste na remoção das ferramentas, uma a uma, usando apenas um incremento temporal por ferramenta (punção, matriz, …), forçando o equilíbrio a cada incremento através de um ciclo iterativo implícito. Na terceira estratégia, realiza-se a recuperação elástica num único incremento, removendo todas as ferramentas em simultâneo e forçando o corpo deformável a entrar em equilíbrio. Nesta última estratégia, denominada de *One Step Springback*, desaparecem todas as restrições impostas pelas ferramentas no início da fase de retorno elástico. Uma vez que a solução inicial para o esquema implícito é a configuração no final da fase de conformação, não existe necessidade de encontrar uma solução inicial aproximada.

A definição dos dados de entrada para uma simulação é realizada com o auxílio de ficheiros ANSI ASCII, de formato pré-definido o que é uma abordagem típica nos códigos baseados no MEF. Na subsecção seguinte descreve-se de forma sucinta o tipo de ficheiros utilizados, de modo a evidenciar as vantagens e desvantagens da sua utilização.

1.1.1. Ficheiros de dados de entrada

O conjunto mínimo de ficheiros de entrada necessários para definir um modelo no DD3IMP é apresentado na Figura 1.2, que mostra também a interacção entre eles. Tal como mencionado anteriormente, o formato adoptado para os ficheiros de dados de entrada é o ANSI ASCII e a extensão por defeito é **.dat*. Globalmente, estes ficheiros contêm a seguinte informação:

- **DD3_bcon**: ficheiro utilizado para impor as condições de fronteira do problema. Estes podem ser planos com deslocamento restringido (ex. planos de simetria) ou pontos específicos com deslocamentos fixos (ex. pontos utilizados no controlo da fase de retorno elástico);
- **DD3_contact**: ficheiro utilizado para definir os *sets* de contacto, i.e. associar regiões específicas potenciais de contacto com ferramentas específicas, no intuito de minimizar os problemas associados à pesquisa de contacto entre corpos com grandes dimensões. O coeficiente de atrito de Coulomb, entre o corpo deformável e a ferramenta, é também definido neste ficheiro. É possível definir um coeficiente de atrito global, coeficientes de atrito diferentes para cada *set* de contacto (ex. dois coeficientes de atrito diferentes, um para a zona entre superfície superior do corpo deformável e a ferramenta, outro para a zona entre a superfície inferior do corpo

deformável e a ferramenta) e um coeficiente de atrito para cada superfície paramétrica que descreve a geometria da ferramenta;

- **DD3_input**: ficheiro utilizado para definir todos os parâmetros numéricos (ex. critério de convergência, número máximo de iterações, tolerâncias e resíduos) bem como os dados de saída (ex. número de ficheiros de saída e conjunto de variáveis armazenadas para visualização de resultados em pós-processamento);
- **DD3_mater**: ficheiro utilizado para definir os parâmetros do material, em concordância com a lei de encruamento e o critério de plasticidade pré-seleccionado. Este ficheiro é também utilizado para definir a orientação do sistema de eixos material (direcção de laminagem) em relação ao sistema de coordenadas globais. Para cada material é necessário definir um ficheiro com toda esta informação. A associação entre o material X, definido na conectividade da discretização do corpo deformável, e as suas propriedades mecânicas é feita na designação do ficheiro, DD3_materX.dat;
- **DD3_mesh**: ficheiro utilizado para definir a discretização do corpo deformável utilizando elementos finitos: coordenadas de cada nó, a conectividade de cada elemento (i.e. os nós que definem as arestas de cada elemento) e o número do material (informação omissa caso exista apenas um material). Actualmente, estão disponíveis vários formatos para este ficheiro, baseados no pré-processador utilizado para criar a malha. A definição da discretização com o gerador interno do DD3IMP surge associada ao ficheiro DD3_mesh.dat;
- **DD3_phase**: ficheiro utilizado para definir as condições de processo, incluindo o número total de ferramentas e de fases, e a acção de cada ferramenta em cada fase. O deslocamento inicial das ferramentas é também definido neste ficheiro, bem como a associação entre cada ferramenta e os *sets* de contacto, previamente definidos no DD3_contact.dat;
- **DD3_tool**: ficheiro utilizado para definir a geometria das ferramentas. A definição das superfícies paramétricas pode ser feita com superfícies Bézier ou Nagata. A utilização de superfícies Nagata impõe a definição de uma discretização das superfícies com elementos finitos, sendo o vector normal de cada nó da superfície discretizada calculado a partir desta discretização ou, opcionalmente, quando a informação acerca da geometria da ferramenta está disponível no formato IGES, a

normal em cada nó da discretização pode ser avaliada usando este ficheiro. A discretização pode ser obtida usando por exemplo o pré-processador GID[2]. Desta forma, ao utilizar as superfícies Nagata, é necessário definir para cada uma das N ferramentas um ficheiro DD3_toolN.igs e um ficheiro DD3_toolN.msh.

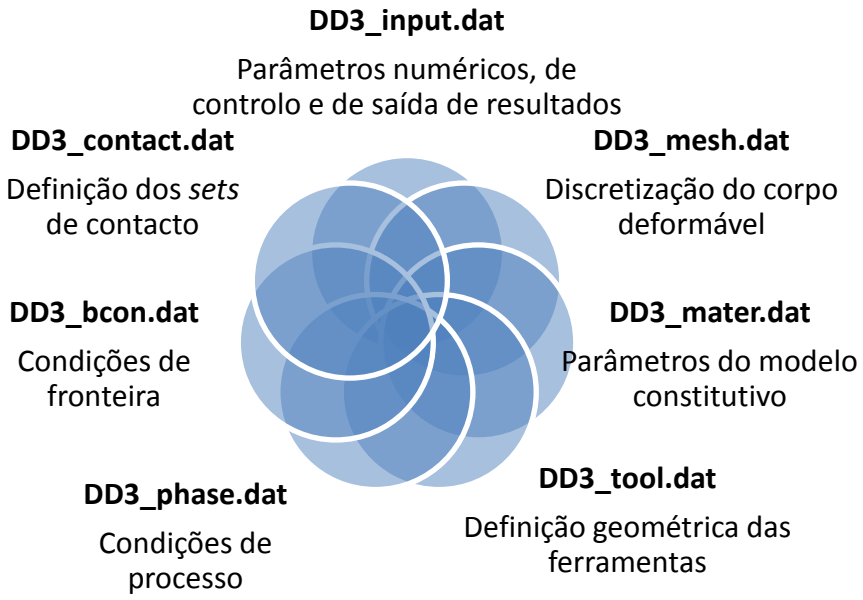


Figura 1.2. Conjunto mínimo de ficheiros de entrada necessários para definir um modelo no DD3IMP.

A Figura 1.3 apresenta a título de exemplo o ficheiro de entrada DD3_bcon.dat, necessário para definir as condições de fronteira do problema. A figura permite realçar as principais características, comuns à maioria dos ficheiros de entrada: (i) a informação é estruturada de forma colunar; (ii) cada parâmetro tem um número de caracteres fixo associado ao seu formato, sendo necessário conhecer este formato para poder introduzir dados de entrada válidos. Este tipo de *layout* permite uma fácil leitura para utilizadores familiarizados. No entanto, o processo de familiarização nem sempre é expedito graças ao enorme conjunto de parâmetros e opções. No processo de aprendizagem, direccionado para os processos de conformação plástica de chapas metálicas, apenas são relevantes um conjunto limitado de parâmetros e opções. De facto, nesse processo é sobretudo importante que os utilizadores tenham acesso a diferentes modelos, para os quais possam estudar o impacto nos resultados de um determinado

[2] http://www.gidhome.com/

parâmetro, de forma isolada. Por outro lado, importa realçar que este tipo de *layout* favorece a ocorrência de erros, cuja detecção é dificultada pela elevada quantidade de parâmetros a ter em conta, em particular para utilizadores menos experientes. Assim, um dos objectivos definidos para a plataforma interactiva foi procurar ultrapassar esta desvantagem, sem condicionar a familiarização com a estrutura dos ficheiros de entrada.

```
1010******       < DD3IMP O10.x >   File "bcon.dat"  01.07.2009      ***********
**********        MANDATORY file to input Boundary Conditions        ***********
====================+=================================================================
Boundary conditions|  NSC=   2      ERR=     1.D-5                      BCID
  Ax+By+Cz=D       |          A          B          C          D   COORD     ID  NRES
                P1 |          1          0          0          0       1      2     0
                P2 |          0          1          0          0       2      2     0
  Restrictions     |      COORD    COND     VALUE      ERR2
              P2R1 |          .       .          .     1.D-5
====================+=================================================================
OSS One Step       | NRES=   1
    Springback     |          x          y          z     COORD   Xinit
                N1 |        0.0        0.0        0.0         3       1
====================+=================================================================

Remarks:
  COND: 0 -> .EQ.   | COORD:  1 -> X       |
        1 -> .GE.   |         2 -> Y       |
       -1 -> .LE.   |         3 -> Z       |
                               4 -> all     |
                               5 -> radius  |
  Xinit: 1 -> yes
         0 -> no

  BCID:  = 2  -> symmetry condition
         >10  -> ID group of DOFs
```

Figura 1.3. Ficheiro de entrada de dados para as condições de fronteira DD3_bcon.dat.

1.1.2. Ficheiros de saída

No decurso da simulação o conjunto mínimo de informação gerado é organizado em quatro tipos distintos de ficheiros: (i) stat.res, que contém a informação relativa ao número total de incrementos, número de iterações, número de tentativas, valor do critério de convergência e tempo de cálculo; (ii) [TOOLname].res, que contém a informação relativa à força e ao deslocamento da ferramenta TOOLname, em cada um dos

eixos do sistema global de coordenadas; (iii) conjunto de ficheiros do tipo [GIDname].msh, [GIDname].res e [GIDname].tool.msh, que contém a informação necessária para a visualização no pós-processador GID; e (iv) [FILEname].ufo, do tipo não formatado, que contém toda a informação relativa a um determinado incremento.

Em geral, no processo de aprendizagem, os utilizadores apenas acedem ao ficheiro stat.res para comparar tempos de computação e observar o comportamento do processo iterativo em função da dimensão do incremento de tentativa seleccionado. Os ficheiros do tipo [TOOLname].res são utilizados para construir o gráfico de evolução da força das ferramentas, em função do seu deslocamento, que é um parâmetro importante para analisar a energia despendida num processo de conformação. Neste contexto, é fundamental que os utilizadores tenham um acesso facilitado a esta informação, em particular para a ferramenta que controla a fase.

Em relação ao conjunto de ficheiros do tipo [GIDname].msh, [GIDname].res e [GIDname].tool.msh, existem várias opções para o GIDname, como se mostra na Figura 1.4. O conjunto de ficheiros do tipo GIDname = GID_ini é sempre produzido no início da simulação de modo a permitir a visualização das condições iniciais do processo. O conjunto de ficheiros do tipo GIDname = GID_simu é também produzido no início da simulação, mas permite armazenar informação a cada número fixo de incrementos (iGID), de modo a visualizar a evolução do processo. Quando é activada a opção NSTART=98, é gravado o conjunto de ficheiros do tipo GIDname = GID_DD3[NST], que armazena a informação relativa ao processo iterativo, no incremento NST. Para todos estes tipos de ficheiros, as variáveis a armazenar são activadas no ficheiro DD3_input.dat (como indicado na Figura 1.4) antes de executar a simulação.

Tal como mencionado anteriormente, no decurso da simulação são igualmente guardados ficheiros não formatados do tipo [FILEname].ufo. Podem ser gerados três tipos de ficheiros não formatados: (i) info[NST].ufo, guardados em função do número entre incrementos, NOUT, pré-definido no ficheiro DD3_input.dat; (ii) p[IPH]dep[FBP].ufo, guardados em função da fase, IPH, e do valor pré-definido para o deslocamento da ferramenta que controla a fase, FBP, no ficheiro DD3_input.dat; e (iii) p[IPH]end.ufo, que é guardado sempre que é validada uma mudança de fase. Cada ficheiro do tipo [FILEname].ufo pode ser tratado, executando o DD3IMP indicando como argumento o nome do ficheiro, de modo a produzir o conjunto de ficheiros do tipo

GID_[FILEname].msh, GID_[FILEname].res e GID_[FILEname].tool.msh, para pós-processamento no GID. As variáveis a armazenar são activadas no ficheiro DD3_input.dat, como indicado na Figura 1.4. No entanto, como neste caso toda a informação está armazenada estas opções podem ser alteradas após a execução da simulação.

Em resumo, como os ficheiros necessários para visualização no GID são sempre gerados pelo DD3IMP, o utilizador pode sempre proceder à análise utilizando este pós-processador.

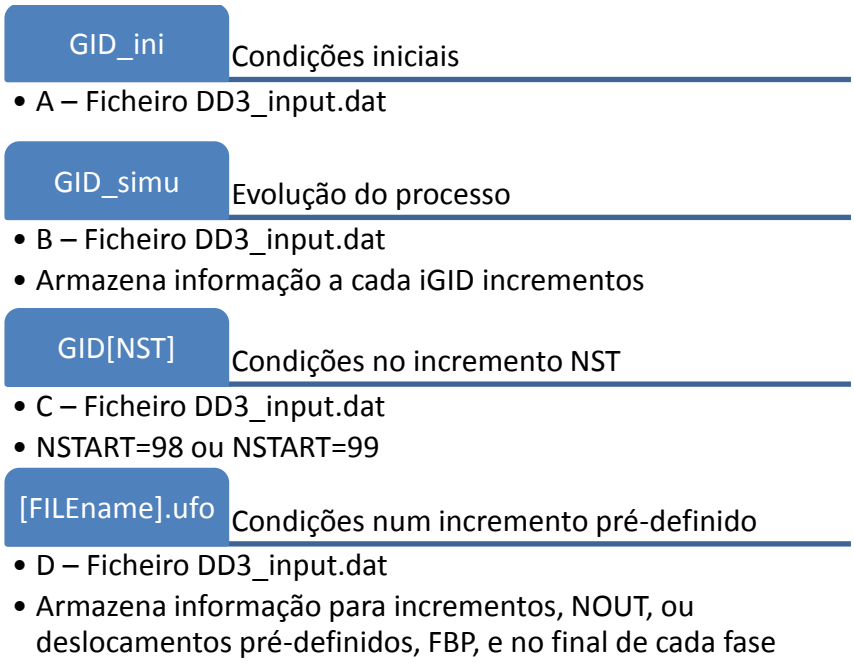


Figura 1.4. Opções disponíveis para a produção do conjunto de ficheiros para visualização no pós-processador GID.

1.2. Objectivos

Os objectivos principais deste trabalho são diminuir a complexidade da utilização do DD3IMP e diminuir o tempo e esforço necessário para a atingir a fase de interpretação dos dados de uma simulação. Neste contexto, estabeleceu-se como objectivo inicial a criação de uma plataforma interactiva que permitisse a selecção de um processo de estampagem, a partir de uma panóplia de simulações para as quais existem ficheiros de dados de entrada previamente definidos. Desta forma, a selecção implica a reorganização estruturada da informação subjacente aos ficheiros dos dados de entrada.

A plataforma a criar deveria ainda permitir editar os parâmetros dos ficheiros de entrada, bem como executar a respectiva simulação com o DD3IMP. Esta plataforma seria direccionada para o ensino e aprendizagem de métodos de produção virtual associados às tecnologias de estampagem de chapas metálicas, devendo o seu desenvolvimento visar colmatar as necessidades didácticas, facilitando o acesso a problemas estudados anteriormente e reduzindo a complexidade da utilização do DD3IMP.

No intuito de armazenar e estruturar a informação relativa a problemas anteriormente estudados, retomou-se o desenvolvimento da base de dados inicialmente criada para o efeito. Implicitamente, uma vez que a base de dados foi construída com recurso ao MS Access, a plataforma teria que ser desenvolvida numa tecnologia compatível com este tipo de organização de informação.

A plataforma deveria ser apresentada perante o utilizador sob a forma de um interface gráfico, permitindo a selecção intuitiva de processos de estampagem constantes da base de dados, e opcionalmente permitir a edição de qualquer tipo de parâmetros da simulação. No desenvolvimento do interface gráfico foi ainda considerada a possibilidade de actualização da versão do DD3IMP a partir da base de dados. Assim, tentou-se minimizar a quantidade necessária de alterações no interface gráfico num dado procedimento de actualização.

A plataforma interactiva viria desta forma a ser constituída por um interface gráfico e uma base de dados, tendo sido denominada de DD3LT – *Deep Drawing 3D Learning and Teaching*.

1.3. Enquadramento e guia de leitura

Neste capítulo introdutório procurou-se identificar e justificar os objectivos estabelecidos para o trabalho desenvolvido no âmbito desta tese. Este consistiu na elaboração da plataforma interactiva DD3LT, com recurso à integração de uma base de dados com um ambiente interactivo, de modo a incrementar a amigabilidade do programa DD3IMP.

Após um estudo extensivo das tecnologias existentes, e tendo em conta:

1) A dependência do método de acesso à base de dados;
2) A compatibilidade de formatos ANSI e UNICODE;
3) A compatibilidade com plataformas x86 e x64;

4) A compatibilidade com os sistemas operativos *Windows XP*, *Windows Vista*, *Windows Seven*;
5) A compatibilidade com múltiplas configurações de DPI;
6) As perspectivas de suporte no futuro;
7) A compatibilidade com bibliotecas de renderização 3D (VTK);
8) As licenças de *software* disponíveis no Departamento de Engenharia Mecânica – FCTUC;

Foram seleccionadas para o desenvolvimento da aplicação:

a) A linguagem *C++* na plataforma de desenvolvimento *MS Visual Studio 2010 Professional SP1*;
b) As classes MFC10.0 disponíveis no *.NET FrameWork 4.0*.

O acesso à base de dados foi é feito por intermédio do driver *ODBC32.dll*.

Os capítulos seguintes procuram descrever de forma sucinta as aplicações desenvolvidas. O objectivo desta descrição é sobretudo facilitar a utilização futura da base de dados, e da aplicação. O desenvolvimento requer conhecimentos especializados, para os quais são referidas algumas referências bibliográficas. No capítulo 2 descreve-se a estrutura global da base de dados, de acordo com o procedimento recomendado para a introdução de uma nova simulação. No capítulo 3 apresenta-se a estrutura global da interface gráfica, com ênfase para a ligação com a base de dados. O capítulo 4 é dedicado a apresentar a interface gráfica construída de modo a guiar a sua futura utilização. Por último, no capítulo 5 apresentam-se algumas conclusões e recomendações de trabalho futuro.

2. BASE DE DADOS

A base de dados foi construída de modo a armazenar toda a informação necessária para executar uma simulação, para um modelo pré-definido, organizando essa informação de forma estruturada. Esta organização permite aceder facilmente aos ficheiros de dados de entrada de modelos existentes, minimizando o tempo de pré-processamento e permitindo identificar os parâmetros utilizados nas simulações. A estrutura da base de dados foi desenvolvida considerando a necessidade de uma organização hierárquica da informação relativa a cada simulação. Por este motivo, para além das tabelas necessárias à estruturação relacional da informação, contida em cada ficheiro de entrada de dados, foram construídas tabelas auxiliares que organizam a informação de cada simulação, e agilizam o processo de selecção da simulação de acordo com a estrutura:

1. Geometria das ferramentas: este é o primeiro parâmetro a ser definido, uma vez que está relacionado com as condições de processo (i.e. número e tipos de fases) e condições de contacto (i.e. zonas de contacto). A geometria das ferramentas também pode ditar as condições de fronteira, isto porque quando toda a geometria é definida as ferramentas podem ser utilizadas para simular o modelo completo ou apenas metade ou um quarto do componente.
2. Material: esta selecção permite seleccionar o material a utilizar na simulação numérica, mas também o modelo constitutivo a ser utilizado, i.e. a lei de encruamento e o critério de plasticidade.
3. Geometria do corpo deformável: esta selecção é baseada nas principais características geométricas das ferramentas, i.e. quadrada, rectangular ou circular. Em função destas características, podem ser seleccionadas diferentes discretizações no plano e em espessura, uma vez que a plataforma interactiva permite o redimensionamento das discretizações armazenadas na base de dados. Podem também ser seleccionados diferentes tipos de elementos finitos e de integração numérica.
4. Parâmetros numéricos e de controlo de saída de informação.

A base de dados construída com base nesta estrutura relacional apresenta um total de 39 tabelas, a que estão associadas chaves primárias e estrangeiras. Por definição, a

chave primária é o campo que serve de índice e de identificador principal de uma tabela. Uma chave estrangeira é um campo que é um atributo de uma tabela sendo simultaneamente a chave primária doutra. É a definição destas chaves, em cada tabela, que permite definir o tipo de relacionamento entre elas.

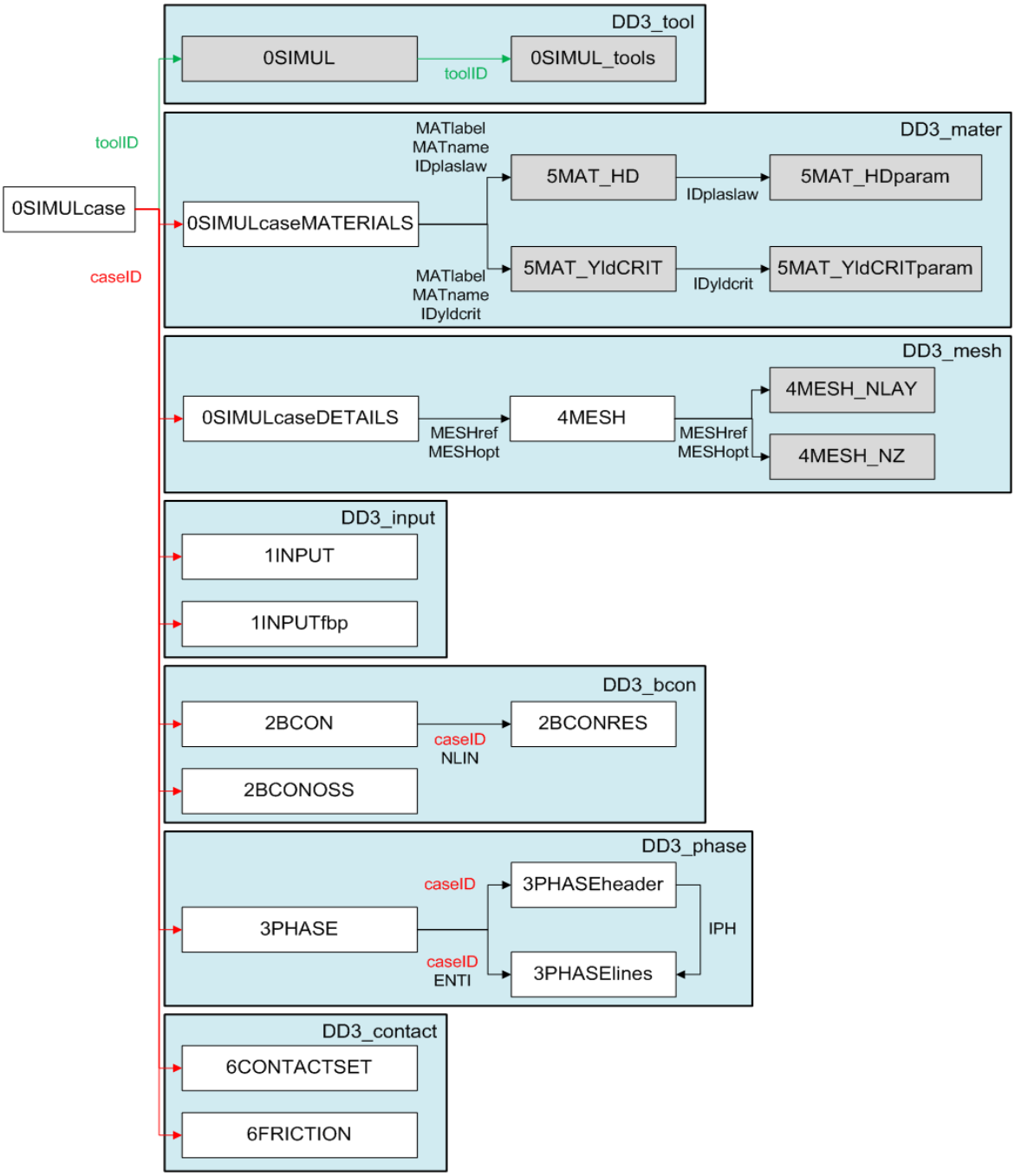


Figura 2.1. Representação esquemática das principais tabelas, utilizadas na base de dados para a definição dos ficheiros de dados de entrada, e das relações estabelecidas entre elas com o auxílio da chave primária *caseID* e de chaves estrangeiras.

Na base de dados construída é utilizada uma chave primária, que identifica de modo inequívoco o conjunto de parâmetros necessários para construir todos os ficheiros de entrada para realizar uma simulação numérica com o DD3IMP. Esta chave primária corresponde aos campos caseID, que como se mostra esquematicamente na Figura 2.1 estabelecem a relação entre as principais tabelas. Esta figura apresenta um resumo das tabelas utilizadas na base de dados para definir uma simulação, e corresponde também à sequência de armazenamento/introdução da informação.

Na introdução de uma nova simulação (novo caseID) a primeira tabela a ser preenchida é a 0SIMULcase, como se mostra na Figura 2.1. Nesta tabela basicamente é identificada a nova simulação, através do campo caseID, e definida a ferramenta associada ao processo (campo toolID). Sendo assim é necessário que a ferramenta seja definida na tabela 0SIMUL. Esta tabela é independente do caseID, como se mostra na Figura 2.1. De facto, a base de dados permite armazenar ferramentas às quais não foi associado nenhum conjunto de ficheiros de entrada. Por outro lado, esta independência permite definir um número ilimitado de simulações relacionadas com a mesma ferramenta. A associação dos materiais a cada simulação (caseID) é realizada na tabela 0SIMULcaseMATERIALS, com base no tipo de material (campo MATname), na metodologia utilizada para a identificação dos parâmetros (campo MATlabel), na lei de encruamento (campo IDplaslaw) e no critério de plasticidade (campo IDyldcrit). A tabela que contém os parâmetros da lei de encruamento (5MAT_HD) e a que contém os parâmetros do critério de plasticidade (5MAT_YldCRIT) são independentes do caseID, como se mostra na Figura 2.1. Deste modo, a base de dados permite armazenar parâmetros constitutivos de materiais aos quais não foi associado nenhuma simulação. Por outro lado, esta independência permite definir um número ilimitado de simulações associadas ao mesmo material. A base de dados permite igualmente definir novas leis de encruamento ou critérios de plasticidade, com o auxílio das tabelas 5MAT_HDparam e 5MAT_YldCRITparam. Sendo assim, é necessário garantir que a lei de encruamento seleccionada (campo IDplaslaw) é definida na tabela 5MAT_HDparam e que o critério de plasticidade pretendido (campo IDyldcrit) existe na tabela 5MAT_YldCRITparam. A selecção das características geométricas e da discretização do corpo deformável é realizada na tabela 0SIMULcaseDETAILS. As opções disponíveis para cada ferramenta (campo MESHref) são armazenadas na tabela 4MESH, sendo cada uma identificada pelo campo MESHopt. Também a tabela 4MESH é

independente do caseID, o que permite armazenar na base de dados diferentes discretizações que não necessitam de estar associadas a um exemplo específico.

As restantes tabelas apresentadas na Figura 2.1 permitem a definição do conjunto de parâmetros associados a cada um dos ficheiros de entrada, como indicado nessa figura. A ligação entre a tabela 0SIMULcase e cada uma destas tabelas e realizada com o auxílio da chave primária, caseID, o que implica o seu preenchimento para cada nova simulação.

A descrição destas tabelas será realizada nas subsecções seguintes, evidenciando a ligação com os ficheiros de dados de entrada do DD3IMP. No ANEXO A apresenta-se um esquema resumo das relações entre as 39 tabelas utilizadas na base de dados, com a indicação dos diferentes campos utilizados em cada tabela.

2.1. 0SIMULcase

Tal como foi referido anteriormente, a tabela 0SIMULcase é utilizada para identificar de forma inequívoca cada nova simulação, com o auxílio do campo caseID. A cada nova simulação é associada uma ferramenta, cujo nome é definido no campo toolID, que permite a ligação à tabela 0SIMUL. Os restantes campos desta tabela correspondem à descrição das principais características da simulação (campo SIMULoption) e a notas acerca de detalhes que o utilizador considere relevantes para memória futura (campo SIMULnotes).

```
1010******       < DD3IMP O10.x >   File "bcon.dat"  01.07.2009      ***********
**********        MANDATORY file to input Boundary Conditions        ***********
 ====================+=============================================================
 Boundary conditions|  NSC=   2      ERR=     1.D-5                       BCID
   Ax+By+Cz=D       |         A         B         C         D   COORD     ID  NRES
                 P1 |         1         0         0         0       1      2     0
                 P2 |         0         1         0         0       2      2     0
   Restrictions     |     COORD    COND     VALUE      ERR2
               P2R1 |         .       .         .     1.D-5
 ====================+=============================================================
```

Figura 2.2. Representação da utilização dos campos DESCR1 e DESCR2, definidos na tabela 0SIMULcase, no cabeçalho dos ficheiros de dados de entrada.

Os campos DESCR1 e DESCR2 devem conter informação resumida acerca da simulação, uma vez que são utilizados para definir a primeira e a segunda linha do cabeçalho dos ficheiros de entrada, respectivamente. A Figura 2.2 apresenta um exemplo

da utilização da informação definida nestes campos no ficheiro DD3_bcon.dat. No Quadro B.4 do ANEXO B apresenta-se a descrição dos campos utilizados nesta tabela.

2.2. 0SIMUL e 0SIMUL_tools

Na tabela 0SIMUL define-se essencialmente a informação relativa à geometria das ferramentas. A definição das ferramentas no DD3IMP é realizada com o auxílio de ficheiros externos, que devem ser armazenados na base de dados.

No caso de pelo menos uma das ferramentas ser definida com o auxílio de superfícies paramétricas do tipo Bézier, o campo TOOLfile_BEZ deve conter a informação necessária para a definição do ficheiro DD3_tool.dat. Caso se opte por utilizar superfícies do tipo Nagata, existem duas opções disponíveis para a definição da discretização da superfície das ferramentas: (i) estarem todas definidas num único ficheiro; (ii) serem utilizados diferentes ficheiros para cada ferramenta. No caso de o utilizador optar por um único ficheiro, o campo TOOLfile_NAG deve conter a informação necessária para a definição do ficheiro DD3_tool.msh. Caso contrário, essa informação deve apenas ser introduzida na tabela 0SIMUL_tools, no campo com o mesmo nome, uma vez que esta tabela permite associar o ficheiro ao número da ferramenta (campo TOOLnum).

Caso sejam utilizadas superfícies Nagata para a definição das superfícies das ferramentas, o utilizador pode também fornecer a informação correspondente à geometria CAD (ficheiro IGES com a definição das NURBS), de modo a permitir o cálculo da normal em cada nó. Caso seja definido um ficheiro único com a discretização de todas as ferramentas, o campo TOOLfile_IGS deve conter a informação necessária para a definição do ficheiro DD3_tool.igs. Caso se opte por diferentes ficheiros com a discretização de cada ferramenta, essa informação deve apenas ser introduzida na tabela 0SIMUL_tools, no campo com o mesmo nome.

Os campos DESCR e OBS são utilizados para introduzir na tabela 0SIMUL a descrição acerca das principais características da ferramenta e algumas observações. O campo IMAGE deve conter uma imagem ilustrativa da ferramenta. Para além disso, esta tabela permite também ao utilizador adicionar ficheiros que possam conter informação relevante acerca da ferramenta (campos FILEaux1 a FILEaux5).

Por último, a tabela 0SIMUL permite também seleccionar a versão do programa DD3IMP a utilizar. As opções para a versão do DD3IMP são definidas na tabela PARAM, mas importa realçar que esta informação não é efectivamente utilizada pela plataforma interactiva. A descrição dos campos utilizados nas tabelas 0SIMUL e PARAM é apresentada no Quadro B.5 e no Quadro B.6 do ANEXO B, respectivamente.

Para que a definição das ferramentas fique completa é necessário preencher também a tabela 0SIMUL_tools. De facto, tal como referido anteriormente, podem ser definidas diferentes ferramentas, utilizando superfícies Bézier e/ou Nagata. Caso se opte por superfícies Bézier, o ficheiro DD3_tool.dat contém a informação necessária para associar cada superfície a uma ferramenta específica. Caso se opte por utilizar um único ficheiro DD3_tool.msh para definir todas as ferramentas, o mesmo é verdade, porque o campo correspondente ao índice de material na conectividade é utilizado para definir a ferramenta. No entanto, caso se opte por utilizar diferentes ficheiros para definir cada uma das ferramentas com superfícies Nagata, essa informação não está disponível, sendo por isso fundamental associar cada ficheiro ao número da ferramenta correspondente. Assim, na tabela 0SIMUL_tools é identificado o número (TOOLnum) e nome (TOOLname) de cada uma das ferramentas utilizadas no processo. Caso se opte por utilizar diferentes ficheiros para definir cada uma das ferramentas com superfícies Nagata, a informação correspondente deve ser fornecida no campo TOOLfile_NAG (e TOOLfile_IGS, caso esteja disponível). A descrição dos campos utilizados na tabela 0SIMUL_tools é apresentada no Quadro B.7 do ANEXO B.

A informação relativa ao nome da ferramenta (campo TOOLname) é utilizada para fazer a correspondência correcta no ficheiro DD3_phase.dat, como se mostra na Figura 2.3. Importa no entanto realçar que este campo é uma excepção na base de dados, uma vez que se encontra duplicado na tabela 3PHASE, que será descrita na secção 2.9. Assim, optou-se por verificar primeiro se o campo TOOLname está preenchido na tabela 3PHASE. Caso não esteja, recorre-se à tabela 0SIMUL_tools para o preencher. Por último, importa referir que o campo TOOLname é utilizado para definir o nome dos ficheiros de saída com a informação relativa às ferramentas, como mencionado anteriormente na secção 1.1.2.

```
1010******       < DD3IMP O11.x >  File "phase.dat"  11.01.2010      ***********
**********        MANDATORY file to input phase data                  ***********
                NPH=    4   NbTOOL=    4   NbBCID=    0
====+=================================================================================
    |Initial Displacements and naming of the tools -  DIS0(I)
----+---------------------------------------------------------------------------------
   0|         X         Y         Z    <naming--------> BCID SET    lim-     lim+
    +---------+---------+---------+----+----------------+---+-------+-------+
    1       0.0       0.0       0.0    T1_Die                1
    2       0.0       0.0      0.78    T2_BlankHolder        2    -0.1    +1.0
    3       0.0       0.0      0.78    T3_Punch              2
====+=================================================================================
```

Figura 2.3. Representação da utilização do campo TOOLname, definidos na tabela 0SIMUL_tools, na zona de definição das ferramentas no ficheiro DD3_phase.dat.

2.3. 0SIMULcaseMATERIALS

A tabela 0SIMULcaseMATERIALS é utilizada para seleccionar cada um dos materiais associados a uma determinada simulação. O número total de materiais é definido na discretização do corpo deformável, com o auxílio da conectividade. Na tabela 0SIMULcaseMATERIALS é feita a correspondência entre cada índice de material, definido na conectividade, e as suas propriedades mecânicas, com o auxílio do campo iMAT. A tabela permite identificar o material (campo MATname) e a origem dos dados utilizados na identificação dos parâmetros (campo MATlabel). Como podem estar disponíveis diferentes modelos, nesta tabela é também feita a selecção do modelo constitutivo adoptado: lei de encruamento (campo IDplaslaw) e critério de plasticidade (campo IDyldcrit).

Por último, nesta tabela é ainda indicada a orientação do sistema de eixos do material em relação ao sistema de eixos global. No campo MxRDOX define-se o ângulo (em º) entre a direcção de laminagem e o eixo Ox do sistema global. No campo MxRDOZ define-se o ângulo (em º) entre a direcção de normal ao plano de ortotropia e o eixo Oz do sistema de eixos global. Esta informação pode ser distinta para cada um dos materiais utilizados no modelo e é preenchida no ficheiro DD3_mater[iMAT].dat correspondente, como se mostra na Figura 2.4. No Quadro B.8 do ANEXO B apresenta-se a descrição dos campos utilizados na tabela 0SIMULcaseMATERIALS.

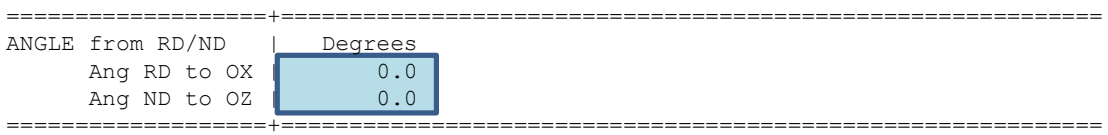


Figura 2.4. Representação da utilização dos campos MxRDOX e MxRDOZ, definidos na tabela 0SIMULcaseMATERIALS, no ficheiro DD3_mater.dat.

A tabela 0SIMULcaseMATERIALS relaciona-se com as tabelas 5MAT_HD e 5MAT_YldCRIT, através dos campos MATname, MATlabel, IDplaslaw e IDyldcrit, como se mostra na Figura 2.1. De facto, os parâmetros constitutivos associados à lei de encruamento são definidos na tabela 5MAT_HD, em função da lei seleccionada (campo IDplaslaw). De igual modo, os parâmetros constitutivos associados ao critério de plasticidade são definidos na tabela 5MAT_YldCRIT, em função do critério seleccionado (campo IDyldcrit). A selecção de um material implica que os campos MATname e MATlabel definido na tabela 0SIMULcaseMATERIALS tenham a informação correspondente nas tabelas 5MAT_HD e 5MAT_YldCRIT. De igual modo, é necessário garantir a correspondência para a lei de encruamento seleccionada no campo IDplaslaw, nas tabelas 0SIMULcaseMATERIALS e tabelas 5MAT_HD. Também é necessário garantir a correspondência no campo IDyldcrit entre as tabelas 0SIMULcaseMATERIALS e tabelas 5MAT_YldCRIT.

Tal como foi referido na secção 1.1, no DD3IMP estão implementadas diferentes leis de encruamento e critérios de plasticidade, sendo que cada utiliza um número distinto de parâmetros. De modo a facilitar a integração de novos modelos constitutivos, i.e. leis de encruamento ou critérios de plasticidade, optou-se por construir tabelas auxiliares para a definição dos parâmetros associados a cada modelo. Assim, a definição dos parâmetros de cada lei de encruamento é realizada na tabela 5MAT_HDparam e de cada critério de plasticidade na tabela 5MAT_YldCRITparam. Nas subsecções 2.3.1 e 2.3.2 descrevem-se as tabelas utilizadas para a definição da lei de encruamento e do critério de plasticidade, respectivamente.

2.3.1. 5MAT_HDparam e 5MAT_HD

A definição das leis de encruamento disponíveis no DD3IMP é realizada na tabela 5MAT_HDparam. Esta tabela é constituída por um número de entradas igual ao

número de leis de encruamento actualmente implementadas no DD3IMP, sendo que em cada campo é apresentada a descrição de cada um dos parâmetros necessários à sua definição. A correspondência entre a lei de encruamento seleccionada (campo IDplaslaw) e a selecção realizada no ficheiro DD3_mater[iMAT].dat é realizada com o auxílio do campo PlasLaw. O campo DESCR é utilizado para fazer uma descrição global da lei de encruamento correspondente. Os campos EM, PR e Y0 são por defeito associados ao módulo de elasticidade, ao coeficiente de Poisson e à tensão limite de elasticidade. Para além destes campos, na tabela 5MAT_HDparam estão pré-definidos 19 campos (Param02 a Param20) que podem ser associados a diferentes parâmetros da lei de encruamento. Este número de parâmetros foi seleccionado de modo a garantir alguma flexibilidade na implementação de novos modelos. Por último, no campo MATERx é definida a máscara para o ficheiro DD3_mater[iMAT].dat, correspondente a cada lei de encruamento. No Quadro B.9 do ANEXO B apresenta-se a descrição dos campos utilizados na tabela 5MAT_HDparam.

A definição do valor numérico associado às propriedades elástica e a cada um dos parâmetros da lei de encruamento seleccionada é realizada na tabela 5MAT_HD, para cada material. Sendo assim, para além dos campos MATlabel, MATname e IDplaslaw que permitem a selecção do material, esta tabela contém os campos EM, PR, Y0 e Param02 até Param20, que devem ser preenchidos de acordo com definição estabelecida na tabela 5MAT_HDparam. No Quadro B.10 do ANEXO B apresenta-se a descrição dos campos utilizados na tabela 5MAT_HD.

A informação definida nas tabelas 5MAT_HDparam e 5MAT_HD é utilizada para preencher a zona do ficheiro DD3_mater[iMAT].dat correspondente às propriedades elásticas e aos parâmetros da lei de encruamento, como se mostra Figura 2.5.

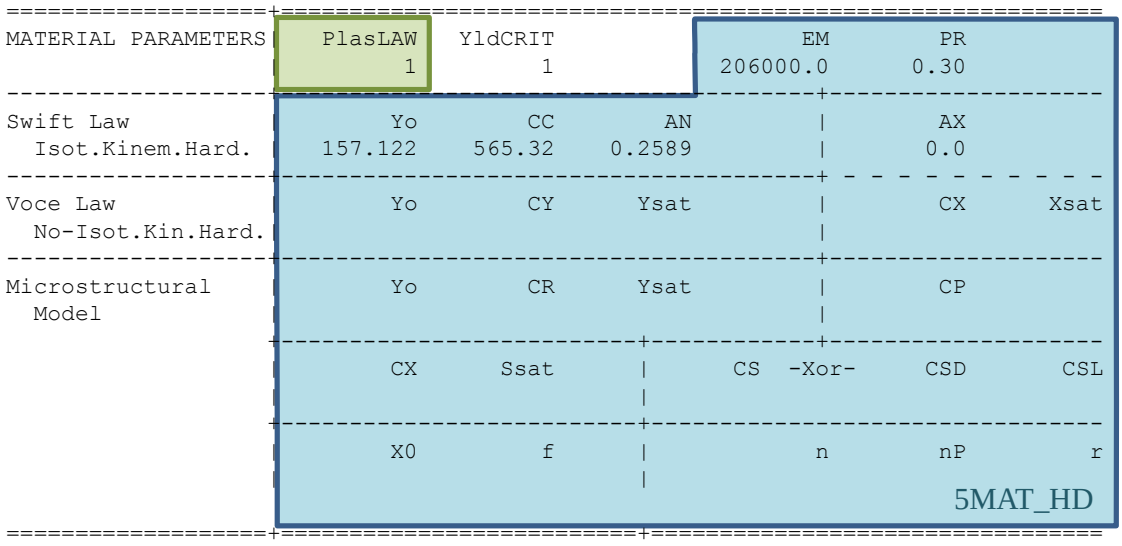


Figura 2.5. Representação da utilização do campo Plaslaw da tabela 5MAT_HDparam e EM, PR, Y0 e Param02 até Param20 da tabela 5MAT_HD, na zona de definição das propriedades elásticas e dos parâmetros da lei de encruamento no ficheiro DD3_mater.dat.

2.3.2. 5MAT_YldCRITparam e 5MAT_YldCRIT

A definição dos critérios de plasticidade disponíveis no DD3IMP é realizada na tabela 5MAT_YldCRITparam. Esta tabela é constituída por um número de entradas igual ao número de critérios de plasticidade actualmente implementados no DD3IMP, sendo que em cada campo é apresentada a descrição de cada um dos parâmetros necessários à sua definição. A correspondência entre o critério de plasticidade seleccionado (campo IDyldcrit) e a selecção realizada no ficheiro DD3_mater[iMAT].dat é realizada com o auxílio do campo YldCRIT. O campo DESCR é utilizado para fazer uma descrição global do critério de plasticidade correspondente. Para além destes campos, na tabela 5MAT_YldCRITparam estão pré-definidos 30 campos (Param01 a Param30) que podem ser associados a diferentes parâmetros do critério de plasticidade. Este número de parâmetros foi seleccionado de modo a garantir alguma flexibilidade na implementação de novos modelos. Por último, no campo MATERx é definida a máscara para o ficheiro DD3_mater[iMAT].dat, correspondente a cada critério. No Quadro B.11 do ANEXO B apresenta-se a descrição dos campos utilizados na tabela 5MAT_YldCRITparam.

A definição do valor numérico associado a cada um dos parâmetros do critério de plasticidade é realizada na tabela 5MAT_YldCRIT, para cada material. Sendo assim, para além dos campos MATlabel, MATname e IDyldcrit que permitem a selecção do material, esta tabela contém os campos Param01 até Param30, que devem ser preenchidos

de acordo com definição estabelecida na tabela 5MAT_YldCRITparam. No Quadro B.12 do ANEXO B apresenta-se a descrição dos campos utilizados na tabela 5MAT_YldCRIT.

A informação definida nas tabelas 5MAT_YldCRITparam e 5MAT_ YldCRIT é utilizada para preencher a zona do ficheiro DD3_mater[iMAT].dat correspondente aos parâmetros do critério de plasticidade, como se mostra Figura 2.6.

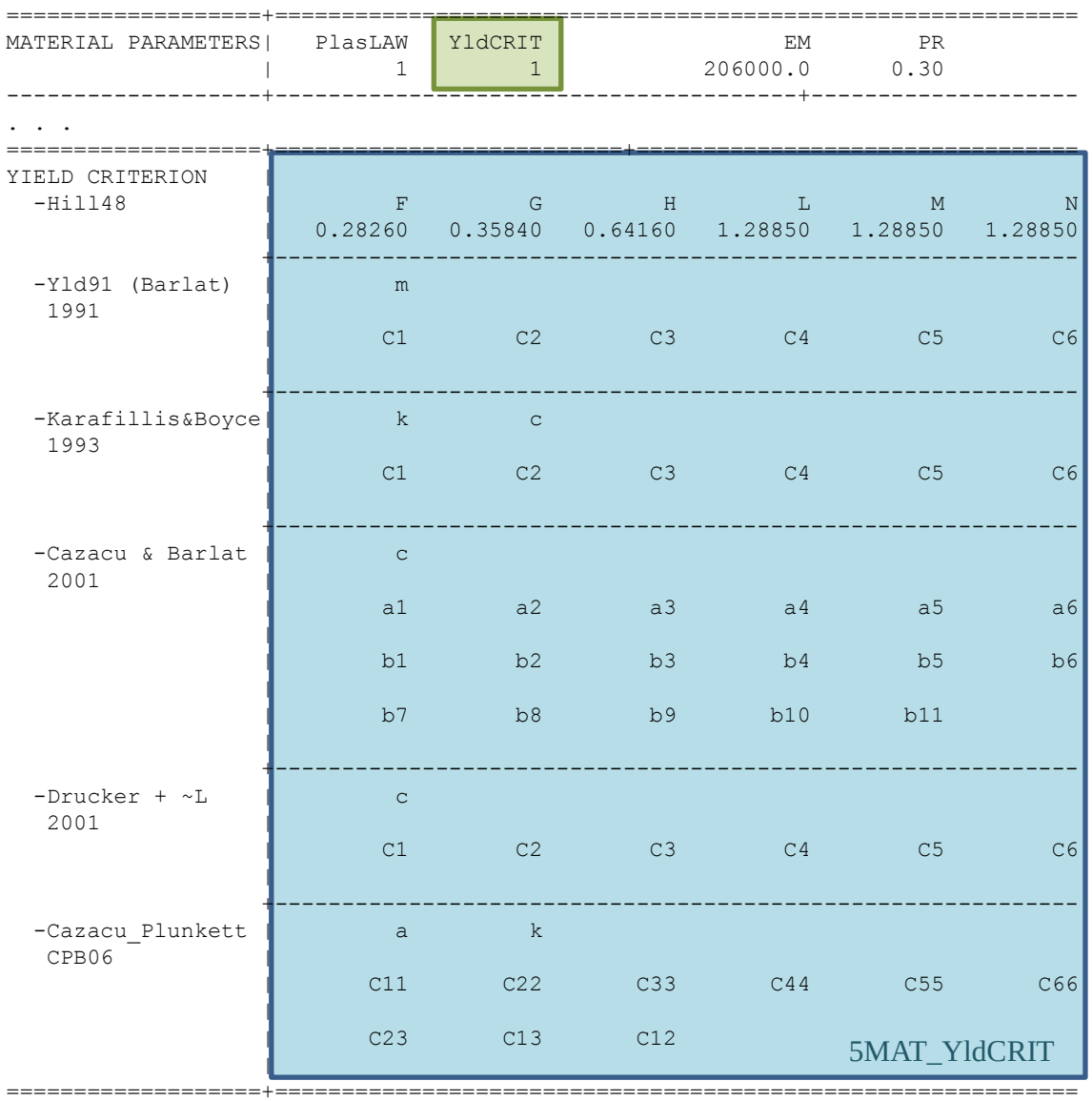


Figura 2.6. Representação da utilização do campo YldCRIT da tabela 5MAT_YldCRITparam e Param01 até Param30 da tabela 5MAT_YldCRIT, na zona de definição dos parâmetros do critério de plasticidade no ficheiro DD3_mater.dat.

2.4. 0SIMULcaseDETAILS

A tabela 0SIMULcaseDETAILS é utilizada para definir as principais características acerca do corpo deformável e da discretização adoptada. O campo MESHref permite associar uma geometria do corpo deformável a uma determinada ferramenta. Deste modo, uma vez seleccionada a ferramenta (campo toolID) é possível verificar quais as geometrias disponíveis. Para cada geometria do corpo deformável podem ser definidas diferentes opções de discretização na tabela 4MESH. Na tabela 0SIMULcaseDETAILS define-se qual é a discretização adoptada para a simulação em causa com o auxílio do campo MESHopt, que faz a ligação com a tabela 4MESH, como se mostra na Figura 2.1. As discretizações podem ser fornecidas com o auxílio de ficheiros externos, ou podem ser geradas internamente com o DD3IMP. No caso de ser utilizado o gerador interno o campo MESHopt deve tomar o valor nulo. Os campos NMATs, NPH, Nbtools e NbBCIDs da tabela 0SIMULcaseDETAILS são utilizados para indicar o número total de materiais, o número total de fases e o número total de condições de fronteira suplementares. Na prática esta informação não é utilizada no preenchimento dos ficheiros de dados de entrada, uma vez que é sempre possível verificar o número total noutras tabelas.

De modo a permitir a manipulação e alteração de discretizações fornecidas a partir de ficheiros externos, foram definidos campos auxiliares na tabela 0SIMULcaseDETAILS. Os campos Blank_dimX, Blank_dimY e Blank_dimZ são utilizados para armazenar as dimensões máximas no corpo deformável na direcção Ox, Oy e Oz. Deste modo, será possível definir factores de escala em cada direcção (campos Scalling_Ox, Scalling_Oy e Scalling_Oz) de modo a permitir que o DD3IMP altere as dimensões globais do corpo deformável. De igual modo, os campos Shifting_Ox, Shifting_Oy e Shifting_Oz são utilizados para permitir deslocar a posição inicial do corpo deformável e os campos Rotation_1, Rotation_2 e Rotation_3 para permitir a sua rotação, em relação ao sistema de eixos global.

Tal como foi referido na secção 1.1 o DD3IMP permite apenas a utilização de elementos finitos sólidos. No entanto, em geral, a geometria do corpo deformável corresponde a uma chapa plana, cuja discretização em espessura pode ser facilmente obtida por empilhamento em espessura. Assim, considera-se que a discretização fornecida com um ficheiro externo pode corresponder apenas ao plano inferior da chapa. Nesse caso, no

campo NLayers deve ser indicado o número de camadas que se pretende gerar em espessura, com o auxílio da aplicação Bi2Tri.

Por último, caso se opte pelo gerador interno e se pretenda utilizar elementos finitos sólidos hexaédricos de 12 nós, é necessário activar essa opção no campo HEX12. Uma vez que este campo pode conter diferentes opções, optou-se por o relacionar com uma tabela de opções, designada por SIMULCASEDETAILShex12. O Quadro B.13 e o Quadro B.14, apresentados no Anexo B, descrevem os campos das tabelas 0SIMULcaseDETAILS e SIMULCASEDETAILShex12, respectivamente.

Como foi mencionado anteriormente, uma das principais características a ser definida na tabela 0SIMULcaseDETAILS é a discretização a ser utilizada, i.e. os campos MESHref e MESHopt. Estes campos relacionam essa tabela com a 4MESH, como se mostra na Figura 2.1. A tabela 4MESH será descrita na subsecção seguinte.

2.4.1. 4MESH

Esta é a tabela onde é armazenada toda a informação relativa às discretizações, quer sejam geradas internamente pelo DD3IMP quer sejam definidas por um ficheiro externo, adicionado à base de dados. Tal como foi referido anteriormente, o campo MESHopt permite associar diferentes discretizações ao mesmo tipo de problema (definido com o auxílio do campo MESHref). Estas tanto podem ser discretizações geradas internamente como definidas por um ficheiro externo.

A tabela é constituída por um conjunto de campos informativos que incluem: (i) algumas notas acerca das principais características, no campo infoNOTES; (ii) uma imagem ilustrativa da discretização, no campo infoMESHimage; (iii) o número total de elementos, no campo infoNEDEF; (iv) o número total de nós, no campo infoNNDEF; (v) o número de nós dos elementos utilizados, no campo infoNELMES; (vi) o tipo de integração utilizada, no campo infoITYP; e (vii) o número de materiais presentes, no campo infoNMAT. Para além destes campos informativos, a tabela contém também um campo para a introdução do ficheiro externo, designado 2mesh_file.

Caso se opte pela geração interna da discretização, na tabela 4MESH é ainda necessário definir os parâmetros associados ao ficheiro DD3_mesh.dat. Na Figura 2.7 apresenta-se um exemplo desse ficheiro, indicando a relação dos parâmetros com os

campos de nome idêntico da tabela 4MESH. O Quadro B.15 do ANEXO B apresenta a descrição de todos os parâmetros da tabela 4MESH.

Os campos 1MESHDR, 1NELMES e 1ITYP têm opções pré-definidas pelo que se optou por criar tabelas para a sua definição, designadas MESHmeshdr, MESHnelmes e MESHityp, respectivamente. As opções disponíveis para cada um destes campos são apresentadas no ANEXO B, no Quadro B.16, Quadro B.17 e no Quadro B.18, respectivamente.

Mesh	MESHDR	NELMES	NL	NC	NH	NMAT
Parameters	1	8	20	20	2	1
Geometrical	**STH**	**SLEN**	**SHIG**	**NLAY**	**NZ**	
Definition	75.0	75.0	0.78	2	2	

4MESH

Def. of zones	NLAY	THL	NLY	MAT	ITYP
(Thickness)	1	50.0	15	1	3
	2	25.0	5	1	3

4MESH_NLAY

Def. of zones	NZ	ELZ	NCZ
(Length)	1	55.0	10
	2	20.0	10

4MESH_NZ

Figura 2.7. Representação da utilização dos campos da tabela 4MESH, 4MESH_NLAY e 4MESH_NZ para a definição do ficheiro DD3_mesh.dat.

O gerador interno permite definir diferentes zonas para a discretização, ao longo de duas direcções, uma designada NLAY e outra NZ, que correspondem aos campos 1NLAY e 1NZ da tabela 4MESH. Como se procura mostrar na Figura 2.7, para cada uma dessas zonas é necessário definir um conjunto de parâmetros. Assim, optou-se por definir duas tabelas, uma designada 4MESH_NLAY e outra 4MESH_NZ, que serão descritas nas subsecções seguintes.

2.4.1.1. 4MESH_NLAY

Esta tabela só é necessária caso seja utilizado o gerador interno. Permite a definição de diferentes zonas, com diferentes dimensões, número de elementos, materiais e tipos de integração, ao longo da dimensão global 1STH, previamente definida na tabela 4MESH. A relação entre estas tabelas é realizada com o auxílio dos campos MESHref e MESHopt, descritos na secção anterior. Cada zona é identificada no campo 1NLIN, de modo a permitir a definição dos campos 1THL, 1NLY, 1MAT e 1ITYP, para cada zona.

Estes campos estão directamente relacionados com os campos de nome idêntico do ficheiro DD3_mesh.dat, como se mostra na Figura 2.7 (THL, NLY, MAT e ITYP). O Quadro B.19 do ANEXO B apresenta a descrição destes campos.

2.4.1.2. 4MESH_NZ

Esta tabela também só é necessária caso seja utilizado o gerador interno. Neste caso, permite a definição de diferentes zonas, com diferentes dimensões e número de elementos, ao longo da dimensão global 1SLEN, previamente definida na tabela 4MESH. Também neste caso, a relação entre as tabelas é realizada com o auxílio dos campos MESHref e MESHopt, descritos na secção anterior. Cada zona é identificada no campo 1NLIN, de modo a permitir a definição dos campos 1ELZ e 1NCZ, para cada zona. Estes campos estão directamente relacionados com os campos de nome idêntico do ficheiro DD3_mesh.dat, como se mostra na Figura 2.7 (ELZ e NCZ). O Quadro B.19 do ANEXO B apresenta a descrição dos campos desta tabela.

2.5. 1INPUT

Na tabela 1INPUT define-se a informação relativa aos parâmetros numéricos e de controlo de saídas de informação no curso da simulação. Esta tabela relaciona-se com a tabela 0SIMULcase através do parâmetro caseID, como se mostra na Figura 2.1. O número de entradas da tabela é assim igual ao número de simulações pré-definidas. Para além do campo caseID, os restantes campos da tabela 1INPUT correspondem directamente aos parâmetros com o mesmo nome do ficheiro DD3_input.dat. Na Figura 2.8 apresentam-se os parâmetros do ficheiro DD3_input.dat que são definidos pelos campos homónimos da tabela 1INPUT. No Quadro B.21 apresentado no ANEXO B descrevem-se os campos desta tabela.

Os campos NSTART e MEPOPT podem tomar apenas valores pré-definidos, pelo que se optou por definir estas opções nas tabelas INPUTnstart e INPUTmepopt, respectivamente. A tabela INPUTnstart é descrita no Quadro B.22 do ANEXO B, sendo que o campo ORDER apenas é utilizado para pré-definir a ordem das opções para o campo NSTART. Os campos da tabela INPUTmepopt são apresentados no Quadro B.23, do mesmo anexo.

Simulation and Output Data	NSTART	NEND	NOUT	iGID	INC	DEV
	1	5000	0	10	51	0
Tolerances and residues	TOLEQ	TEQOUT	RAPEQ	TOLST	CUNL	dampOSS
	1.0E-02	1.0E-01	1.0E+09	1.0E-08	0.999	1.00
Maximum number of iterations	IRMAX	IEQMAX	NMAXST			
	1	50	25			
Max. Increments for each NST	DEMAX	DWMAX	DSNMAX	DSTMAX		
	0.0500	2.5000	10.0	3.0		
Rmin Strategy	RINF	RSUP	DFNMAX	DFT1MAX	DFT2MAX	
	0.0010	5.000	0.0	0.0	0.0	
Solver Parameters	LEVEL	TOLCGV				
	4	1.0E-13				
Input data, Cep	MEPOPT	iphOSS				
	1	4				

Figura 2.8. Representação da utilização dos campos definidos na tabela 1INPUT na definição do ficheiro DD3_input.dat.

2.6. 1INPUTfbp

Na tabela 1INPUTfbp define-se a informação relativa aos deslocamentos associados a saídas intermédias do tipo p[IPH]dep[FBP].ufo. Esta tabela relaciona-se com a tabela 0SIMULcase através do parâmetro caseID, como se mostra na Figura 2.1. No entanto, o número de entradas da tabela depende do número de saídas pretendidas para cada simulação pré-definida. De facto, de forma idêntica à estrutura utilizada para esta informação no ficheiro DD3_input.dat, é necessário associar a cada caseID todos os valores de deslocamento (definido no campo FBP), para os quais se pretendem gerar saídas intermédias do tipo p[IPH]dep[FBP].ufo. Estes valores devem ser introduzidos de modo sequencial para cada uma das fases da simulação (IPH).

Para além do campo caseID, os campos da tabela 1INPUTfbp são o número da saída (campo IDline) e o deslocamento correspondente (campo FBP). Todos estes campos são definidos no Quadro B.24 do ANEXO B. Na Figura 2.9 procura-se exemplificar a utilização da informação armazenada nos campos IDline e FBP na definição do ficheiro DD3_input.dat.

```
================================================================================
INTERMEDIATE OUTPUTS (1 value per line..., maximum 50 lines)
--------------------------------------------------------------------------------
    1.0000
    2.0000
   -5.0000
  -10.0000
  -15.0000
  -30.0000
  -35.0000
================================================================================
```

Figura 2.9. Representação da utilização do campo FBP da tabela 1INPUTfbp na zona de definição de saídas intermédias no ficheiro DD3_input.dat. A ordem é definida pelo campo IDline.

2.7. 2BCON

A tabela 2BCON é utilizada para definir as condições de fronteira a utilizar no modelo, durante todas as fases do processo. Esta tabela relaciona-se com a tabela 0SIMULcase através do parâmetro caseID, como se mostra na Figura 2.1. O número de entradas da tabela depende do número de condições de fronteira impostas nas simulações pré-definidas. O número de condições de fronteira associadas a cada exemplo é definido com o auxílio do campo NLIN. Para além do campo caseID e NLIN, os restantes campos da tabela 2BCON correspondem directamente aos parâmetros com o mesmo nome do ficheiro DD3_bcon.dat.

A definição das condições de fronteira é tipicamente realizada com o auxílio de planos, de acordo com a equação:

$$Ax + By + Cz = D. \tag{2.1}$$

Na tabela 2BCON são introduzidos os valores dos parâmetros A, B, C e D, nos campos homónimos. De modo a garantir a correcta selecção dos nós pretendidos é também necessário definir a tolerância admissível para a distância do nó ao plano. Esta variável corresponde ao campo ERR da tabela 2BCON. Para cada um dos planos podem ser definidas condições de fronteira para as diferentes coordenadas globais (Ox, Oy e Oz ou para todas). A selecção dessa opção é realizada no campo COORD da tabela. O tipo de condição de fronteira é definido na variável ID, que corresponde ao campo NCR_ID da mesma tabela. Por último, a cada plano podem ser associadas um conjunto de restrições. O número total de restrições é definido no campo NRES.

Na Figura 2.10 apresentam-se os parâmetros do ficheiro DD3_bcon.dat que são definidos pelos campos previamente descritos da tabela 2BCON. No Quadro B.25

apresentado no ANEXO B descrevem-se os campos desta tabela. As opções pré-definidas para o campo COORD são definidas na tabela BCONcoord, descritas no Quadro B.26 do mesmo anexo.

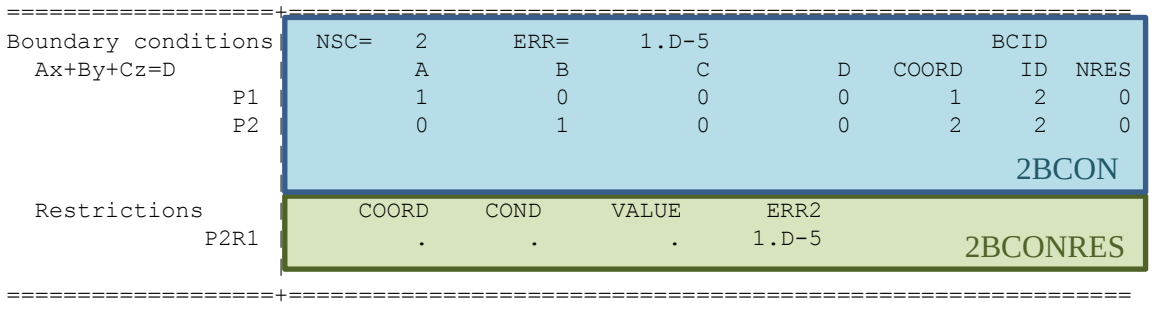


Figura 2.10. Representação da utilização dos campos definidos na tabela 2BCON e 2BCONRES na definição do ficheiro DD3_bcon.dat.

Tal como foi mencionado anteriormente, cada um dos planos pode estar sujeito a diferentes tipos de restrições, cujo número total é definido no campo NRES. As condições associadas às restrições são definidas na tabela 2BCONRES, descrita na subsecção seguinte.

2.7.1. 2BCONRES

A tabela 2BCONRES define as restrições associadas a cada uma das condições de fronteira, definidas na tabela 2BCON. Esta tabela relaciona-se com a tabela 2BCONRES através dos parâmetros caseID e NLIN, como se mostra na Figura 2.1. O número de entradas da tabela depende do número de restrições impostas às condições de fronteira previamente definidas para cada simulação. O número de restrições associadas a cada condição de fronteira, definida pelos campos caseID e NLIN, é definido com o auxílio do campo NLINRES. Os restantes campos da tabela 2BCONRES são homónimos aos parâmetros a introduzir no ficheiro DD3_bcon.dat, assinalados na Figura 2.10. O Quadro B.27 do ANEXO B apresenta a descrição destes campos.

As restrições podem ser associadas a diferentes tipos de coordenadas com o auxílio da variável COORD, que é introduzida no campo homónimo. As opções disponíveis estão para este campo são pré-definidas na tabela 2BCONcoordres, apresentada no Quadro B.28 do ANEXO B. Cada restrição é definida igualmente por um valor (campo VALUE) a associar a essa coordenada, de acordo com uma condição (campo

COND). O tipo de condições também toma valores pré-definidos, pelo que se optou por criar na base de dados a tabela COMPARISON, cujas opções são apresentadas no Quadro B.29 do ANEXO B.

2.8. 2BCONOSS

A tabela 2BCONOSS é utilizada para definir as condições de fronteira associadas à estratégia *One Step Springback* (OSS). Esta tabela relaciona-se com a tabela 0SIMULcase através do parâmetro caseID, como se mostra na Figura 2.1. O número de entradas da tabela depende do número de condições de fronteira impostas para a fase de retorno elástico, realizada com o auxílio da estratégia OSS. O número de condições de fronteira associadas a cada exemplo é definido com o auxílio do campo NLINOSS.

Na fase de OSS as condições de fronteira são adicionadas a nós específicos, seleccionados com base nas suas coordenadas globais, x, y e z. Neste caso não é definida qualquer tolerância, de modo a garantir que é sempre seleccionado o nó que apresenta as coordenadas mais próximas das pré-seleccionadas. Estas podem ser definidas em relação à configuração inicial do corpo deformável ou final, de acordo com a selecção realizada no campo XINIT. Para cada nó podem ser definidas condições de fronteira para as diferentes coordenadas globais (Ox, Oy e Oz). A selecção dessa opção é realizada no campo COORD da tabela. Assim, como se mostra na Figura 2.11, para além dos campos caseID e NLINOSS, os restantes campos são homónimos dos parâmetros a introduzir no ficheiro DD3_bcon.dat.

OSS One Step Springback	NRES= 1				
	x	y	z	COORD	Xinit
N1	0.0	0.0	0.0	3	1
N2	.	.	.	.	

Figura 2.11. Representação da utilização dos campos definidos na tabela 2BCONOSS na definição do ficheiro DD3_bcon.dat.

O Quadro B.30 do ANEXO B apresenta o resumo dos campos da tabela 2BCONOSS. As condições válidas para os campos COORD e XINIT são sumariadas nas

tabelas homónimas, COORD e XINIT, que são descritas respectivamente no Quadro B.31 e Quadro B.32 do mesmo anexo.

2.9. 3PHASE, 3PHASEheader e 3PHASElines

A tabela 3PHASE é utilizada para definir as condições iniciais associadas a cada ferramenta. Esta tabela relaciona-se com a 0SIMULcase através do campo caseID, como se mostra na Figura 2.1. Cada ferramenta é identificada no campo ENTI, pelo que o número de entradas desta tabela depende do número de ferramentas associadas a cada exemplo pré-definido. Para cada ferramenta é definido o deslocamento inicial a impor na direcção X, Y e Z, do sistema de eixos global, com o auxílio dos campos DIS0X, DIS0Y e DIS0Z. O campo TOOLname pode ser utilizado para associar um nome à ferramenta, tal como foi referido na secção 2.2. Os restantes campos desta tabela relacionam-se directamente com os parâmetros homónimos da zona de definição das condições iniciais do ficheiro DD3_phase.dat, como se mostra na Figura 2.12. O número total de ferramentas é associado ao valor máximo do campo ENTI, do caseID correspondente. O Quadro B.33 do ANEXO B apresenta a descrição de todos estes campos da tabela.

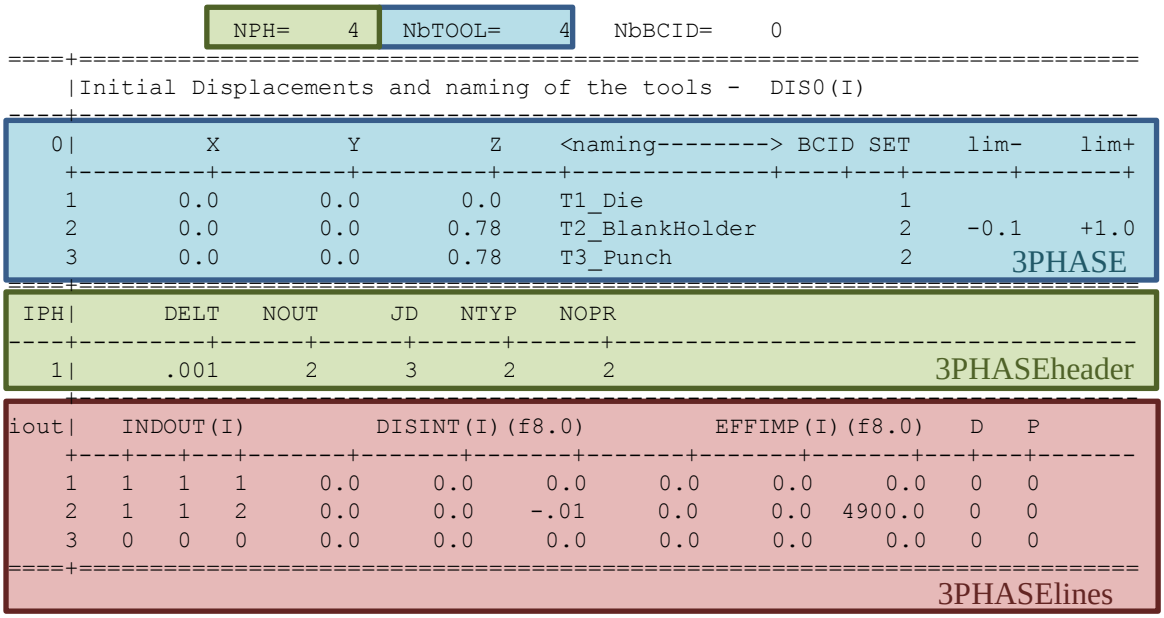


Figura 2.12. Representação da utilização dos campos definidos nas tabelas 3PHASE, 3PHASEheader e 3PHASElines na definição do ficheiro DD3_phase.dat.

Para cada fase do processo é necessário definir o tamanho do incremento, qual a ferramenta e a direcção de controlo e o tipo de controlo, como se mostra na Figura 2.12. Estas variáveis são definidas na tabela 3PHASEheader, que se relaciona com a tabela 3PHASE através do campo caseID. Cada fase é identificada no campo IPH, pelo que o número de entradas desta tabela depende do número de fases associadas a cada exemplo pré-definido. Os restantes campos desta tabela relacionam-se directamente com os parâmetros homónimos da zona de definição de fase no ficheiro DD3_phase.dat, como se mostra na Figura 2.12. O número total de fases é associado ao valor máximo do campo IPH, do caseID correspondente. O Quadro B.34 do ANEXO B apresenta a descrição de todos estes campos da tabela.

A direcção de controlo é pré-definida de acordo com as opções disponibilizadas na tabela COORD, descrita no Quadro B.31 do ANEXO B. As opções associadas ao tipo de controlo da fase, definidas no campo NTYP, são definidas na tabela PHASEntyp, como se mostra no Quadro B.35. A condição de controlo que dita o final da fase, campo NOPR, também tem condições pré-definidas na tabela PHASEnopr, como se mostra no Quadro B.36.

Finalmente, para cada fase é necessário definir o estatuto, o deslocamento e o valor de força imposta, para cada ferramenta. Estes parâmetros são definidos na tabela 3PHASELINES, que se relaciona com a tabela 3PHASE através dos campos caseID e ENTI, mas também com a tabela 3PHASEheader através do campo IPH. Os restantes campos desta tabela relacionam-se directamente com os parâmetros homónimos da zona de definição de fase no ficheiro DD3_phase.dat, como se mostra na Figura 2.12. Assim, estatuto de cada ferramenta, em cada direcção do sistema de eixos global, é definido nos campos INDOUTx, INDOUTy e INDOUTz. O deslocamento de cada ferramenta, em relação a cada direcção do sistema de eixos global, é definido nos campos DISINTx, DISINTy e DISINTz. Finalmente, o valor da força imposta à ferramenta, em cada direcção do sistema de eixos global, é definido nos campos EFFIMPx, EFFIMPy, EFFIMPz. O Quadro B.37 do ANEXO B apresenta da descrição dos campos da tabela 3PHASELINES.

Como o estatuto da ferramenta, em cada direcção, tem opções pré-definidas optou-se por organizar essas hipóteses na tabela PHASEindout, descrita no Quadro B.38 do ANEXO B.

2.10. 6CONTACTSET

A tabela 6CONTACTSET é utilizada para definir zonas do corpo deformável como potenciais de contacto (*set*) com uma determinada ferramenta. Esta tabela relaciona-se com a tabela 0SIMULcase através do campo caseID, como se mostra na Figura 2.1. O número total de entradas da tabela depende do número de zonas de contacto associados a cada exemplo, que é definido com o auxílio do campo NCONTSET.

Cada zona pode ser definida com um plano, tal como definido na equação (2.1). Pode também ser definida com um cilindro, de acordo com:

$$Ax^2 + By^2 + Cz^2 = D^2, \tag{2.2}$$

ou um cilindro de base elíptica:

$$\left(\frac{x}{A}\right)^2 + \left(\frac{y}{B}\right)^2 + \left(\frac{z}{C}\right)^2 = 1. \tag{2.3}$$

Na tabela 6CONTACTSET são introduzidos os valores dos parâmetros A, B, C e D, nos campos homónimos. O tipo de zona, plana ou cilíndrica, é definido com o auxílio do campo STYPE. A Figura 2.13 apresenta a zona do ficheiro DD3_contact.dat a que correspondem os campos definidos na tabela 6CONTACTSET. O Quadro B.39 do ANEXO B apresenta a descrição dos campos desta tabela. As opções disponíveis para o campo STYPE encontram-se agrupadas na tabela CONTACTstype, que é descrita no Quadro B.40 do mesmo anexo. Nesta tabela o campo ORDER apenas é utilizado para pré-definir a ordem das opções.

CONTACT SETS						
	NSET	2				Pressure
ISET Ax+By+Cz=D	X	A	B	C	D	Tool
1	P	0	0	1	L	
2	P	0	0	1	H	

Figura 2.13. Representação da utilização dos campos definidos na tabela 6CONTACTSET na definição do ficheiro DD3_contact.dat.

2.11. 6FRICTION

A tabela 6FRICTION é utilizada para definir as condições de contacto com atrito, bem como os parâmetros numéricos para o lagrangiano aumentado. Esta tabela

relaciona-se com a tabela 0SIMULcase através do campo caseID, como se mostra na Figura 2.1. O número total de entradas da tabela depende do número de zonas com diferentes condições de contacto, associadas a cada exemplo, que é definido com o auxílio do campo NLIN.

De facto, as condições de contacto com atrito podem ser definidas de forma global, por *set* de contacto ou por superfície das ferramentas, de acordo com o campo GSP. Caso o utilizador opte por introduzir as condições de contacto específicas para um determinado *set* de contacto (campo GSP=S), a correspondência com a sua definição na tabela 6CONTACTSET é feita com o auxílio do campo SetNUM. Caso se opte por definir as condições de contacto para um determinado conjunto de superfícies (campo GSP=P), a associação com o seu número é feita recorrendo os campos FPatch e LPatch, que correspondem ao primeiro e último número do conjunto sequencial de superfícies.

O campo FriCoef corresponde ao valor do coeficiente de atrito a utilizar em cada zona, que é representado pelo parâmetro MU no ficheiro DD3_Contact.dat, como se mostra na Figura 2.14. Os restantes campos da tabela 6FRICTION correspondem aos parâmetros homónimos do ficheiro DD3_contact.dat, como se descreve no Quadro B.41 do ANEXO B.

```
==================+============================================================
FRICTION          |
    xxx : xxx     |        MU      CPen   IPenvar     CPenD
G                 |     0.144   20600.0
S     1           |     0.100
P    10 :  15     |     0.020
- - - - - - - - - -+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Figura 2.14. Representação da utilização dos campos definidos na tabela 6FRICTION na definição do ficheiro DD3_contact.dat.

Os parâmetros numéricos para o lagrangiano aumentado são definidos apenas uma vez, de forma global. Como podem ser utilizadas diferentes estratégias para o parâmetro de penalidade, definidas com o auxílio do campo IPenvar, as opções disponíveis são resumidas na tabela FRICTIONipenvar, descrita no Quadro B.42. As opções disponíveis para o campo GSP são definidas na tabela FRICTIONgsp e são resumidas no Quadro B.43 do ANEXO B.

2.12. Estrutura dos ficheiros de entrada

Tal como foi referido na subsecção 1.1.1 o *layout* dos ficheiros de entrada para o DD3IMP é rígido. De modo a preservar este *layout*, optou-se por definir a estrutura de cada ficheiro na forma de máscara, sobre a qual são indicados os campos a utilizar para o preenchimento de cada parâmetro. As máscaras dos ficheiros de entrada são definidas na tabela _MASKfiles. No entanto, como se referiu na secção 2.3, as máscaras para as zonas do ficheiro DD3_mater.dat correspondentes às leis de encruamento e aos critérios de plasticidade são definidas nas tabelas 5MAT_HDparam e 5MAT_YldCRITparam. De facto, para cada ficheiro de entrada foram identificadas zonas distintas, de modo a permitir gerar os ficheiros com a estrutura pretendida, com base na informação introduzida na base de dados. Nas tabelas 5MAT_HDparam e 5MAT_YldCRITparam a máscara é pré-definida em função da lei de encruamento e do critério de plasticidade seleccionada, respectivamente. Na tabela _MASKfiles as diferentes zonas de cada ficheiro de entrada são identificadas com o auxílio do campo key. O Quadro B.44 do ANEXO B apresenta a descrição dos campos desta tabela. Na Figura 2.15 apresenta-se, a título de exemplo, a estrutura definida para a máscara do ficheiro DD3_bcon.dat. Neste exemplo, cada máscara correspondente às key 2, 4 e 6 são preenchidas tantas vezes quantas o número de linhas NLIN da tabela 2BCON, NLIN da tabela 2BCONRES e NLINOSS da tabela 2BCONOSS, respectivamente.

A associação entre as diferentes variáveis necessárias à construção dos ficheiros de entrada e os campos da base de dados é realizada na tabela ZFIELDS, com o auxílio do campo ATn. A tabela foi construída de modo a garantir que este campo é único para cada campo de cada tabela principal utilizada na base de dados. Deste modo, ao indicar na máscara o parâmetro @ATn (ver Figura 2.15), é identificado de forma inequívoca um campo de uma determinada tabela.

A estrutura global da tabela ZFIELDS é apresentada no Quadro B.45 do ANEXO B. Esta tabela releva-se de enorme importância para a ligação entre a base de dados e a aplicação interactiva pois nela é também definido, para cada campo: (i) o formato de escrita do campo (campo FORMAT); (ii) o nome a utilizar na aplicação interactiva (campo GUIname); e (iii) a descrição do parâmetro (campo DESCfield). De qualquer modo importa realçar, que para um utilizador que apenas pretenda introduzir

novas simulações na base de dados a informação das tabelas _MASKfiles e ZFIELDS não é necessária.

_MASKfiles

key	DD3_bcon
1	@54 @55 ==========+========== Boundary conditions| NSC= NNN ERR=@202 BCID Ax+By+Cz=D | A B C D COORD ID NRES
2	Px |@203 @204 @205 @206 @207 @208 @209
3	Restrictions | COORD COND VALUE ERR2
4	PxRX | @243 @244 @245 @246
5	==========+========== OSS One Step | NRES=NNNN Springback | x y z COORD Xinit
6	Nx |@222 @223 @224 @225 @226
7	==========+========== Remarks: COND: 0 -> .EQ. | COORD: 1 -> X | 1 -> .GE. | 2 -> Y | -1 -> .LE. | 3 -> Z | 4 -> all | 5 -> radius | Xinit: 1 -> yes 0 -> no BCID: = 2 -> symmetry condition >10 -> ID group of DOFs

Figura 2.15. Definição da máscara para o ficheiro DD3_bcon.dat de acordo com a tabela _MASKfiles.

3. INTERFACE GRÁFICA

A aplicação gráfica para o DD3LT foi desenvolvido na plataforma *Microsoft Visual Studio 2010 Professional* em *Visual C++*. O *Microsoft Visual Studio 2010* é um ambiente de desenvolvimento integrado IDE[3] que permite a programação de aplicações em várias linguagens (como VB.NET, C#, VC++, …). Para facilitar a sua utilização, existem ferramentas como *wizards*, que permitem gerar automaticamente o código que constitui o esqueleto para uma aplicação, e ferramentas que auxiliam na depuração do código desenvolvido. Uma outra parte integrante do IDE acima mencionado são as várias bibliotecas de funções, entre as quais consta a biblioteca de funções *Microsoft Foundation Class* (MFC) que fornece um invólucro orientado a objectos sobre grande parte do sistema operativo *Microsoft Windows 32 bit* (Win32) e aplicações de interface *Component Object Model* (COM). Embora possa ser utilizada para criar aplicações de *desktop* muito simples, a sua utilidade real só é revelada perante a necessidade de desenvolver interfaces de utilizador mais complexas, envolvendo controlos múltiplos. Esta biblioteca está documentada *online* na *Microsoft Developer Network* (MSDN), onde é feita a cobertura de todas as classes, funções globais e macros que constituem a MFC 10 presente no *".NET Framework 4.0"* (Microsoft, 2012). A documentação para cada classe inclui um resumo, um sumário categórico e tópicos de cada função membro (método), dados membro (propriedades).

O código da aplicação DD3LT foi implementado respeitando o sistema de notação húngaro, visando facilitar o reconhecimento do tipo de cada variável e tornar mais simples a leitura do código. Este sistema de notação consiste na utilização de prefixos nos nomes das variáveis de forma a codificar informação sobre o tipo de dados da variável (foi utilizado conjuntamente com o sistema de capitalização *camel-casing*). Perante a necessidade de alteração do tipo de variável, este sistema de notação implica uma alteração do nome da variável em todas as referências feitas a esta, motivo pelo qual é actualmente

[3] Um ambiente de desenvolvimento integrado (IDE) é uma aplicação que oferece amplas instalações para o desenvolvimento de software. Um IDE normalmente consiste num: (1) editor; (2) um ambiente para compilação; e (3) um depurador.

desaconselhado pela Microsoft. No entanto este sistema de notação permite uma fácil interpretação do código e é essencialmente por este motivo que foi adoptado.

No código da aplicação DD3LT as "Macros" são sempre definidas com letras maiúsculas, as funções e os nomes estruturados começam sempre com letra maiúscula, os nomes de objectos começam com uma ou mais letras maiúsculas, em acordo com o sistema de capitalização *Pascal-casing*. Na atribuição de nomes a funções tentou-se fazer com que o nome da função seja auto explicativo da tarefa nela desempenhada (por exemplo a função *CDD3BoundCond::WriteToFile()* que grava o conteúdo do objecto *CDD3BoundCond* para um ficheiro), tentado evitar abreviações de nomes sempre que possível (cuja excepção é, por exemplo, o caso de *CDD3BoundCond* cuja alternativa seria *CDD3BoundaryConditions*).

Perante a natureza finita da mente humana, no desenvolvimento de *software* é essencial saber lidar com a complexidade. Existem essencialmente dois métodos que permitem ao ser humano lidar com a complexidade (Milewski, 2001):

- Método *divide-and-conquer*;
- Método *abstraction*;

O método *divide-and-conquer* é muito conhecido entre programadores de Fortran e Pascal, consistindo na subdivisão de um problema em conjuntos de problemas mais pequenos, sendo estes posteriormente subdivididos até que, eventualmente, a solução de cada problema seja trivial.

O método *abstraction* consiste em subtrair características não essenciais. Pense-se, por exemplo, na quantidade de características que são possíveis subtrair a um carro (i.e. faróis, pintura) sem que este deixe de ser reconhecível como um carro. Uma abstracção é o produto deste processo, é um substantivo super-categórico para todos os conceitos subordinados, no qual são associadas características e funções base.

A divisão e a abstracção interligam-se naquilo que é chamado o paradigma *divide-and-abstract*. Um sistema complexo pode ser visualizado como uma grande rede de nós interligados. Essa rede pode ser dividida em alguns subconjuntos de nós a que chamamos objectos. Uma boa divisão faz com que existam o mínimo de ligações inter-objecto possíveis. Para descrever os objectos resultantes utilizamos abstracções. Esta é a filosofia base da programação orientada a objectos, que foi utilizada no desenvolvimento da plataforma DD3LT.

Neste capítulo procura-se descrever a estrutura global da aplicação construída, em particular as alterações realizadas às funções globais e macros que constituem a MFC 10. Devido à complexidade do tema em análise, recomenda-se a consulta de alguma bibliografia de base (Horton, 2010; Björnander, 2008; Prosise, 1999; Schach, 2011) antes da sua leitura. O objectivo deste capítulo não é explicar como desenvolver uma aplicação mas orientar os futuros desenvolvimentos da aplicação DD3LT. Neste contexto, nas secções seguintes procura-se descrever os principais procedimentos para o acesso à informação disponível na base de dados, visualização e alteração.

3.1. Document/View

O *wizard* de criação de aplicações MFC cria por defeito um esqueleto de uma aplicação com uma classe *document* e uma classe *view*. Na biblioteca MFC a gestão de dados é separada nestas duas classes. Os dados são guardados na classe *document* que também coordena a actualização de múltiplas vistas dos dados. Na classe *view* é feita a exibição dos dados que constam da classe *document*, bem como a gestão da interacção do utilizador com os dados (ex. selecção, edição, …). Este tipo de arquitectura denomina-se *Document/View*. O objecto documento MFC é responsável pela leitura e escrita dos dados para um armazenamento persistente, podendo também servir de interface a dados armazenados noutro tipo de suporte, como por exemplo numa base de dados. O objecto vista exibe dados a partir do documento, e comunica ao documento eventuais alterações dos dados. Este tipo de arquitectura tem vantagens persuasivas como a possibilidade de exibir vistas múltiplas dos mesmos dados em simultâneo, tal como uma vista gráfica e uma vista estruturada numa tabela. O modelo *Document/View* permite que existam separadamente um objecto *view* para cada vista sobre os dados, enquanto o código comum a todas as vistas (sub-rotinas de tratamento de dados, etc.) pode residir no documento, que actualiza as vistas sempre que os dados são alterados. Estas são as classes da plataforma MFC mais visíveis a ambos utilizador e programador, pelo que a maior parte do trabalho de desenvolvimento consiste em escrever as classes *document* e *view*.

A implementação do modelo *Document/View* na libraria MFC separa os dados em si da sua visualização e das operações do utilizador sobre os dados, como se mostra na Figura 3.1. Todas as alterações sobre os dados são feitas através do documento. A vista utiliza o documento como interface para aceder aos dados e para os actualizar.

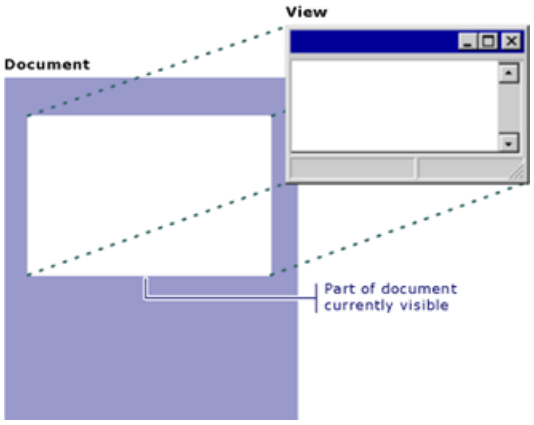


Figura 3.1. Representação esquemática da ligação entre o documento e a vista (Microsoft, 2012).

O processo de desenvolvimento de aplicações MFC consiste em especializar as classes MFC originais de forma a poder alterar, adicionar ou suprimir comportamentos nessas mesmas classes, algo comum em programação orientada a objectos. A especialização das classes é realizada através da criação de novas classes, que herdam a classe que se pretende especializar. Passa então a ser possível redefinir as funções virtuais cujo comportamento se pretende alterar, e adicionar novas funções e variáveis. Uma vez que estas novas classes herdam a classe especificada, todas as características inalteradas da classe de base (herdada) passam a estar disponíveis na classe especializada acrescendo ainda todas as funcionalidades adicionadas a essa nova classe. Desta forma, as funcionalidades acrescidas da classe derivada (classe que herda a classe de base) são integradas na arquitectura das classes MFC.

Os documentos, as vistas associadas e as molduras das vistas são criados por um *document template* (*CMultiDocTemplate*), que é responsável por criar e gerir todos os documentos de um dado tipo. Desta forma na criação de cada *document template* é necessário associar a respectiva classe *document*, *child window* e classe *view*. Desta forma, na criação de cada *CMultiDocTemplate* é necessário associar a respectiva classe derivada de *CDocument*, a classe derivada de *CMDIChildWndEx* e a classe derivada de *CView*. É necessário ainda atribuir um identificador de recursos (ID) a cada *template*. Este identificador especifica os recursos utilizados por um dado tipo de documento.

Perante a necessidade de exibir vistas múltiplas sobre os mesmos dados, na aplicação DD3LT passa a existir um *document template* para cada vista (como se mostra

na Tabela 3.1), encontrando-se todas as vistas ligadas ao mesmo documento (que retêm os dados), partilhando assim a mesma classe derivada da classe *CDocument* entre todos os *document templates*.

Tabela 3.1. Relação entre o *document template,* as vistas e os ID do recurso utilizados na aplicação DD3LT.

Nome do Template	Vista - Classe Derivada	Vista - Classe de Base	ID do Recurso
pDocTempInput	CTextEditInputView	CEditView	IDR_INPUT_VIEW
pDocTempBCon	CTextEditBConView	CEditView	IDR_BCON_VIEW
pDocTempPhase	CTextEditPhaseView	CEditView	IDR_PHASE_VIEW
pDocTempContact	CTextEditContactView	CEditView	IDR_CONTACT_VIEW
pDocTempMesh	CTextEditMeshView	CEditView	IDR_MESH_VIEW
pDocTempTool	CTextEditToolView	CEditView	IDR_TOOL_VIEW
pDocTempMater	CTextEditMaterView	CEditView	IDR_MATER_VIEW
pDocTempVTK	CVTKView	CView	IDR_VTK_VIEW
pDocTempChartVtk	CVTKGraphView	CView	IDR_CHART_VIEW

Todos os dados provenientes da base de dados pertencem à classe *CDD3ParamDoc*, que é derivada da classe *CDocument*. Esta classe foi criada para encapsular os dados que serão utilizados pela aplicação DD3LT. A classe *CChildFrame* deriva de *CMDIChildWndEx* e encapsula as funcionalidades da moldura que suporta cada vista da aplicação. Uma vez que as molduras que suportam as vistas do DD3LT devem exibir comportamentos idênticos, esta classe é partilhada por todos os *document templates*.

Uma vez definida a moldura é necessário proceder à definição das vistas. Na subsecção seguinte procura-se elucidar a forma como é criada uma nova vista.

3.1.1. View

Para facilitar a explicação acerca da criação de uma nova vista, opta-se por fazer uma descrição sucinta da implementação da vista do ficheiro "DD3_bcon.dat" no DD3LT. O processo inicia-se pela criação de uma nova classe derivada da classe MFC *CEditView*, que por sua vez fornece suporte à visualização e edição de texto. A criação desta nova classe à qual se deu o nome de *CTextEditBConView*, pode ser feita manualmente ou recorrendo ao utilitário *Class Wizard*. Uma vez criada a classe é

necessário adicionar e alterar funcionalidades de forma a obter o comportamento desejado. Como referido anteriormente, as alterações são feitas através da redefinição das funções virtuais. No início da visualização desta vista é necessário mudar o nome da janela, a fonte do texto que será exibido e por fim carregar o texto em si. Existe na classe de base um método virtual público com o nome *OnInitialUpdate()* que é chamado pela *framework* no final da instanciação do objecto *CEditView*. Como tal basta redefinir este método para adicionar as propriedades acima descritas. Note-se estas operações são realizadas assumindo que nesta fase da execução o conteúdo do ficheiro "DD3_bcon.dat" se encontra disponível na instância do objecto *CDD3ParamDoc* sob a forma de vector de caracteres.

Pretende-se também que o conteúdo deste ficheiro seja actualizado na respectiva vista sempre que os respectivos dados sofram modificações. Existe na classe base um método virtual denominado *OnUpdate()*, que possibilita a actualização dos dados exibidos. Este método é invocado pela classe MFC *CDocument* (classe de base para o objecto *CDD3ParamDoc*) através do método *CDocument::UpdateAllViews()*, que será chamado sempre que terminem operações de modificação de dados.

As alterações do texto contido na vista do ficheiro devem também ser efectuadas ao nível dos dados de referência, localizados na instância do objecto *CDD3ParamDoc*. A classe base *CEditView* possibilita a interacção e exibição de texto através da inclusão de um controlo *CEdit*. Este controlo permite que, sempre que o utilizador realize uma acção de modificação de dados, seja desencadeada uma notificação sobre a forma de uma mensagem assíncrona *EN_CHANGE*. Esta mensagem destina-se à janela que detém a posse sobre este controlo, permitindo que as alterações sejam devidamente processadas. Dito isto, para actualizar os dados de referência é necessário implementar um método *OnEnChange()* que execute uma cópia do texto exibido para o vector de caracteres que detêm o conteúdo do ficheiro "DD3_bcon.dat", e associar a execução deste método à mensagem *EN_CHANGE* no mapa de mensagens da classe *CTextEditBConView*.

3.1.2. View - Document template

Uma vez implementada a especialização da classe *CEditView* é necessário criar o identificador de recurso (*resource* ID) para o novo *document template*. O indentificador de recurso identifica a *string* a utilizar pelo documento. Esta *string* consiste num conjunto

de sete *strings* separadas pelo caracter *new line* (‘\n’), que descrevem o tipo de documento, e que se encontram ordenadas pela seguinte ordem:

- Título do *MainFrame*;
- Título do *ChildFrame*;
- Título do tipo de documento;
- Descrição do filtro do documento;
- Extensão do ficheiro do documento;
- Identificação do tipo de ficheiro no *regedit*;
- Descrição do tipo de ficheiro no *regedit*;

A plataforma MFC cria um ícone para cada *document template* cuja string associada contenha as seguintes informações:

- Título do tipo de documento;
- Descrição do filtro do documento;
- Extensão do ficheiro do documento;

Para que não existam tipos de documento múltiplos no DD3LT, apenas um *document template* possuirá a informação acima mencionada.

O indentificador de recurso IDR_BCON_VIEW identifica a posição da *string* “\nBCONVIEW\n\n\n\nDD3LT.Document\nDD3LT.Document” na *string table* do gestor de recursos. Esta *string* descreve as propriedades do *document template*, designado por *pDocTempBCon*, cuja instanciação é feita na redefinição do método virtual *InitInstance()*, na classe *CDD3ParamApp*, que deriva da classe MFC *CWinAppEx*. A *framework* MFC invoca este método na inicialização da aplicação DD3LT, que é iniciada ao instanciar o objecto *CDD3ParamApp*. O objecto *pDocTempBCon* necessita de ser instanciado numa fase anterior à instanciação da *MainFrame* da aplicação.

3.1.3. Document

Nas subsecções anteriores, assume-se que a visualização do conteúdo relativo ao ficheiro “DD3_bcon.dat” é feita numa fase em que os dados se encontram disponíveis na respectiva instância do objecto documento. Nesta subsecção, procura-se mostrar a forma como os dados estão estruturados no objecto *CDD3ParamDoc*. Para tal, recorre-se novamente a uma descrição da implementação do ficheiro “DD3_bcon.dat” na aplicação DD3LT.

Com o objectivo de encapsular os dados e as operações sobre os dados do ficheiro "DD3_bcon.dat", foi criada uma classe *CDD3BoundCond* que deriva da classe MFC *CObject*. Esta nova classe é integrada no objecto *CDD3ParamDoc* sob a forma de um atributo. Na Figura 3.2 mostra-se a relação de associação entre as classes *CDD3ParamDoc* e *CDD3BoundCond*, bem como as respectivas heranças.

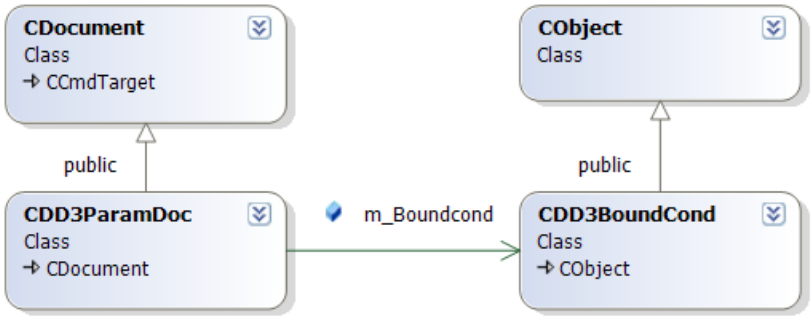


Figura 3.2. Esquema UML[4] representativo da relação de associação entre as classes *CDD3ParamDoc* e *CDD3BoundCond*, bem como as respectivas heranças.

Foram também criadas as classes auxiliares *CSpringRestrict*, *CSymCond* e *CSymRestrict*. A classe *CSpringRestrict* fornece um invólucro sobre os dados relativos a cada condição definida para a fase de retorno elástico com a estratégia *One Step Springback*. De forma análoga, a classe *CSymRestrict* fornece um invólucro sobre os dados relativos à uma dada restrição, definida para uma condição de simetria. A classe *CSymCond*, possui um vector para a definição destas restrições (instâncias *CSymRestrict*), fazendo o encapsulamento dos dados de cada condição de simetria. Na classe *CDD3BoundCond* existe um vector de instâncias da classe *CSymCond* e um vector de instâncias da classe *CSpringRestrict*, de forma a organizar as condições de simetria e as restrições para a fase de retorno elástico, respectivamente. A Figura 3.3 apresenta o esquema representativo da relação de composição entre a classe *CDD3BoundCond* e as classes *CSpringRestrict*, *CSymCond*, *CSymRestrict*,

[4] A UML (*Unified Modeling Language*) é uma linguagem gráfica padrão para a elaboração da estrutura de projectos normalmente baseados na programação com objectos.

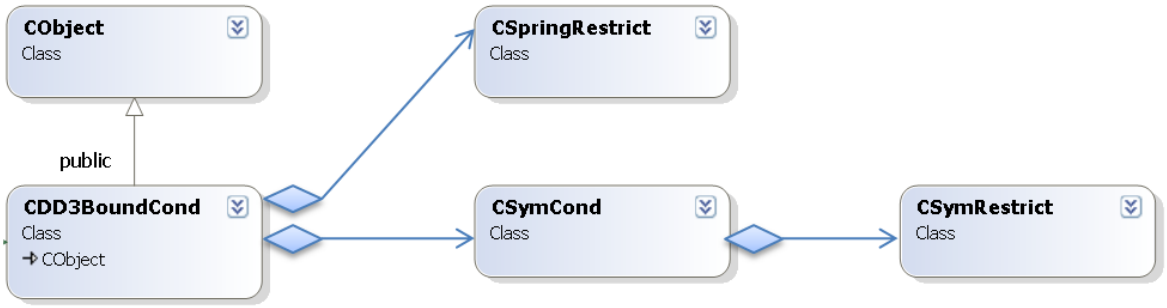


Figura 3.3. Esquema UML representativo da relação de composição entre a classe *CDD3BoundCond* e as classes *CSpringRestrict*, *CSymCond*, *CSymRestrict,* bem como as respectivas heranças.

Para além de reterem os dados relativos ao ficheiro "DD3_bcon.dat", é nas classes *CSymCond*, *CSymRestrict* e *CSpringRestrict* que é feita a inicialização das classes que encapsulam os parâmetros: *CSingleParameter<TYPE>* e *CMultiParameter<TYPE>*. Por sua vez, as classes *CSingleParameter<TYPE>* e *CMultiParameter<TYPE>* realizam a tarefa de reunir a informação relativa a cada parâmetro a inicializar, e permitem realizar algumas tarefas de tratamento e gestão de dados. Estas classes serão descritas na subsecção 3.3.

As classes *CSymCond*, *CSymRestrict* e *CSpringRestrict* são funcionalmente análogas, pelo que as suas implementações são muito semelhantes. Nas estruturas destas classes é possível encontrar a instanciação de uma *string*, que guarda a máscara que detém o formato da representação dos parâmetros em texto (ver secção 2.12), a instanciação de uma classe auxiliar para cada parâmetro, e ainda a instanciação de uma lista de apontadores para cada parâmetro. Nas estruturas destas classes existem ainda os métodos: *GetMask()*; *InitializeParameters()*; *EncodeFileData()*; construtor e o destrutor. O método *GetMask()* copia a máscara correcta, da base de dados para a *string* acima mencionada. No método *InitializeParameters()* é definido o nome sobre o qual se encontra cada parâmetro na base de dados bem como os nomes das tabelas que detêm esses mesmos parâmetros. Este método permite fazer a inicialização dos parâmetros. O método *EncodeFileData()* é responsável pela conversão dos valores actuais de cada parâmetro numa única *string* que será posteriormente adicionada ao texto que constitui o ficheiro de entrada. O construtor da classe executa a sequência correcta dos membros no momento da instanciação. O destrutor da classe faz a dealocação das variáveis dinâmicas utilizadas pela classe durante a execução.

O objecto *CDD3BoundCond* fornece um invólucro sobre o ficheiro de entrada DD3_bcon.dat. Neste objecto são integradas as classes que guardam os dados (*CSymCond*, *CSymRestrict* e *CSpringRestrict*), bem como métodos que operam sobre eles. Analogamente às classes acima descritas, neste objecto é também feita a inicialização de parâmetros cuja ocorrência não justifica a criação de classes auxiliares. Como tal também é possível encontrar neste objecto os métodos: *GetMask()*; *InitializeParameters()*; *EncodeFileData()*;. As funcionalidades destes métodos são as mesmas que as anteriormente descritas, com a excepção do método *EncodeFileData()* que neste caso codifica todos os parâmetros do ficheiro de DD3_bcon.dat na sua representação em texto. Adicionalmente foram criados os métodos *GetParamValFromDb()* e *WriteToFile()*. No método *GetParamValFromDb()* recorre-se a classes derivadas do objecto *CRecordSet* (criadas para encapsular os dados presentes na base de dados) para definir o valor de cada parâmetro do ficheiro "DD3_bcon.dat", de acordo com os dados armazenados para o problema seleccionado pelo utilizador. O método *WriteToFile()*é uma adaptação do método da *CObject::Serialize(CArchive& ar)*, permitindo gravar o ficheiro "DD3_bcon.dat" na localização definida para o parâmetro de entrada *CArchive*.

3.2. Encapsulamento da base de dados

O acesso aos parâmetros contidos na base de dados é feito por intermédio de duas classes MFC. A primeira é a classe *CDatabase* que proporciona um invólucro em torno do *driver* que comunica com a base de dados (neste caso ODBC32.dll). A segunda é a classe *CRecordset* que proporciona o encapsulamento sobre as instruções SQL (*Structured Query Language*) utilizadas para aceder à base de dados.

A classe *CRecordset* possibilita a sincronização de variáveis com os campos de *querys* submetidas às tabelas contidas na base de dados, bem como a navegação entre os registos e a aplicação de filtros na listagem do conteúdo destas tabelas. Na aplicação DD3LT existe uma classe derivada da classe *CRecordset* para cada tabela presente na base de dados, com implementações muito semelhantes. Com o objectivo de elucidar a forma como são implementadas estas especializações, retoma-se o exemplo do ficheiro "DD3_bcon.dat". De seguida, é feita uma descrição sucinta da classe *CBoundCondRecordSet* (derivada da classe MFC *CRecordset*) que fornece à aplicação

DD3LT um interface com a tabela "2BCON" presente na base de dados. Na implementação da classe *CBoundCondRecordSet*, foram criadas variáveis membro em formatos de dados coerentes com os campos da tabela "2BCON" da base de dados. A sincronização destas variáveis com os campos da tabela é feita com recurso às macros RFX. Estas macros são utilizadas na redefinição do método virtual *CBoundCondRecordSet::DoFieldExchange()*, que é invocado pela *framework* sempre que existe uma mudança da posição do cursor do registo. A tabela disponibilizada na instância da classe *CBoundCondRecordSet* é o resultado da *query* SQL, definida na reimplementação do método virtual *CBoundCondRecordSet::GetDefaultSQL()*. À query SQL podem ser adicionados parâmetros de filtro, definidos na variável *CRecordset::m_strFilter*, e critérios de ordenação definidos na variável *CRecordset::m_strSort*. A base de dados à qual é realizada a *query* é indicada pelo objecto *CDatabase*, apontado pelo membro *CRecordset::m_pDatabase*, que é definido no momento da instanciação.

A classe *CDD3DataBase* foi criada para fornecer um invólucro à ligação do DD3LT com a base de dados. Nesta classe é inicialmente instanciado um objecto *CDatabase* sob a forma de membro *CDD3DataBase::m_db*. Este membro é inicializado no método *CDD3DataBase::ConnectDB()* que estabelece a ligação à base de dados definida em *CDD3DataBase::GetConectionString()*. Esta conexão à base de dados é posteriormente utilizada por todos os objectos derivados de *CRecordset* (membros da classe *CDD3DataBase*). Foi ainda criado o método *CDD3DataBase::SetRecordsFilters()* com o objectivo de fixar os filtros de cada classe derivada de *CRecorset*, de forma a isolar os dados relativos ao problema seleccionado pelo utilizador.

3.3. Generalização dos dados

Como referido na subsecção 3.1.3, as classes *CSingleParameter<TYPE>* e *CMultiParameter<TYPE>* reúnem a informação necessária às operações sobre os parâmetros de entrada do DD3IMP, e fornecem à aplicação DD3LT uma camada de abstracção sobre o tipo de dados destes parâmetros. Estas classes foram construídas utilizando *templates* de classes. Na linguagem c++, estes *templates* são classes ou funções que podem operar tipos de dados genéricos. Esta funcionalidade permite criar funções

template cuja funcionalidade pode ser adaptada a diferentes tipos de dados sem ter que repetir o código para cada tipo. Quando o compilador encontra uma chamada para uma função *template*, utiliza-o para gerar automaticamente uma função, substituindo o tipo genérico de dados pelo tipo de dados declarado na chamada. Este processo é automaticamente realizado pelo compilador e é invisível ao programador. A nível da aplicação DD3LT, as classes supramencionadas permitem que após a sua declaração, seja possível ignorar o tipo de dados de cada parâmetro.

A classe *CSingleParameter<TYPE>* deriva da classe *CParameterIface* e implementa todas as funções virtuais puras da classe de base. A classe *CMultiParameter<TYPE>* deriva da classe *CSingleParameter<TYPE>* e adiciona à classe de base as funcionalidades e propriedades necessárias ao processamento de parâmetros com opções múltiplas. A Figura 3.4 apresenta esquematicamente a relação de herança entre a classe *CMultiParameter<TYPE>*, *CSingleParameter<TYPE>*, *CParameterIface* e *CObject*.

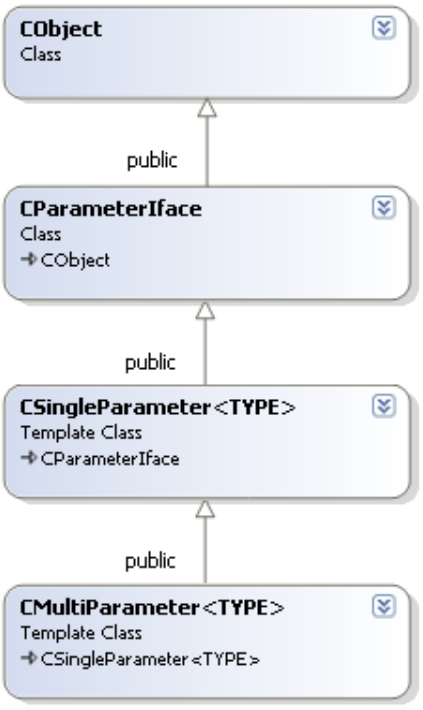


Figura 3.4. Esquema UML representativo da relação de herança entre a classe *CMultiParameter<TYPE>*, *CSingleParameter<TYPE>*, *CParameterIface* e *CObject*.

A classe *CParameterIface* é uma classe abstracta que fornece um interface para um parâmetro genérico. Nesta classe são declaradas funções virtuais puras que exercem

operações comuns a todos os parâmetros de entrada do DD3IMP. Esta classe possibilita a organização dos parâmetros em colecções MFC, permitindo invocar as funções abstractas da interface de uma forma indexada.

3.4. Janela da interface

A classe MFC *CFrameWnd* deriva da classe MFC *CWnd* conferindo-lhe as funcionalidades necessárias para suportar a exibição de um interface de documento único, contendo as implementações necessárias para desempenhar o papel de uma janela principal de uma aplicação Windows. Uma *frame CFrameWnd* deve monitorizar a vista activa que por sua vez é independente da janela da aplicação que está activa no sistema operativo Windows, e da janela que detêm o foco de input, sendo responsável por notificar a vista activa sempre que a janela é reactivada. A *frame CFrameWnd* é também responsável pela delegação de mensagens de comando e notificação aos seus *child frames* e pela gestão da posição de menus, barras de controlo e vistas dentro da sua área cliente. Esta *frame* faz a actualização dos estados dos seus controlos em tempo ocioso (*idle*), lida com os atalhos de teclado e fornece suporte à ajuda sensível ao conteúdo. A classe MFC *CMDIFrameWnd* deriva da classe MFC *CFrameWnd* adicionando às suas funcionalidades a capacidade de gerir múltiplos documentos, funcionando em conjunção com os *child frames* nos quais são exibidas as vistas sobre os documentos. A Figura 3.5 representa esquematicamente a classe MFC *CMDIFrameWnd* bem como os conteúdos pelos quais ela é responsável.

A classe MFC *CMDIFrameWndEx* deriva da classe MFC *CMDIFrameWnd*, estendendo os efeitos visuais disponibilizados e fornecendo um conjunto de características que contribuem para o desenvolvimento de uma aplicação mais amigável como painéis flutuantes, menus *ribbon*, *tabbed views* e uma interacção com o registo do Windows que permite memorizar o *layout* personalizado por cada utilizador. A classe *CMainFrame* deriva da classe MFC *CMDIFrameWndEx*, acrescentando funcionalidades e propriedades necessárias à aplicação DD3LT, como a criação de novas vistas sobre o documento activo, e a exibição de controlos que executam operações sobre os dados. A criação de controlos passa pela criação de objectos na barra de menu, aos quais são associados identificadores de mensagens de comando. Após o accionamento de um controlo (por parte do utilizador) a *framework* gera uma mensagem de comando com o respectivo identificador associado,

sendo esta mensagem roteada através do *loop* de mensagens até ao método que foi definido no mapa de mensagens para efectuar o seu processamento. Por exemplo, para permitir a exibição da vista do ficheiro "DD3_bcon.dat" foi criado o identificador ID_VIEWBCON, associado a um controlo criado para o efeito na barra de menu. Foi também criado um membro *OnShowBConView()*, o qual foi posteriormente associado ao identificador ID_VIEWBCON no mapa de mensagens. Desta forma, durante a execução da aplicação, quando é recebida uma mensagem de comando com o identificador ID_VIEWBCON a *framework* invoca o método *OnShowBConView()*, que processa a mensagem. Esta é seguidamente eliminada do *loop* de mensagens. Sempre que é invocado o método *OnShowBConView()*, este delega a criação da nova vista ao método *ShowDD3View()*, que recorre ao respectivo *document template* para criar a nova vista e a associar ao documento activo, ou que simplesmente activa a vista caso esta já esteja instanciada.

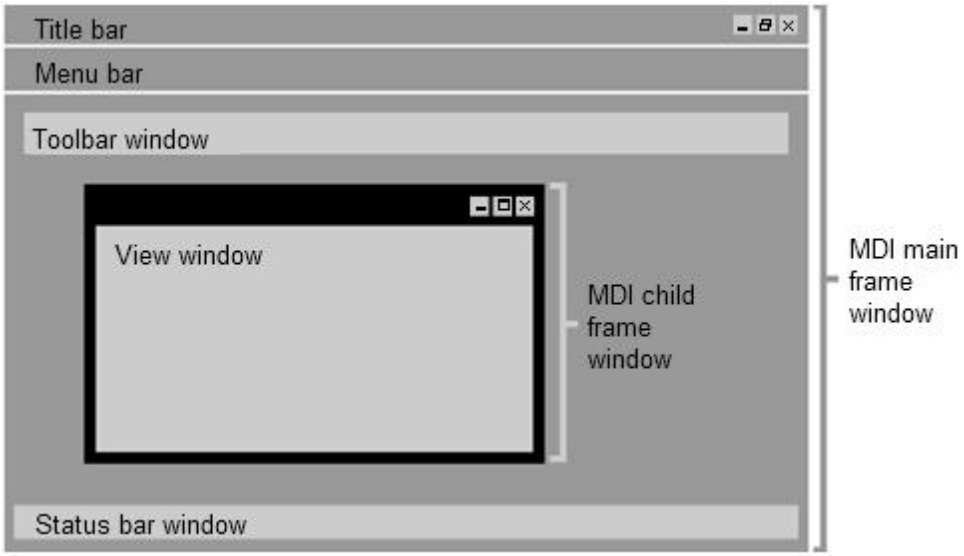


Figura 3.5. Representação da classe MFC *CMDIFrameWnd* com indicação dos conteúdos geridos.

3.5. Grelha de Propriedades

Na interface DD3LT, a alteração dos dados que constam no objecto documento é feita com recurso a grelhas de propriedades encapsuladas por classes que derivam da classe MFC *CMFCPropertyGridCtrl*. Estas classes detêm a capacidade de gerir objectos derivados da classe MFC *CMFCPropertyGridProperty*, que por sua vez constituem um invólucro sobre cada propriedade da grelha. Estas grelhas são disponibilizadas na interface através de um painel flutuante (encapsulado na classe *CPropertiesWnd* que deriva da

classe MFC *CDockablePane*), no qual é possível seleccionar a grelha de propriedades que reúne os parâmetros pretendidos.

Tal como nas secções anteriores, opta-se por descrever estas funcionalidades com o auxílio do ficheiro "DD3_bcon.dat". Neste caso recorre-se à classe *CBConPropertyList*, que deriva da classe MFC *CMFCPropertyGridCtrl*, e é responsável pelas alterações deste ficheiro de dados de entrada. A sua implementação é análoga a todas as outras grelhas de propriedades existentes no DD3LT. Na classe *CBConPropertyList* foi criado um membro apontador para um objecto documento que é ciclicamente actualizado (em tempo ocioso) pela *frame* principal da aplicação através do método *CBConPropertyList::SetDocument()*, garantindo que em cada instância desta grelha de propriedades existe sempre um apontador para o documento activo. Sempre que o documento activo muda as propriedades da grelha são actualizadas. Se não existirem documentos, este apontador assume o valor nulo e as propriedades da grelha são removidas. A inicialização das propriedades da grelha é feita no método *CBConPropertyList::InitPropList()* que recorre ao método auxiliar *CBConPropertyList:: AddParameterProperty()* sempre que é necessário adicionar uma nova propriedade relativa a um dado parâmetro. No método *CBConPropertyList::AddParameterProperty()* cada propriedade é instanciada com: o identificador, o valor original, o valor actual e a lista de opções (caso existam) do parâmetro correspondente. Perante a alteração dos dados de uma das propriedades da grelha, a *framework* invoca o método virtual *CBConPropertyList::OnPropertyChanged()*, enviando como argumento um apontador para o objecto *CMFCPropertyGridProperty* que foi modificado. Neste método virtual é identificado o tipo de modificação, sendo o processamento de cada tipo de modificação delegado aos respectivos métodos:

- O processamento de acções de modificação de parâmetros existentes no documento é delegado ao método *CBConPropertyList::ParameterChange()*, onde é interpretado o grupo de parâmetros a que o parâmetro modificado pertence. Uma vez identificado o grupo, este é enviado como argumento para o método *CBConPropertyList::UpDateParameter()* onde é feita a actualização dos dados a nível do documento.
- O processamento de acções de selecção de índice de condições/restrições (grupos de parâmetros) é delegado ao método *CBConPropertyList::SelectSet()*, no qual é

identificado o tipo de índice seleccionado: condição de simetria, restrição de simetria ou restrição de retorno elástico. Após a identificação, o processamento é delegado a um dos métodos *CBConPropertyList::SetSymCond()*, *CBConPropertyList::SetSymCondRest()* ou *CBConPropertyList::SetOSSRest()*, respectivamente. Todos estes recorrem ao método *CBConPropertyList::UpDateProperties()* para actualizar a exibição dos valores relativos ao grupo seleccionado.

• O processamento de acções de adição ou subtracção de condições/restrições é delegado ao método *CBConPropertyList::AddDelSet()*, onde é feito o processamento destas operações.

3.6. Vista de selecção de simulação

Na interface DD3LT, a selecção do problema é feita através de uma vista encapsulada pela classe *CNewFromDBView* que deriva da classe MFC *CScrollView*, permitindo desta forma fazer *scroll* na vista. A vista de selecção do problema foi implementada sem recurso à arquitectura *document/view*, uma vez que esta deve ser exibida numa fase em que não existe simulação definida. Nesta vista foram introduzidos controlos *CMFCButton*, *CComboBox*, *CEdit*, *CStatic*, *CBrowseFolder* e *CGroupBox* que permitem ao utilizador seleccionar parametricamente o problema. Na implementação do objecto *CNewFromDBView* recorre-se à base de dados para listar as opções disponíveis em listas de selecção. Progressivamente, cada selecção realizada pelo utilizador restringe as opções disponibilizadas nas restantes listas de selecção, conduzindo ao isolamento do problema pretendido. Este procedimento é feito da forma descrita no capítulo 4.

A selecção dos materiais é feita com recurso a um objecto denominado *CMaterGridEx* que deriva da classe *CGridListCtrlEx* constante na biblioteca desenvolvida por Kristensen (2012). No momento da instanciação, o objecto *CMaterGridEx* copia a lista de materiais existentes na base de dados para a memória, organizando-os de uma forma estruturada. Durante o processo de selecção este objecto mantém o registo dos modelos constitutivos disponíveis para o problema seleccionado.

Por defeito a pasta inicialmente definida para guardar os ficheiros a gerar pelo DD3LT é a pasta "Os meus documentos". Para permitir alterar a localização de

armazenamento dos dados do problema a simular, foi criado e integrado um objecto denominado *CBrowseFolder* que disponibiliza um painel de selecção de pastas.

Após a conclusão da selecção de todos os parâmetros necessários é permitido ao utilizador executar a simulação ou avançar para a edição dos ficheiros de dados de entrada.

3.7. Executar a simulação

A interacção entre o DD3LT e o DD3IMP é feita por intermédio do objecto *CConsoleSniffer*. Este objecto foi criado para encapsular as tarefas relativas à execução, monitorização e paragem de uma aplicação de consola *win32*, e têm a capacidade de executar uma dada aplicação de consola numa *thread* paralela, desviando o *output stream* da aplicação para um interface gráfico com recurso a um sistema de mensagens síncrono. Foram criados ainda métodos que monitorizam e controlam o estado da aplicação, sendo desta forma possível cancelar o processo iniciado.

O lançamento da simulação é feito por intermédio de um botão com o identificador ID_DD3_LM, localizado na barra de ferramentas. Como todos os outros botões da barra de ferramentas este botão é criado durante a inicialização do objecto *CMainFrame*, e é mapeado no mapa de mensagens deste mesmo objecto. A simulação é iniciada pelo método *CMainFrame::OnDD3LaunchAndMonitor()*. Depois de actualizar os ficheiros de entrada, este método recorre ao objecto *CConsoleSniffer* para verificar o estado da simulação, cancelando a simulação caso esta esteja a decorrer, ou lançando uma nova simulação no caso de não existir simulação em curso.

O estado deste botão é actualizado em tempo oficioso (pelo método *CMainFrame::OnDD3LaunchAndMonitor()*) de acordo com o resultado da verificação da existência de uma simulação em curso.

3.8. Instalação da DD3LT

Visando simplificar a tarefa de instalar a aplicação DD3LT foi criado um assistente de instalação/desinstalação com o modelo *Visual Studio Installer* disponível no Visual Studio 2010 Professional Sp1. Este assistente instala e regista todos os ficheiros necessários à execução do DD3LT e do DD3IMP.

No momento da instalação o utilizador pode especificar um caminho personalizado para a instalação do DD3LT, ou aceitar o caminho por defeito "\Program Files\UC\DD3LT " onde serão instalados:

- O executável da aplicação DD3LT.exe
- A base de dados DD3data.mdb
- Os utilitários de registo wrapper.exe, RegIntelMKL.bat

O assistente de instalação copia também todos os ficheiros dependentes para a pasta de sistema \Windows\system32:

- OPENGL32.dll
- ODBC32.dll
- \Ifort_lib\32
  - libiomp5md.dll
  - mkl_core.dll
  - mkl_intel_thread.dll
  - mkl_sequential.dll
  - nkl_p4m.dll
- \Ifort_lib\64
  - libiomp5md.dll
  - mkl_core.dll
  - mkl_intel_thread.dll
  - mkl_sequential.dll
  - nkl_p4n.dll

O caminho para as bibliotecas necessárias à execução do DD3IMP é adicionado à *path* pelo *batch script RegIntelMKL.bat* que é executado na instalação por intermédio do executável wrapper.exe.

No momento da desinstalação, o assistente remove todas as modificações feitas no acto da instalação, removendo desta forma a aplicação DD3LT do sistema.

3.9. Visualização do modelo

A visualização gráfica da discretização do corpo deformável, das superfícies que definem as ferramentas e dos gráficos de evolução das variáveis de processo é feita recorrendo à biblioteca *Visualization Toolkit* (VTK) da Kitware. O VTK é um *software open source* gratuito de visualização e processamento de gráficos 3D, que consiste numa biblioteca de classes C++. O VTK foi criado e é continuamente desenvolvido pela Kitware que disponibiliza também serviços de suporte profissional. O VTK suporta uma enorme variedade de algoritmos de visualização incluindo: escalares, vectores, tensores e métodos volumétricos.

A estrutura central do VTK é um *pipeline* de dados, desde a fonte da informação até à imagem renderizada no ecrã. Uma aplicação típica tem vários destes *pipelines* que correspondem a diferentes itens que aparecem no ecrã. Estes itens podem ser representações diferentes do mesmo conjunto inicial de dados, diferentes selecções dos dados de *input* ou objectos e imagens totalmente independentes. A Figura 3.6 apresenta a estrutura central utilizada pelo VTK.

O VTK foi integrado na aplicação DD3LT através dos objectos *CVTKView* e *CVTKGraphView*, derivados da classe MFC *CView*, estabelecendo uma adaptação do VTK à framework MFC. Estes objectos foram criados para adaptar o VTK ao DD3LT, delegando as tarefas de actualização gráfica e processamento de eventos da janela que contém a vista, à framework VTK. Os *inputs* do utilizador são directamente delegados à framework VTK através do reencaminhamento das mensagens: WM_LBUTTONDOWN, WM_LBUTTONUP, WM_MBUTTONDOWN, WM_MBUTTONUP, WM_RBUTTON DOWN, WM_RBUTTONUP, WM_MOUSEMOVE, WM_CHAR, WM_TIMER; permitindo ao utilizador manipular o ambiente gráfico.

As actualizações gráficas relacionadas com o redimensionamento do tamanho e *refresh* da janela de visualização são respectivamente processadas na redefinição dos métodos virtuais *CView::OnSize()* e *CView::OnDraw()*, nos quais a tarefa de actualização gráfica é delegada à instância do objecto *vtkWin32OpenGLRenderWindow*. O evento relacionado com a mensagem WM_ERASEBKGND não faz sentido no contexto do VTK, e como tal é "silenciado" na redefinição do método virtual *CView::OnEraseBkgnd()*.

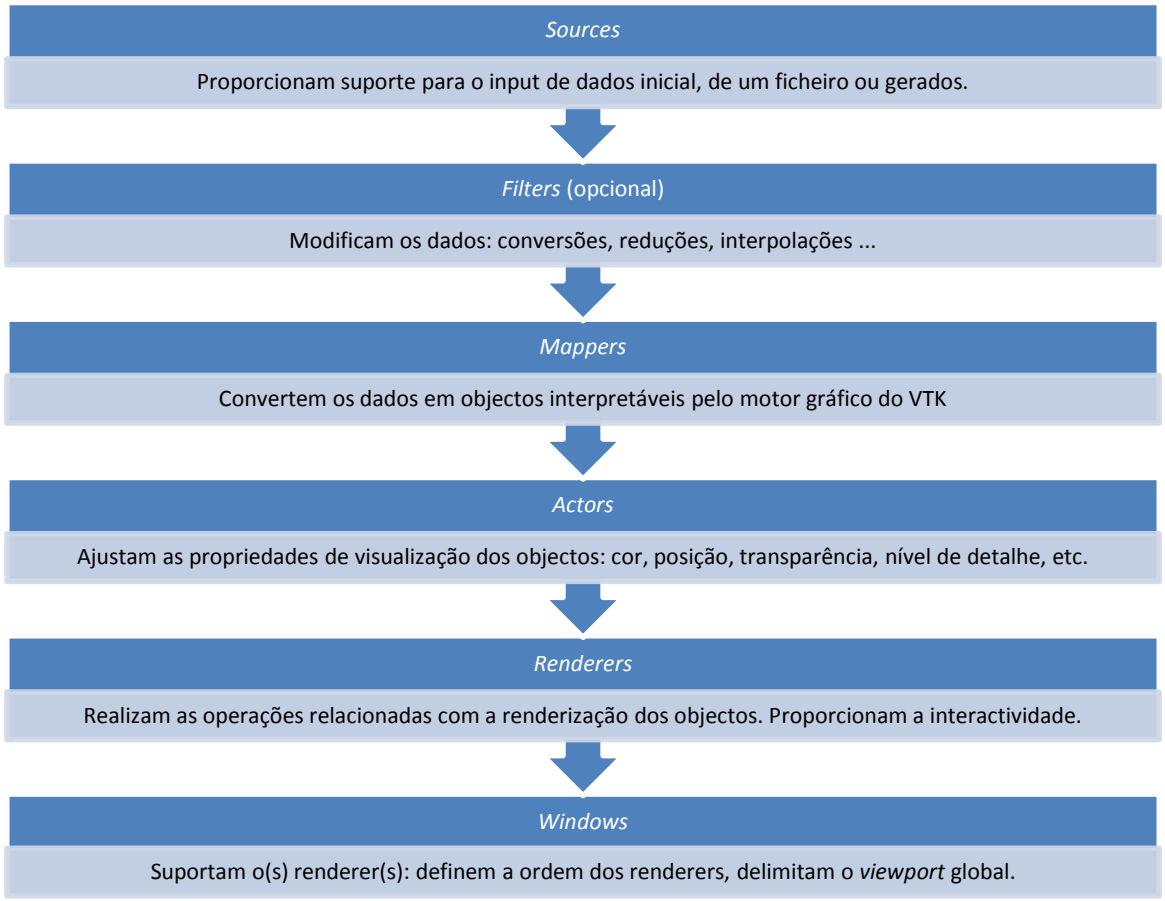


Figura 3.6. Representação esquemática do *pipeline* de informação utilizado pelo VTK.

O pseudo algoritmo genérico de uma aplicação VTK pode ser resumido da seguinte forma:

1. Criar uma *window*;
2. Criar um *renderer*;
3. Associar o *renderer* à *window*;
4. Criar uma geometria;
5. Criar um *mapper*;
6. Associar a geometria ao *mapper*;
7. Criar um actor;
8. Associar o *mapper* ao actor;
9. Associar o actor ao *renderer*;
10. Renderizar a *window*.

No objecto *CVTKView*, a criação dos objectos do VTK foi implementada nos métodos *CVTKView::Alloc_VTK()* e *CVTKView::Alloc_Geometry()*. A associação dos *renderers* à *window* e respectiva e inicialização foi implementada nos métodos *CVTKView::Setup_VTK()* e *CVTKView::IF_Initialize_VTK()*. Por fim, as associações das geometrias aos *mappers*, dos *mappers* aos actores e dos actores aos *renderers* foram implementadas no método *CVTKView::Pipeline()*. Desta forma, no DD3LT o pseudo algoritmo acima mencionado é distribuído pelos métodos indicados na Figura 3.7.

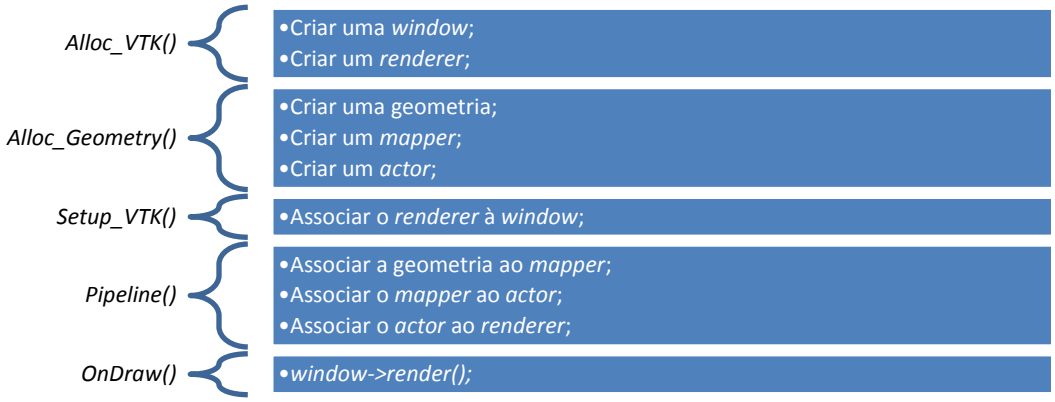


Figura 3.7. Pseudo algoritmo da aplicação VTK implementada no DD3LT.

De modo a isolar os dados da respectiva visualização, foi criado um objecto *CVTKModel*, onde são encapsulados os dados e os parâmetros de visualização. Este objecto reúne um conjunto de operações de gestão dos dados exibidos no VTK.

No intuito de organizar os dados relativos às malhas de elementos finitos, foram criadas as classes *CMesh* e *CMeshNode*. De forma análoga foram criadas as classes *CBezier*, *CBezierControlPoint* e *CBezierPatch* para organizar os dados relativos às superfícies paramétricas do tipo Bézier.

A funcionalidade de exibição de superfícies paramétricas do tipo Bézier foi adicionada ao VTK através da criação do objecto *vtkParametricBezier*, derivado da classe *vtkParametricFunction*, no qual foram implementados métodos que calculam as coordenadas cartesianas no sistema local, e as derivadas parciais para um dado ponto no domínio paramétrico.

Visando agilizar a criação de geometria interpretável pelo VTK, foi criado o objecto *CDD3VTKObjFactory*. Este objecto é responsável pela criação de geometria a

partir das classes *CMesh* e *CBezier*, lidando com as conversões necessárias de conectividade das primitivas geométricas (GID, DD3IMP, VTK).

4. DD3LT

Tal como referido anteriormente, o principal objectivo da aplicação interactiva é reduzir a complexidade de utilização do programa DD3IMP. O facto de o utilizador ter acesso a um vasto conjunto de parâmetros é importante para a sua formação. No entanto, contribuí para aumentar as suas dificuldades numa primeira abordagem. Assim, a redução da complexidade tem como principal objectivo reduzir o tempo e o esforço necessários para atingir a fase de análise de resultados, que é a etapa mais importante no estudo do processo tecnológico de estampagem. Neste contexto, a aplicação interactiva foi construída com base no princípio que o utilizador não está familiarizado com todos os parâmetros disponíveis nos ficheiros de entrada. Assim, apesar de todos os parâmetros estarem visíveis e o utilizador receber informação acerca deles, a aplicação procura controlar a validade das alterações.

Neste capítulo descrevem-se as principais funcionalidades da aplicação interactiva DD3LT. O funcionamento da aplicação baseia-se na selecção de uma simulação pré-definida, de modo a que o utilizador proceda a alterações nesse modelo. Assim, as principais funcionalidades da aplicação incluem:

1. Seleccionar uma simulação existente na base de dados;
2. Seleccionar uma simulação existente na base de dados e alterar:
    a) o(s) material(is) ou o modelo constitutivo;
    b) a discretização do corpo deformável, por exemplo, o tipo de elementos finitos, a estratégia de integração, as dimensões do corpo;
    c) as condições de processo, por exemplo, o coeficiente de atrito entre o corpo deformável e cada ferramenta, o valor da força de aperto do cerra-chapas, o valor do deslocamento total do punção, a estratégia de remoção das ferramentas;
    d) os parâmetros numéricos, por exemplo, a dimensão do incremento tentativa de cada fase, o número máximo de iterações em cada incremento, o parâmetro de penalidade do lagrangiano aumentado;
3. Executar a simulação seleccionada/configurada.

4. Visualizar a configuração inicial.
5. Gerar o gráfico de evolução das forças das ferramentas, aquando da conclusão da simulação.

Nas secções seguintes exemplificam-se algumas das opções disponíveis, de modo a orientar a futura utilização da plataforma interactiva.

4.1. Executar uma simulação pré-definida

A Figura 4.1 apresenta o *MainFrame* da aplicação DD3LT, correspondente à activação da selecção de uma simulação pré-existente na base de dados. Após a selecção desta opção, surge o *ChildFrame* correspondente à selecção das opções para a simulação pré-existente, como se mostra na Figura 4.2. De modo a guiar o utilizador, neste painel identificam-se os passos para a selecção. Tal como referido no capítulo 3, a primeira opção consiste na selecção da ferramenta, como indicado pelo número 1. Para cada ferramenta seleccionada a interface permite visualizar a ferramenta (campo IMAGE, da tabela 0SIMUL) bem como uma pequena descrição (campos DESCR e OBS, da tabela 0SIMUL). A opção seguinte corresponde à selecção das opções disponibilizadas para essa ferramenta (campo SIMULoption da tabela 0SIMULcase), como indicado pelo número 2 na Figura 4.2. Para cada opção seleccionada é mostrada a discretização por defeito (campo infoMESHimage da tabela 4MESH) bem como uma breve descrição (campo SIMULnotes da tabela 0SIMULcase). O passo seguinte corresponde à selecção da discretização, de entre as opções disponibilizadas (campo MESHopt da tabela 4MESH), como indicado pelo número 3 na Figura 4.2. Neste caso, para cada opção seleccionada é actualizada a imagem (campo infoMESHimage da tabela 4MESH).

O passo seguinte corresponde à selecção do material, como indicado pelo número 4 na Figura 4.2. Neste quadro optou-se por permitir seleccionar qualquer material que conste da base de dados. Assim, o utilizador começa por seleccionar o material (campo MATname comuns às tabelas 5MAT_HD/5MAT_YldCRIT). De seguida o utilizador deve seleccionar a origem da identificação dos parâmetros dos modelos constitutivos para esse material (campo MATlabel comuns às tabela 5MAT_HD/5MAT_YldCRIT). A selecção destes dois campos permite verificar na base de dados quais são os modelos constitutivos disponíveis, i.e. lei de encruamento (campo IDplaslaw da tabela 5MAT_HD) e critério de plasticidade (campo IDyldcrit da tabela 5MAT_YldCRIT). No entanto, no quadro opta-se

por mostrar o campo correspondente à descrição do modelo constitutivo (campo DESCR da tabela 5MAT_HDparam e 5MAT_YldCRITparam) de modo a auxiliar o utilizador.

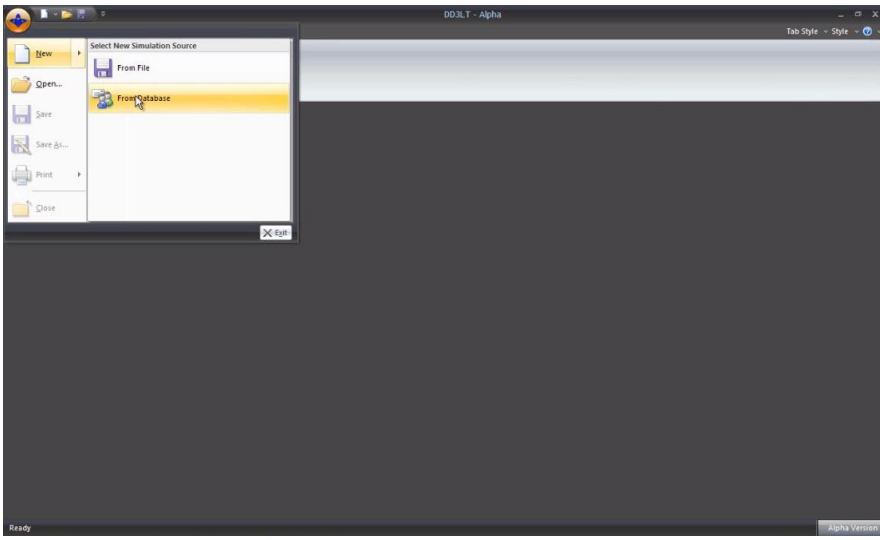


Figura 4.1. *MainFrame* da aplicação DD3LT.

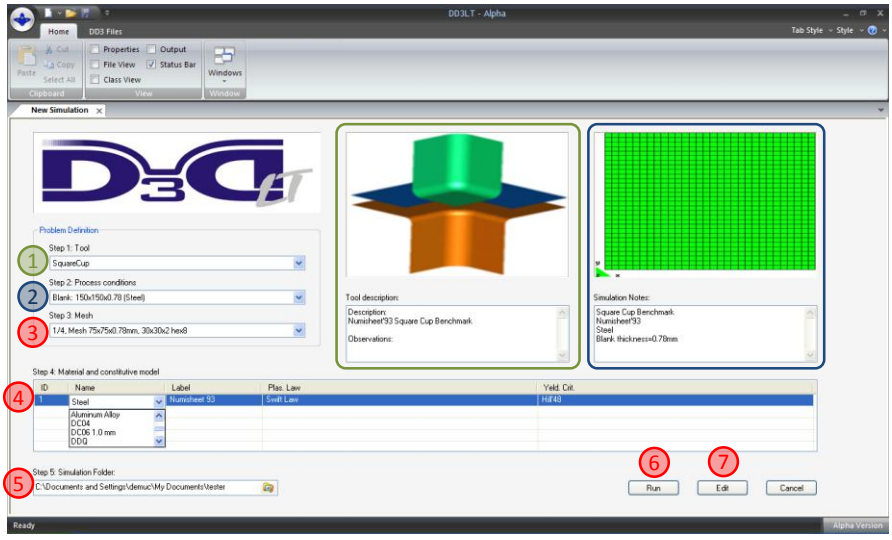


Figura 4.2. *ChildFrame* correspondente à selecção de uma simulação pré-definida DD3LT.

O quinto passo corresponde à selecção/criação da directoria onde serão armazenados os ficheiros de entrada e executada a simulação, como indicado pelo número 5 na Figura 4.2. Por último, o utilizador pode executar a simulação seleccionando a opção *Run*, como indicado pelo número 6 na Figura 4.2.

4.2. Alterar uma simulação pré-definida

Após a selecção de um exemplo pré-definido, como descrito na secção anterior, o utilizador pode optar por editar os ficheiros de entrada associados a essa simulação. Para tal, basta seleccionar a opção *Edit*, como indicado pelo número 7 na Figura 4.2. A aplicação avança, por defeito, para o painel de visualização das ferramentas e do corpo deformável, como se mostra na Figura 4.3. Para visualizar os ficheiros de entrada é necessário seleccionar a opção DD3 Files, e seleccionar o ficheiro de entrada pretendido. A selecção de um ficheiro inicializa o *ChildFrame* para a sua visualização, como se mostra na Figura 4.4. Para activar o *FloatingPane* correspondente à edição dos parâmetros do ficheiro é necessário activar a opção *Properties*, em *Home*, como se mostra na Figura 4.5. Uma vez activada esta opção, este painel mostra os parâmetros correspondentes ao ficheiro de entrada seleccionado.

O utilizador pode optar por apenas visualizar as condições pré-definidas e lançar a simulação. Para tal, basta seleccionar a opção *Run DD3IMP*, como se mostra na Figura 4.4.

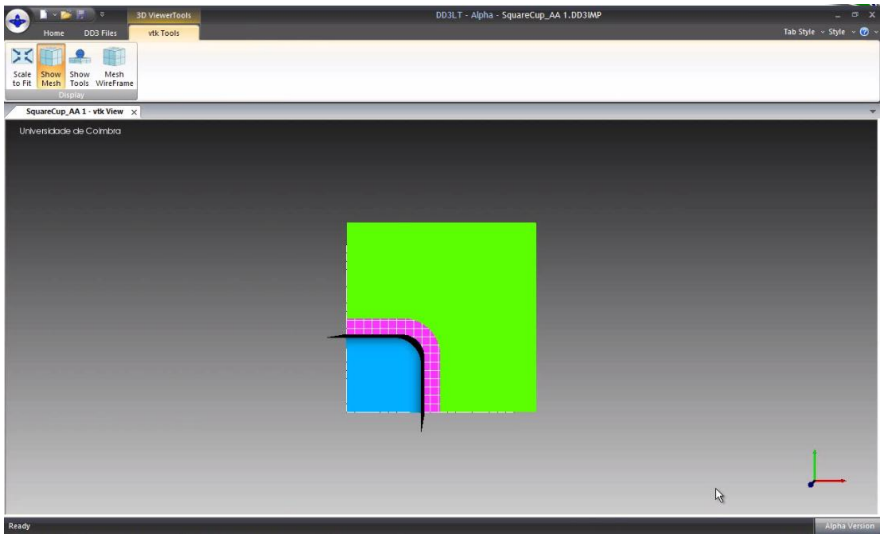


Figura 4.3. *ChildFrame* correspondente à visualização das condições iniciais de uma simulação no DD3LT.

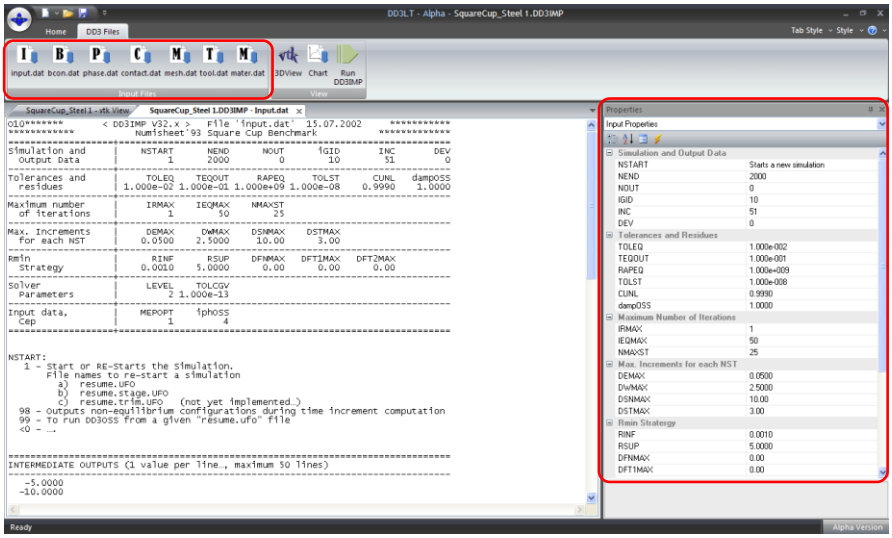


Figura 4.4. *ChildFrame* e *FloatingPane* correspondente à visualização e edição do ficheiro DD3_input.dat de uma simulação no DD3LT.

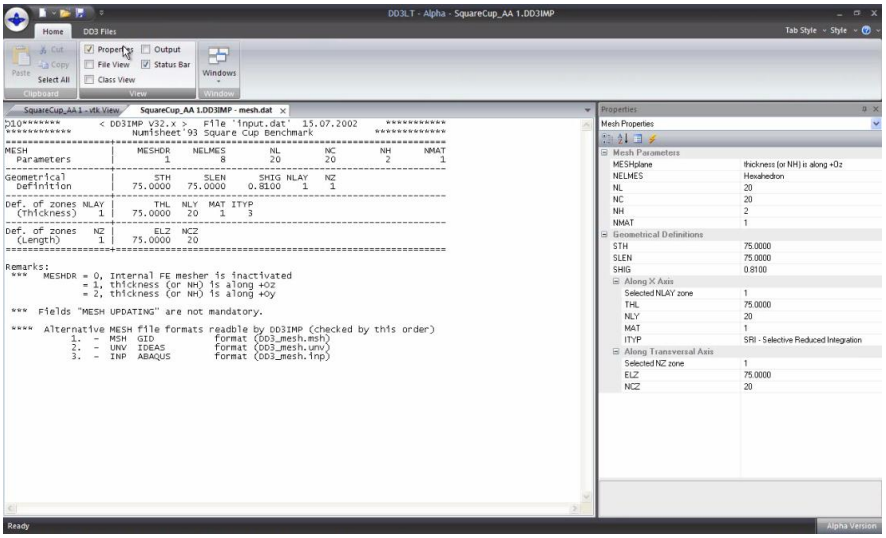


Figura 4.5. Activação da opção *Properties* para visualização do *FloatingPane* correspondente à visualização e edição do ficheiro DD3_mesh.dat de uma simulação no DD3LT.

Para proceder à alteração de parâmetros em qualquer dos ficheiros de entrada recorre-se à utilização do *FloatingPane*: *Properties*, uma vez que ao seleccionar um determinado parâmetro é fornecida ao utilizador uma breve descrição (campo DESCfield da tabela Zfields), como se mostra na Figura 4.6. Por outro lado, o *FloatingPane* permite o controlo das variáveis uma vez que impõe o formato de escrita no ficheiro e limita as opções às existentes na base de dados. De modo a guiar o utilizador, os parâmetros

alterados são apresentados a negrito quer na *ChildFrame* do ficheiro quer no *FloatingPane*.

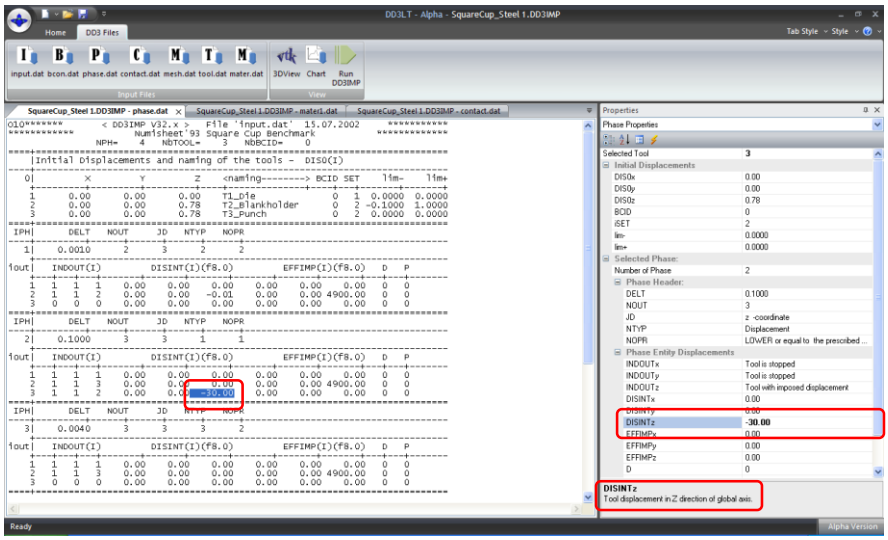


Figura 4.6. Exemplo de alteração do parâmetro DISINTz do ficheiro DD3_phase.dat de uma simulação no DD3LT.

Uma vez que o *ChildFrame* de selecção de uma simulação permite a escolha de qualquer material introduzido na base de dados, como se explicou na subsecção 4.1, na prática permite redefinir a simulação de partida. No entanto, optou-se por permitir alterar também a selecção do material no *FloatingPane*: *Properties* associado ao ficheiro DD3_mater.dat, como se mostra na Figura 4.7. Também neste caso, o utilizador começa por seleccionar o material (campo MATname comuns às tabelas 5MAT_HD/5MAT_Yld CRIT), para de seguida seleccionar a origem da identificação dos parâmetros dos modelos constitutivos para esse material (campo MATlabel comuns às mesmas tabelas). Uma vez seleccionado um determinado material e modelo constitutivo o utilizador pode alterar qualquer parâmetro. Deste modo, a aplicação permite ao utilizador definir um material que não consta da base de dados. A pré-selecção da lei de encruamento e do critério de plasticidade garante que apenas os parâmetros associados ao modelo constitutivo estarão activos no *FloatingPane*: *Properties*, o que facilita a introdução de dados.

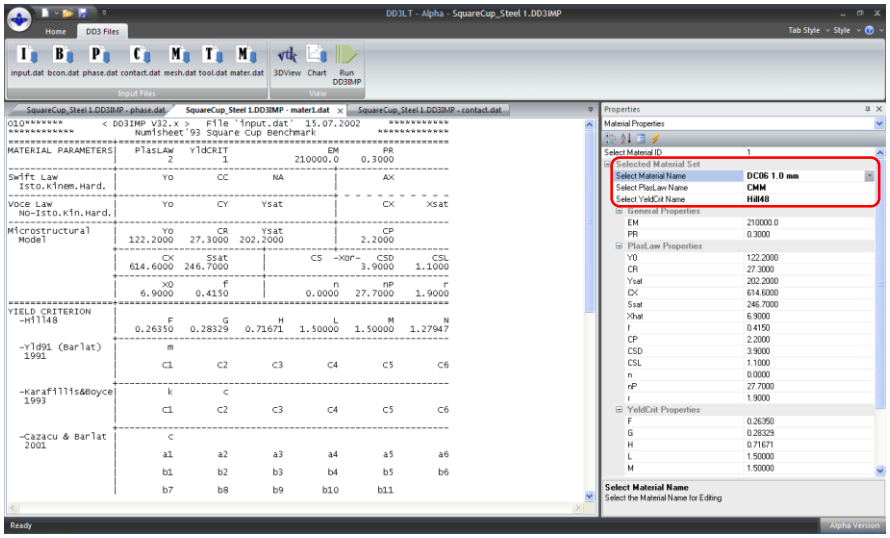


Figura 4.7. Exemplo de alteração do material seleccionado para o ficheiro DD3_mater.dat de uma simulação no DD3LT.

A aplicação interactiva permite a visualização das alterações realizadas, em tempo real, com ao auxílio da opção *3DView*, como se mostra na Figura 4.8. Esta opção é particularmente interessante para visualizar as alterações realizadas às opções de discretização do gerador interno do DD3IMP, como se mostra na Figura 4.9. Uma vez concluído o procedimento de alteração de dados basta seleccionar a opção Run DD3IMP, como se mostra na Figura 4.4, para executar a simulação.

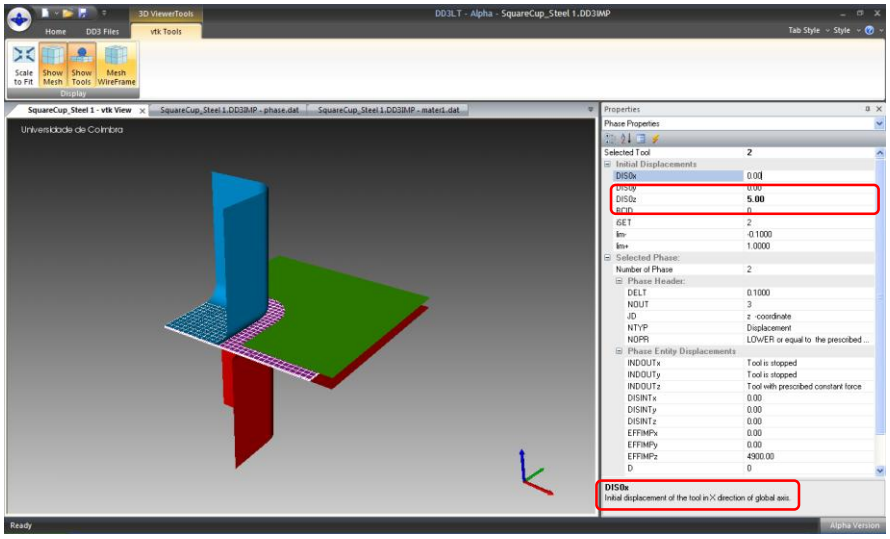


Figura 4.8. Exemplo de alteração do parâmetro DIS0z do ficheiro DD3_phase.dat de uma simulação no DD3LT.

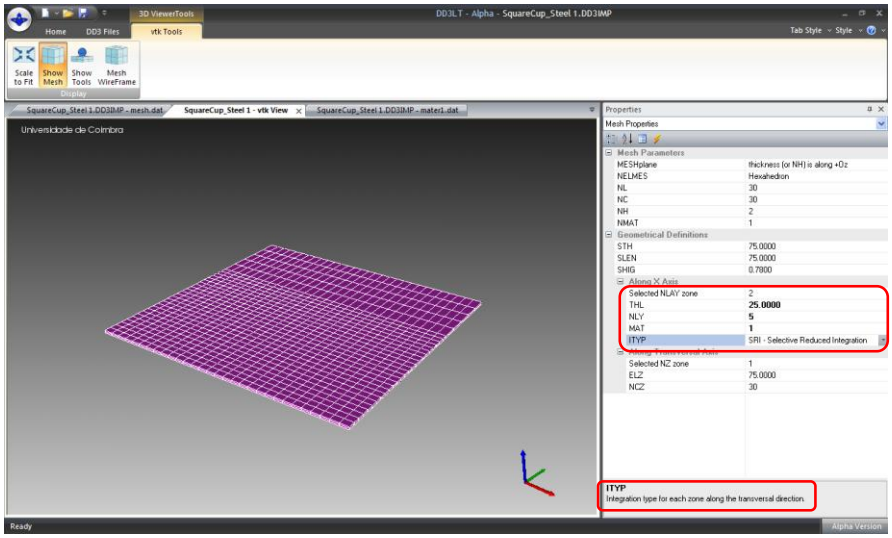


Figura 4.9. Exemplo de alteração da discretização no ficheiro DD3_mesh.dat de uma simulação no DD3LT.

4.3. Análise de resultados

Ao lançar a simulação surge automaticamente o *FloatingPane*: *Output*, onde o utilizador pode acompanhar o decurso da simulação, como se mostra na Figura 4.10. Este recurso pode ser activado ou desactivado no menu *Home: Output*, como se mostra na Figura 4.5. No final da simulação o utilizador pode construir o gráfico de evolução das forças, seleccionando a opção *Chart*, do menu *DD3 Files*. O *FloatingPane*: *Properties* associado a esta opção permite seleccionar a ferramenta pretendida e a direcção da força e do deslocamento, como se mostra na Figura 4.11.

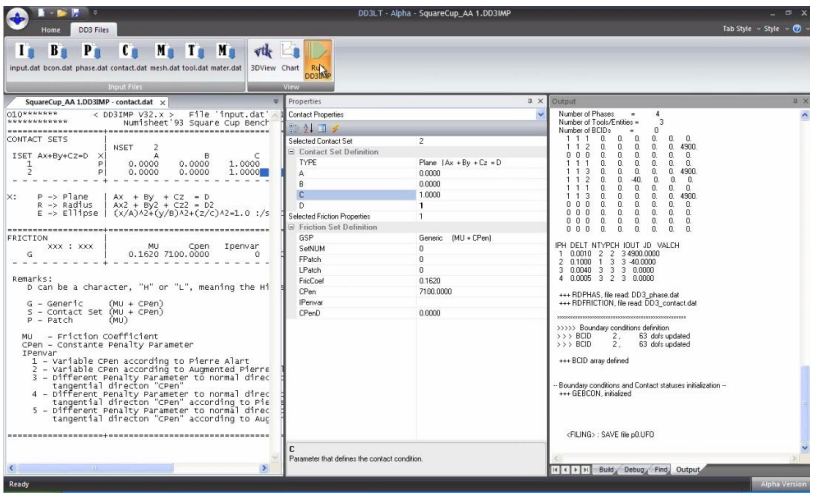


Figura 4.10. *FloatingPane*: *Output* correspondente à visualização de uma simulação no DD3LT.

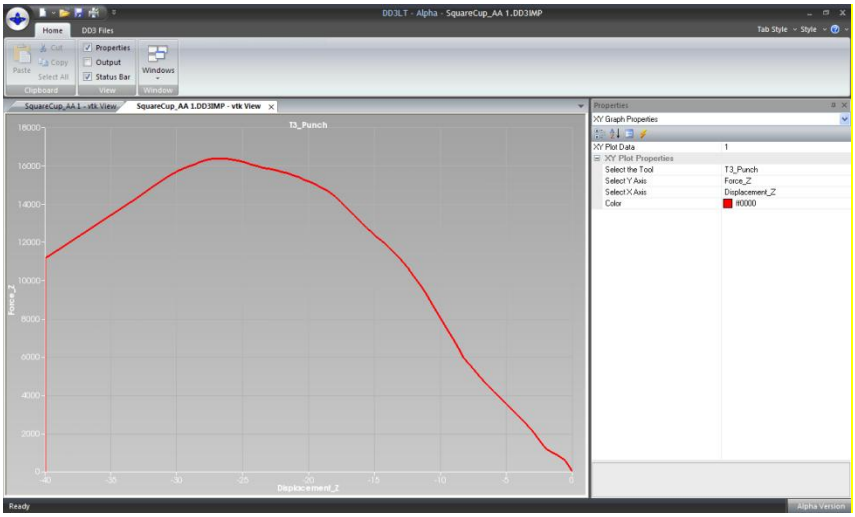


Figura 4.11. Exemplo de construção do gráfico de evolução da força em função do deslocamento.

5. CONCLUSÕES

Dado a enorme quantidade de parâmetros e as interacções entre estes, a desconstrução dos ficheiros de introdução de dados necessários à definição de uma simulação do DD3IMP revelou-se uma tarefa extensa e morosa, tendo sido a complexidade de execução desta tarefa superior ao inicialmente esperado. Contudo esta tarefa foi absolutamente necessária para estruturar correctamente a base de dados e definir a estrutura da plataforma DD3LT.

O desenvolvimento da plataforma DD3LT na framework MFC revelou, em primeira instância, exigir um conhecimento sólido das funcionalidades da linguagem C++. Este facto foi inicialmente subestimado, resultando em dificuldades na implementação que exigiram um aprofundamento dos conhecimentos necessários para as ultrapassar. Em segunda instância, revelou-se difícil a tarefa de encontrar documentação actualizada que proporcione uma introdução ao desenvolvimento de aplicações MFC. A documentação disponível para este efeito encontra-se desactualizada, não sendo representativa da realidade das classes MFC actuais. Por outro lado, a documentação actualizada está apenas disponível *online* na Microsoft Developer Network (MSDN), e é orientada para o desenvolvedor experiente, consistido numa descrição resumida dos métodos e atributos das classes MFC, esta documentação apresenta uma escassez de exemplos de aplicação/implementação.

A integração da biblioteca VTK revelou-se mais complexa do que o inicialmente esperado. A interacção desta com a framework MFC encontra-se muito pouco documentada, limitando-se a um exemplo simples de implementação. Desta forma a integração do VTK na plataforma DD3LT foi conseguida através de um processo progressivo de tentativa erro. Apesar das dificuldades, o resultado desta integração foi bem-sucedido. Uma vez integrado, o desenvolvimento das funcionalidades associadas ao VTK é relativamente mais simples uma vez que esta tarefa se encontra documentada. A biblioteca VTK está bem documentada, existindo disponíveis vários exemplos de aplicação e teste. De facto, o VTK é uma plataforma tão flexível que possibilita a ocorrência de

implementações inadequadas que resultam numa diminuição de eficiência computacional e robustez.

Todos estes factores contribuíram globalmente para um aumento do tempo de desenvolvimento inicialmente previsto, revelando a dificuldade de estabelecer um plano de trabalhos para um projecto desta natureza.

De uma forma geral, considera-se que os objectivos inicialmente propostos foram alcançados, pelo que o trabalho desenvolvido permite realizar as tarefas inicialmente requeridas. Por outro lado, ao longo do desenvolvimento foi considerada a hipótese de adicionar funcionalidades úteis no contexto do uso desta plataforma, como por exemplo a possibilidade de adicionar simulações à base de dados a partir dos ficheiros de input, não tendo sido implementadas dada a impossibilidade de o fazer no tempo reservado a esta tese, podendo no entanto vir a ser adicionadas futuramente.

Dado a natureza iterativa do ciclo de desenvolvimento de *software*, este trabalho pode ser considerado de certa forma inacabado. Perante a possibilidade de continuação deste trabalho, deverá ser feita uma avaliação da estrutura da plataforma DD3LT considerando uma reavaliação das funcionalidades necessárias, e uma posterior restruturação do código tendo em vista o aumento da versatilidade e da robustez da plataforma.

Por último, o trabalho desenvolvido revelou-se extenso e complexo, e apresenta algumas lacunas a nível de documentação do código implementado. A plataforma foi desenvolvida de forma a facilitar a edição da estrutura dos ficheiros de *input* a partir da base de dados, permitindo adicionar novos campos e funcionalidades de uma forma que minimiza a intervenção a nível do código do DD3LT, facilitando o processo de actualização. A base de dados possibilitará a gestão dos dados das simulações realizadas, ou mesmo o desenvolvimento de futuras ferramentas que com ela interajam, sendo provavelmente esta a principal contribuição deste trabalho para o ensino e a aprendizagem dos métodos de produção virtual associados às tecnologias de estampagem de chapas metálicas.

REFERÊNCIAS BIBLIOGRÁFICAS

Alves J.L., Menezes L.F., "Application of tri-linear and tri-quadratic 3-D solid finite elements in sheet metal forming simulations". *Proceedings of the Numiform'01 on Simulation of materials processing: Theory, Methods and Applications*, Mori K.-I. (eds.), Balkema, Rotterdam, 639–644, 2001.

Alves J.L., Simulação numérica do processo de estampagem de chapas metálicas-Modelação mecânica e métodos numéricos, Tese de doutoramento, Departamento de Engenharia Mecânica da Escola de Engenharia da Universidade do Minho, Portugal (2003).

Areias P.M.A., César de Sá, J.M.A., Conceição António C.A., Fernandes A.A., "Analysis of 3D problems using a new enhanced strain hexahedral element". *Int. J. Numer. Meth. Eng.* 58 (11), 1382–1637, 2003.

Bartlat F., Lege D.J., Brem, J.C., "A six-component yield function for anisotropic materials". *Int. J. Plasticity*. 7, 693-712, 1991.

Milewski B., "C++ In Action: Industrial Strength Programming Techniques", Addison-Wesley, 2001.

Cazacu O., Barlat F., "Generalization of Drucker's yield criterion to orthotropy". *Math. Mech. Solids*. 6, 613-630, 2001.

Cazacu O., Plunkett B., Barlat F., "Orthotropic yield criterion for hexagonal closed packet metals". *Int. J. Plasticity*. 22, 1171-1194, 2006.

Chung K., Kuwabara T., Verma R.K., Park T., Huh H., Bae G., "BM 4 - Pre-strain Effect on Spring-back of 2-D Draw Bending". NUMISHEET 2011, The NUMISHEET 2011 Benchmark Study of the 8th International Conference and Workshop on Numerical Simulation of 3D Sheet Metal Forming Processes, Huh H., Chung K., Han S.S., Chung W.J. (eds.), 171-209, 2011.

Drucker D.C., "Relation of experiments to mathematical theories of plasticity". *J. App. Mech.* 16, 349-357, 1949.

Hill R., "A theory of yielding and plastic flow of anisotropic materials". *Proceedings of Mathematical, Physics and Engineering Science*, Royal Society of London, A193, 1948.

Hosford W.F., "A generalized isotropic yield criteria". *J. Appl. Mech.* 39, 607-609, 1972.

Hughes T.J.R., "Generalization of selective integration procedures to anisotropic and nonlinear". *Int. J. Numer. Meth. Eng.* 15, 1413–1418, 1980.

Prosise J., "Programming Windows with MFC", 2ª Edition, Microsoft Press, 1999.

Horton I., "Ivor Horton's Beginning Visual C++ 2010", Wiley Publishing, 2010.

Jiao Z.-P., Li C., "A new formulation of eight-node hexagonal solid element". *Comput. Meth. Appl. Mech. Eng.*, 187 (1–2), 213–217, 2000.

Karafilis A.P., Boyce M.C., "A general anisotropic yield criterion using bounds and a transformation weighting tensor". *J. Mech. Phys. Solids*. 41, 1859-1886, 1993.

Kristensen R., "CGridListCtrlEx - Grid Control Based on CListCtrl". Acedido em 7 de Setembro, em: http://www.codeproject.com/Articles/29064/CGridListCtrlEx-Grid-Control-Based-on-CListCtrl

Lemaître J., Chaboche J-L., "Mechanics of solid materials", *Cambridge University Press*, Cambridge, Reino Unido, 1985.

Li K.P., Carden W.P., Wagoner R.H., "Simulation of springback". *Int. J. Mech. Sci.* 44 (1), 103–122, 2002.

Lingbeek R.A., "Aspects of a Design tool for Springback Compensation". Msc. Thesis, University of Twente, Netherlands, 2003.

Mc-Meeking R.M., Rice J.R., "Finite element formulations for problems of large elastic-plastic deformation", *Int. J. Solids Struct.* 11, 601-616, 1975.

Menezes L.F., Teodosiu C., Makinouchi A., "3-D solid elasto-plastic elements for simulating sheet metal forming processes by the finite element method". *Proceedings FE-simulation of 3-D Sheet Metal Forming Processes in Automotive Industry*, VDI Berichte (eds.), 864, 381–403, 1991.

Menezes L.F., Teodosiu C., "Three-dimensional numerical simulation of the deep-drawing process using solid finite element". *J. Mater. Process. Tech.* 97, 100-106, 2000.

Microsoft, "MSDN Library". Acedido em 7 de Setembro, em: http://msdn.microsoft. com/library/, 2012.

Neto D.M., Oliveira M.C., Menezes L.F., Alves J.L., FEA of frictional contact problems using Nagata patches for surfaces description, WCCM 2012, 10th World Congress on Computational Mechanics, P.M. Pimenta, E.M.B Campello (eds.), p. 304-305, (CD-ROM pp. 20), 2012.

Oliveira M.C., Alves J.L., Menezes L.F., "Improvement of a frictional contact algorithm for strongly curved contact problems". *Int. J. Numer. Meth. Eng.* 58, 2083-2101, 2003.

Oliveira M.C., Alves J.L., Menezes L.F., "Algorithms and Strategies for Treatment of Large Deformation Frictional Contact in the Numerical Simulation of Deep Drawing Process". *Arch. Comput. Meth. Eng.* 15, 113-162, 2008.

Prager W., "The theory of Plasticity: a survey of recent achievements". *Proc. Inst. Mech. Eng.*, 169, 41-57, 1955.

Reese S., "On a physically stabilized one point finite element formulation for three-dimensional finite elastoplasticity". *Comput. Meth. Appl. Mech. Eng.* 194 (45–47), 4685–4715, 2005.

Roll K., Rohleder M., "Complex testing tool for the investigation of springback deviations". NUMISHEET 2002, The 5th International Conference and Workshop on Numerical Simulation of 3D Sheet Metal Forming Processes, Yang D.Y., Oh S.I., Huh H., Kim Y.H. (eds.), 131-136, (2002).

Schach S.R., "Object-oriented and classical software engineering / Stephen R. Schach.", 8th Edition, McGraw-Hill, 2011.

Björnander S., "Microsoft Visual C++ Windows Applications by Example", Packt Publishing, 2008.

Tekkaya A.E., "State-of-the-art of simulation of sheet metal forming". *J. Mater. Process. Tech.* 103, 14–22, 2000.

Teodosiu C., Hu Z., "Microstructure in the continuum modelling of plastic anisotropy", *19th Riso International Symposium on Materials Science: Modelling of Structure and Mechanics of Materials from Microscale to Products*, Riso National Laboratory, 149-168, 1998.

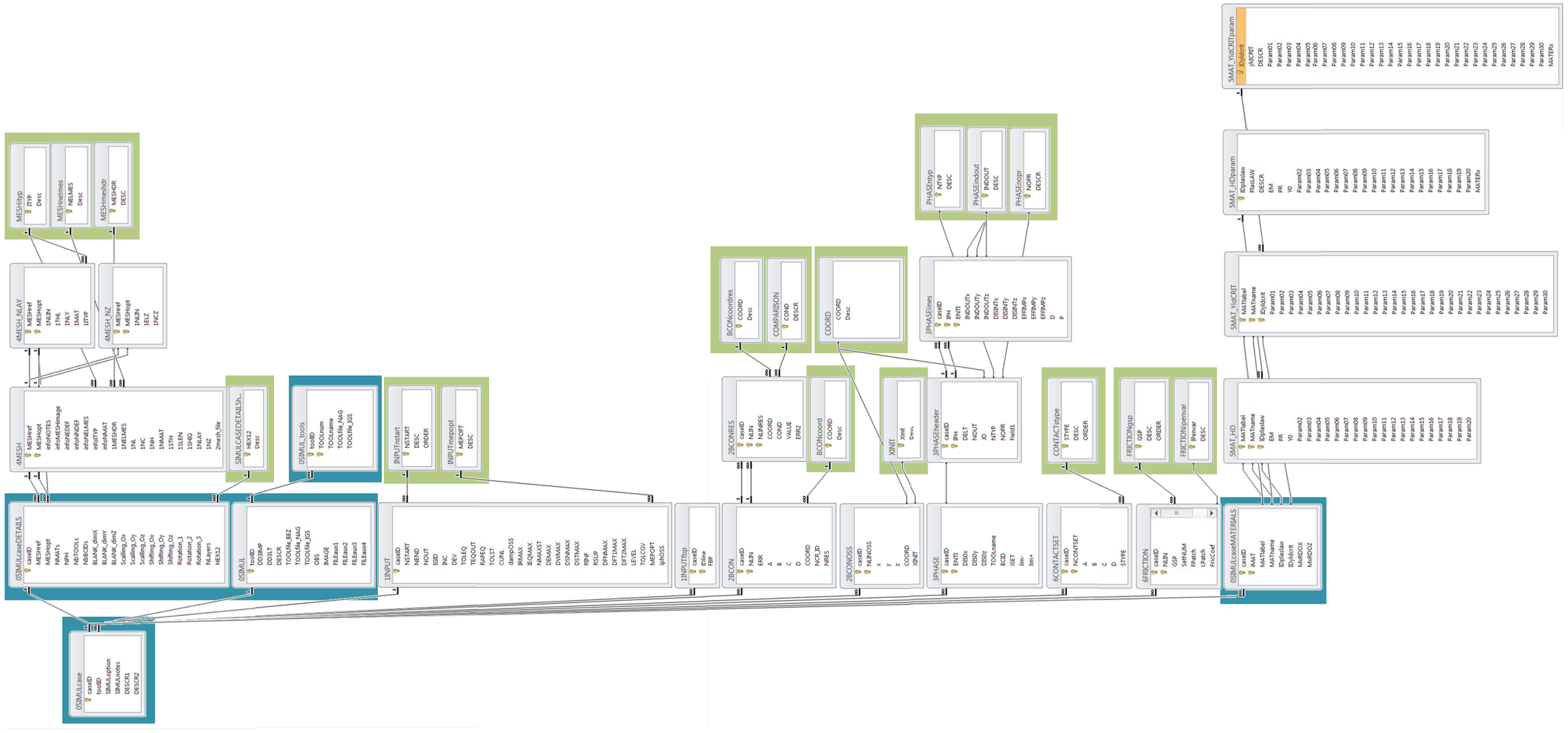
Tisza M., "Numerical modelling and simulation in sheet metal forming". J. Mater. Process. Tech. 151 (1–3), 58–62, 2004.

von Mises R., "Mechanik der festen Körper im plastisch deformablen Zustand". *Göttin. Nachr. Math. Phys.* 1, 582–592, 1913.

Wang J., Wagoner R.H., "A practical large-strain solid finite element for sheet forming". *Int. J. Numer. Meth. Eng.* 63 (4), 473–501, 2005.

Yamada Y., Yoshimur N., Sakurai T., "Plastic stress–strain matrix and its application for the solution of elastic–plastic problems by the finite element method". *Int. J. Mech. Sci.* 10 (5), 348–354, 1968.

ANEXO A: ESQUEMA DA BASE DE DADOS



ANEXO B: DESCRIÇÃO DAS TABELAS

De modo a organizar as tabelas da base de dados foram definidos dois grupos: (i) *Simulation*, do qual constam as tabelas que definem os parâmetros globais de cada simulação e (ii) *Multi Options*, no qual estão agrupadas as tabelas correspondentes a opções disponibilizadas para determinados campos. O Quadro B.1 e o Quadro B.2 descrevem sucintamente as tabelas deste dois grupos. Todas as outras tabelas pertencem ao grupo de defeito *Unassigned Objects*, e são brevemente descritas no Quadro B.3.

Quadro B.1. Descrição das tabelas do grupo *Simulation*.

Tabela	**Descrição**
0SIMUL	Informação relativa à geometria das ferramentas
0SIMUL_tools	Detalhes acerca da geometria das ferramentas
0SIMULcase	Identificação e descrição global de cada simulação
0SIMULcaseDETAILS	Detalhes acerca da discretização associada a cada simulação
0SIMULcaseMATERIALS	Detalhes acerca do material associado a cada simulação

Quadro B.2. Descrição das tabelas do grupo *Multi Options*.

Tabela	**Descrição**
BCONcoord	Opções disponíveis para a variável COORD, definida na primeira parte do ficheiro DD3_bcon.dat
BCONcoordres	Opções disponíveis para a variável COORD, associada às restrições, definidas na primeira parte do ficheiro DD3_bcon.dat
COMPARISON	Opções disponíveis para a variável COND, associada às restrições, definidas na primeira parte do ficheiro DD3_bcon.dat
CONTACTstype	Opções disponíveis para a variável X, definida no ficheiro DD3_contact.dat
COORD	Opções disponíveis para a variável COORD, definida na opção *One Step Springback* do ficheiro DD3_bcon.dat e para a variável NOUT, definida no ficheiro DD3_phase.dat
FRICTIONgsp	Opções disponíveis para a definição das condições de contacto com atrito, definida no ficheiro DD3_contact.dat

Quadro B.2. Descrição das tabelas do grupo *Multi Options*.

Tabela	Descrição
FRICTIONipenvar	Opções disponíveis para o parâmetro de penalidade, definida no ficheiro DD3_contact.dat
INPUTmepopt	Opções disponíveis para a variável MEPOPT, definida no ficheiro DD3_input.dat
INPUTnstart	Opções disponíveis para a variável NSTART, definida no ficheiro DD3_input.dat
MESTityp	Opções disponíveis para a variável ITYP, definida no ficheiro DD3_mesh.dat
MESHmeshdr	Opções disponíveis para a variável MESHDR, definida no ficheiro DD3_mesh.dat
MESHnelmes	Opções disponíveis para a variável NELMES, definida no ficheiro DD3_mesh.dat
PHASEindout	Opções disponíveis para a variável INDOUT, definida no ficheiro DD3_phase.dat
PHASEnopr	Opções disponíveis para a variável NOPR, definida no ficheiro DD3_phase.dat
PHASEntyp	Opções disponíveis para a variável NTYP, definida no ficheiro DD3_phase.dat
SIMULCASEDETAILShex12	Opção de utilização de elementos hexaédricos de 12 nós, definida no ficheiro DD3_mesh.dat
XINIT	Opções disponíveis para a variável XINIT, definida na opção *One Step Springback* do ficheiro DD3_bcon.dat

Quadro B.3. Descrição das tabelas do grupo *Unassigned Objects*.

Tabela	Descrição
_MASKfiles	Definição das máscaras a utilizar para os ficheiros de entrada
1INPUT	Definição dos parâmetros numéricos e de controlo de saída de informação, de acordo com o ficheiro DD3_input.dat
1INPUTfbp	Definição dos deslocamentos para saída de informação, de acordo com o ficheiro DD3_input.dat
2BCON	Definição das condições de fronteira globais, de acordo com o ficheiro DD3_bcon.dat
2BCONOSS	Definição das condições de fronteira para *One Step Springback*, de acordo com o ficheiro DD3_bcon.dat
2BCONRES	Definição das restrições a associar às condições de fronteira globais, de acordo com o ficheiro DD3_bcon.dat

Quadro B.3. Descrição das tabelas do grupo *Unassigned Objects*.

Tabela	**Descrição**
3PHASE	Definição das condições iniciais para as ferramentas, de acordo com o ficheiro DD3_phase.dat
3PHASEheader	Definição das condições de controlo de cada fase, de acordo com o ficheiro DD3_phase.dat
3PHASElines	Definição das condições de controlo de cada ferramenta, em cada fase, de acordo com o ficheiro DD3_phase.dat
4MESH	Definição das discretizações disponíveis, incluindo o tipo de ficheiro de entrada
4MESH_NLAY	Definição dos parâmetros para a discretização com o gerador interno, de acordo com o ficheiro DD3_mesh.dat
4MESH_NZ	Definição dos parâmetros para a discretização com o gerador interno, de acordo com o ficheiro DD3_mesh.dat
5MAT_HD	Definição dos parâmetros para o comportamento elástico e lei de encruamento, de acordo com o ficheiro DD3_materX.dat
5MAT_HDparam	Descrição dos parâmetros para o comportamento elástico e lei de encruamento, de acordo com o ficheiro DD3_materX.dat
5MAT_YldCRIT	Definição dos parâmetros para o critério de plasticidade, de acordo com o ficheiro DD3_materX.dat
5MAT_YldCRITparam	Descrição dos parâmetros para o critério de plasticidade, de acordo com o ficheiro DD3_materX.dat
6CONTACTSET	Definição dos *sets* de contacto, de acordo com o ficheiro DD3_contact.dat
6FRICTION	Definição das condições de contacto com atrito, de acordo com o ficheiro DD3_contact.dat
PARAM	Definição das versões do DD3IMP disponíveis para a realização das simulações
ZFIELDS	Associação dos campos definidos em cada tabela com os ficheiros entrada

Os quadros seguintes apresentam a descrição dos campos de cada uma das tabelas utilizadas na base de dados, incluindo os campos e as chaves primárias e estrangeiras, realçando a respectiva correspondência na tabela Zfields. Opta-se por apresentar as tabelas com a mesma sequência com que são mencionadas no texto, de modo a garantir a proximidade na descrição dos campos.

Quadro B.4. Descrição dos campos da tabela 0SIMULcase e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	50
toolID	Variável caracter que define a geometria da ferramenta	51
SIMULoption	Campo para a descrição das principais características da simulação	52
SIMULnotes	Campo para a descrição das principais notas acerca da simulação	53
DESCR1	Variável caracter (A80) a incluir como cabeçalho dos ficheiros de entrada para a simulação	54
DESCR2	Variável caracter (A80) a incluir como cabeçalho dos ficheiros de entrada para a simulação	55

Quadro B.5. Descrição dos campos da tabela 0SIMUL e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
toolID	CHAVE PRIMÁRIA: Variável caracter que define a geometria da ferramenta	10
DD3IMP	Campo para a activação da opção de utilização do DD3IMP. Opções disponíveis na tabela PARAM	11
DD3LT	Campo para a activação da opção de utilização do DD3LT. Opções disponíveis na tabela PARAM	12
DESCR	Campo para a descrição das principais características da ferramenta	13
TOOLfile_BEZ	Campo para inserir um ficheiro externo (DD3_tool.dat) com a informação relativa à geometria das ferramentas descritas com superfícies de Bézier	14
TOOLfile_NAG	Campo para inserir um ficheiro externo (DD3_tool.msh) com a informação relativa à geometria das ferramentas descritas com superfícies Nagata. Utilizar apenas quando as várias ferramentas estão definidas num único ficheiro	15
TOOLfile_IGS	Campo para inserir um ficheiro externo (DD3_tool.igs) com a informação relativa à geometria das ferramentas descritas com superfícies Nagata, para cálculo dos vectores normais com base no ficheiro IGS. Utilizar apenas quando as várias ferramentas estão definidas num único ficheiro	16
OBS	Campo para incluir observações acerca da ferramenta	17

Quadro B.5. Descrição dos campos da tabela 0SIMUL e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
IMAGE	Campo para inserir um ficheiro externo com a imagem das ferramentas	
FILEaux1	Campo para inserir um ficheiro externo com descrição das ferramentas, casos de estudo associados às ferramentas, etc.	18
FILEaux2	Campo para inserir um ficheiro externo com descrição das ferramentas, casos de estudo associados às ferramentas, etc.	19
FILEaux3	Campo para inserir um ficheiro externo com descrição das ferramentas, casos de estudo associados às ferramentas, etc.	20
FILEaux4	Campo para inserir um ficheiro externo com descrição das ferramentas, casos de estudo associados às ferramentas, etc.	21

Quadro B.6. Descrição dos campos da tabela PARAM.

Campo	Descrição
KEY	CHAVE PRIMÁRIA: Variável inteira de identificação do programa a utilizar
VERSION	Variável caracter para a descrição da versão do programa
CODE	Variável caracter para identificação do programa a utilizar
DD3IMP	Campo para inserir o ficheiro corresponde ao programa
DD3LT	Campo para inserir o ficheiro corresponde ao programa
GIDhist	Campo para inserir informação referente à versão do GID
DATE	Campo para inserir a data de actualização

Quadro B.7. Descrição dos campos da tabela 0SIMUL_tools e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
toolID	CHAVE PRIMÁRIA: Variável caracter que define a geometria da ferramenta	40
TOOLnum	Variável inteira (I4), incremental, que define cada uma das ferramentas. O número máximo corresponde a NbTOOLs	41
TOOLname	Variável caracter (A16) que permite associar um nome a cada uma das ferramentas	42

Quadro B.7. Descrição dos campos da tabela 0SIMUL_tools e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
TOOLfile_NAG	Campo para inserir um ficheiro externo (DD3_tool.msh) com a informação relativa à geometria das ferramentas descritas com superfícies Nagata. Associa o ficheiro ao número da ferramenta	
TOOLfile_IGS	Campo para inserir um ficheiro externo (DD3_tool.igs) com a informação relativa à geometria das ferramentas descritas com superfícies Nagata, para cálculo dos vectores normais com base no ficheiro IGS. Associa o ficheiro ao número da ferramenta	

Quadro B.8. Descrição dos campos da tabela 0SIMULcaseMATERIALS e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	90
iMAT	Variável inteira que identifica o número do material. O valor mínimo é 1 e o valor máximo corresponde a NMATs (ver Quadro B.13)	91
MATlabel	Variável caracter que identifica a origem da identificação dos parâmetros dos modelos constitutivos	92
MATname	Variável caracter que identifica o material seleccionado	93
IDplaslaw	Variável caracter que identifica a lei de encruamento seleccionada	94
IDyldcrit	Variável caracter que identifica o critério de plasticidade seleccionado	95
MxRDOX	Variável real (F10.4) que define o ângulo [º] entre a direcção de laminagem e o eixo Ox	96
MxRDOZ	Variável real (F10.4) que define o ângulo [º] entre a direcção de normal ao plano de ortotropia e o eixo Oz	97

Quadro B.9. Descrição dos campos da tabela 5MAT_HDparam e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
IDplaslaw	CHAVE ESTRANGEIRA: Variável caracter que identifica a lei de encruamento seleccionada	550

Quadro B.9. Descrição dos campos da tabela 5MAT_HDparam e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
PlasLAW	Variável inteira (I4) que identifica a lei de encruamento seleccionada	551
DESCR	Campo para a descrição das principais características da lei de encruamento	552
EM	Campo associado ao módulo de elasticidade	553
PR	Campo associado ao coeficiente de Poisson	554
Y0	Campo associado à tensão limite de elasticidade	555
Param02	Campo associado a um determinado parâmetro da lei de encruamento	556
…		…
Param20	Campo associado a um determinado parâmetro da lei de encruamento	574
MATERx	Campo que define a máscara para o ficheiro DD3_materx.dat, de acordo com a lei de encruamento seleccionada	575

Quadro B.10. Descrição dos campos da tabela 5MAT_HD e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
MATlabel	CHAVE ESTRANGEIRA: Variável caracter que identifica a origem da identificação dos parâmetros dos modelos constitutivos	500
MATname	CHAVE ESTRANGEIRA: Variável caracter que identifica o material seleccionado	501
IDplaslaw	CHAVE ESTRANGEIRA: Variável caracter que identifica a lei de encruamento seleccionada	502
EM	Variável real (F10.1) utilizada para definir o valor do módulo de elasticidade	503
PR	Variável real (F10.4) utilizada para definir o valor do coeficiente de Poisson	504
Y0	Variável real (F10.4) utilizada para definir o valor da tensão limite de elasticidade	505
Param02	Variável real (F10.4) utilizada para definir o valor de um determinado parâmetro da lei de encruamento	506
…		…
Param20	Variável real (F10.4) utilizada para definir o valor de um determinado parâmetro da lei de encruamento	524

Quadro B.11. Descrição dos campos da tabela 5MAT_YldCRITparam e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
IDyldcrit	Variável caracter que identifica o critério de plasticidade seleccionado	650
yldCRIT	Variável inteira (I4) que identifica o critério de plasticidade seleccionado	651
DESCR	Campo para a descrição das principais características do critério de plasticidade	652
Param01	Campo associado a um determinado parâmetro da lei de encruamento	653
…		…
Param30	Campo associado a um determinado parâmetro da lei de encruamento	682
MATERx	Campo que define a máscara para o ficheiro DD3_materx.dat, de acordo com o critério de plasticidade seleccionado	683

Quadro B.12. Descrição dos campos da tabela 5MAT_YldCRIT e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
MATlabel	CHAVE ESTRANGEIRA: Variável caracter que identifica a origem da identificação dos parâmetros dos modelos constitutivos	600
MATname	CHAVE ESTRANGEIRA: Variável caracter que identifica o material seleccionado	601
IDyldcrit	CHAVE ESTRANGEIRA: Variável caracter que identifica o critério de plasticidade seleccionado	602
Param01	Variável real (F10.5) utilizada para definir o valor de um determinado parâmetro do critério de plasticidade	603
…		…
Param30	Variável real (F10.5) utilizada para definir o valor de um determinado parâmetro do critério de plasticidade	632

Quadro B.13. Descrição dos campos da tabela 0SIMULcaseDETAILS e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	60
MESHref	Variável caracter que define a geometria da chapa associada à ferramenta	61
MESHopt	Variável inteira que define a discretização seleccionada para a simulação. Caso a discretização seja gerada pelo malhador interno esta variável toma o valor 0	62
NMATs	Variável inteira que define o número máximo de materiais utilizados na simulação	63
NPH	Variável inteira que define o número máximo de fases da simulação	64
NbTOOLs	Variável inteira que define o número máximo de ferramentas utilizadas na simulação	65
NbBCIDs	Variável inteira que define o número máximo de condições de fronteira impostas em nós específicos da discretização, utilizadas na simulação	66
Blank_dimX	Variável real (F10.2) que define a dimensão inicial do corpo deformável na direcção Ox	67
Blank_dimY	Variável real (F10.2) que define a dimensão inicial do corpo deformável na direcção Oy	68
Blank_dimZ	Variável real (F10.2) que define a dimensão inicial do corpo deformável na direcção Oz	69
Scalling_Ox	Variável real (F10.2) que define o factor de escala a aplicar à discretização inicial, na direcção Ox	70
Scalling_Oy	Variável real (F10.2) que define o factor de escala a aplicar à discretização inicial, na direcção Oy	71
Scalling_Oz	Variável real (F10.2) que define o factor de escala a aplicar à discretização inicial, na direcção Oz	72
Shifting_Ox	Variável real (F10.2) que define o deslocamento a aplicar à discretização inicial, na direcção Ox	73
Shifting_Oy	Variável real (F10.2) que define o deslocamento a aplicar à discretização inicial, na direcção Oy	74
Shifting_Oz	Variável real (F10.2) que define o deslocamento a aplicar à discretização inicial, na direcção Oz	75
Rotation_1	Variável real (F10.2) que define a rotação a aplicar à discretização inicial	76
Rotation_2	Variável real (F10.2) que define a rotação a aplicar à discretização inicial	77

Quadro B.13. Descrição dos campos da tabela 0SIMULcaseDETAILS e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
Rotation_3	Variável real (F10.2) que define a rotação a aplicar à discretização inicial	78
NLayers	Variável inteira (I3) que define o número de camadas a construir para uma determinada discretização no plano, com o auxílio da aplicação Bi2Tri	79
HEX12	Variável inteira (I3) que deve ser activada caso se pretenda utilizar elementos finitos hexaédricos de 12 nós. Opções definidas na tabela SIMULCASEDETAILShex12	80

Quadro B.14. Descrição dos campos da tabela SIMULCASEDETAILShex12.

Campo	Descrição (DESC)	
HEX12	0	Não são considerados elementos hexaédricos de 12 nós
	1	São considerados elementos hexaédricos de 12 nós

Quadro B.15. Descrição dos campos da tabela 4MESH e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
MESHref	CHAVE ESTRANGEIRA: Variável caracter que define a geometria da ferramenta	400
MESHopt	CHAVE ESTRANGEIRA: Variável inteira que define a discretização seleccionada para a simulação. Caso a discretização seja gerada pelo malhador interno esta variável toma o valor 0	401
infoNOTES	Campo para a descrição das principais características da discretização	402
infoMESHimage	Campo para inserir uma imagem da discretização	403
infoNEDEF	Campo informativo acerca do número total de elementos finitos utilizados na discretização	404
infoNNDEF	Campo informativo acerca do número total de nós utilizados na discretização	405
infoNELMES	Campo informativo acerca do número de nós dos elementos finitos utilizados na discretização	406
infoITYP	Campo informativo acerca do tipo de integração dos elementos finitos utilizados na discretização. Opções definidas na tabela MESHityp	407

Quadro B.15. Descrição dos campos da tabela 4MESH e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
infoNMAT	Campo informativo acerca do número de materiais utilizados na discretização	408
1MESHDR	Variável inteira (I4) utilizada pelo gerador interno para definir a orientação da discretização. Opções definidas na tabela MESHmeshdr	409
1NELMES	Variável inteira (I4) utilizada pelo gerador interno para definir o número de nós dos elementos finitos utilizados na discretização. Opções definidas na tabela MESHnelmes	410
1NL	Variável inteira (I4) utilizada pelo gerador interno para definir o número total de elementos finitos a utilizar na direcção STH	411
1NC	Variável inteira (I4) utilizada pelo gerador interno para definir o número total de elementos finitos a utilizar na direcção SLEN	412
1NH	Variável inteira (I4) utilizada pelo gerador interno para definir o número total de elementos finitos a utilizar na direcção SHIG	413
1NMAT	Variável inteira (I4) utilizada pelo gerador interno para definir o número total de materiais a utilizar na discretização	414
1STH	Variável real (F10.4) que define a dimensão total da discretização na direcção OY (1MESHDR=1) ou OZ (1MESHDR=2)	415
1SLEN	Variável real (F10.4) que define a dimensão total da discretização na direcção OX	416
1SHIG	Variável real (F10.4) que define a dimensão total da discretização na direcção OZ (1MESHDR=1) ou OY (1MESHDR=2)	417
1NLAY	Variável inteira (I4) que define o número de zonas distintas da discretização na direcção OY (1MESHDR=1) ou OZ (1MESHDR=2)	418
1NZ	Variável inteira (I4) que define o número de zonas distintas da discretização na direcção OX	419
2mesh_file	Campo para inserir um ficheiro externo com a informação relativa à discretização	420

Quadro B.16. Descrição dos campos da tabela MESHmeshdr.

Campo	**Descrição (DESC)**	
MESHDR	0	Não é utilizado o gerador de malha interno
	1	É utilizado o gerador interno, sendo STH a dimensão na direcção OY, SLEN a dimensão na direcção OX e SHIG a dimensão na direcção OZ
	2	É utilizado o gerador interno, sendo STH a dimensão na direcção OZ, SLEN a dimensão na direcção OX e SHIG a dimensão na direcção OY

Quadro B.17. Descrição dos campos da tabela MESHnelmes.

Campo	**Descrição (DESC)**	
NELMES	4	Tetraedros lineares de 4 nós – Não estão a funcionar
	5	Piramidais (base quadrangular) lineares de 4 nós – Não está a funcionar
	6	Pentaedros lineares de 6 nós
	8	Hexaedros lineares de 8 nós
	10	Tetraedros quadráticos de 10 nós – Não estão a funcionar
	12	Hexaedros, lineares nas faces e quadráticos em espessura, de 12 nós
	20	Hexaedros quadráticos (serendipity) de 20 nós
	27	Hexaedros quadráticos de 27 nós

Quadro B.18. Descrição dos campos da tabela MESHityp.

Campo	**Descrição (DESC)**	
ITYP	1	Integração completa
	2	Integração reduzida
	3	Integração reduzida selectiva

Quadro B.19. Descrição dos campos da tabela 4MESH_NLAY e respectiva correspondência na tabela Zfields.

Campo	**Descrição**	**Tabela Zfields (ATn)**
MESHref	CHAVE ESTRANGEIRA: Variável caracter que define a geometria da ferramenta	
MESHopt	CHAVE ESTRANGEIRA: Variável inteira que define a discretização seleccionada para a simulação. Caso a discretização seja gerada pelo malhador interno esta variável toma o valor 0	

Quadro B.19. Descrição dos campos da tabela 4MESH_NLAY e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
1NLIN	Variável inteira (I4), incremental, associada a cada uma das 1NLAY zonas. O valor mínimo é 1 e o máximo é igual ao número total de zonas 1NLAY	430
1THL	Variável real (F10.4) que define a dimensão de cada uma das 1NLAY zonas. O somatório destas dimensões tem que ser igual a 1STH	431
1NLY	Variável inteira (I4) que define o número de elementos finitos de cada uma das 1NLAY zonas. O somatório destes números de elementos tem de ser igual a 1NL	432
1MAT	Variável inteira (I4) que define o número do material associado a cada uma das 1NLAY zonas.	433
1ITYP	Variável inteira (I4) que define o tipo de integração dos elementos finitos associado a cada uma das 1NLAY zonas. Opções definidas na tabela MESHityp	434

Quadro B.20. Descrição dos campos da tabela 4MESH_NZ e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
MESHref	CHAVE ESTRANGEIRA: Variável caracter que define a geometria da ferramenta	
MESHopt	CHAVE ESTRANGEIRA: Variável inteira que define a discretização seleccionada para a simulação. Caso a discretização seja gerada pelo malhador interno esta variável toma o valor 0	
1NLIN	Variável inteira (I4), incremental, associada a cada uma das 1NZ zonas. O valor mínimo é 1 e o máximo é igual ao número total de zonas 1NZ	440
1ELZ	Variável real (F10.4) que define a dimensão de cada uma das 1NZ zonas. O somatório destas dimensões tem que ser igual a 1SLEN	441
1NCZ	Variável inteira (I4) que define o número de elementos finitos de cada uma das 1NZ zonas. O somatório destes números de elementos tem de ser igual a 1NC	442

Quadro B.21. Descrição dos campos da tabela 1INPUT e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	101
NSTART	Variável inteira (I8) que controla o início do ciclo de incrementos. Opções definidas na tabela INPUTnstart	102
NEND	Variável inteira (I8) que define o número máximo de incrementos para realizar a simulação	103
NOUT	Variável inteira (I8) que define a frequência de gravação de um ficheiro, com base no número NST do incremento, do tipo 'info[nst].ufo'	104
IGID	Variável inteira (I8) que define a frequência de gravação de informação no ficheiro 'GIDsimu.res', com base no número NST do incremento.	105
INC	Variável inteira (I8) que controla a informação de saída	106
DEV	Variável inteira (I8) que controla a informação de saída	107
TOLEQ	Variável real (E10.3) que define a tolerância admissível para o equilíbrio (variável CRIT)	108
TEQOUT	Variável real (E10.3) que define a tolerância para as ferramentas a força imposta	109
RAPEQ	Variável real (E10.3) que define a tolerância para a validação de RMIN	110
TOLST	Variável real (E10.3) que define a tolerância admissível no ciclo iterativo de integração do modelo constitutivo	111
CUNL	Variável real (F10.4) que define a espessura virtual para a superfície limite de elasticidade na descarga	112
dampOSS	Variável real (F10.4) que define o coeficiente de amortecimento para a estratégia *One Step Springback*	113
IRMAX	Variável inteira (I8) que define o número máximo de iterações para a redução de RMIN	114
IEQMAX	Variável inteira (I8) que define o número máximo de iterações do ciclo iterativo de equilíbrio. Valor máximo admissível de 75	115
NMAXST	Variável inteira (I8) que define o número máximo de iterações do ciclo de integração do modelo constitutivo	116
DEMAX	Variável real (F10.4) que define o incremento máximo de deformação	117

Quadro B.21. Descrição dos campos da tabela 1INPUT e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
DWMAX	Variável real (F10.4) que define o incremento máximo de rotação	118
DSNMAX	Variável real (F10.2) que define o incremento máximo de tensão normal	119
DSTMAX	Variável real (F10.2) que define o incremento máximo de tensão tangencial	120
RINF	Variável real (F10.4) que define o valor mínimo admissível de RMIN	121
RSUP	Variável real (F10.4) que define o valor máximo admissível de RMIN	122
DFNMAX	Variável real (F10.2) que define o incremento máximo da componente normal da força de contacto	123
DFT1MAX	Variável real (F10.2) que define o incremento máximo da componente tangencial da força de contacto	124
DFT2MAX	Variável real (F10.2) que define o incremento máximo da componente tangencial da força de contacto	125
LEVEL	Variável inteira (I8) que define o nível inicial de pré-condicionamento para método iterativo de resolução de sistemas de equações lineares	126
TOLCGV	Variável real (E10.3) que define a tolerância admissível para a convergência do método iterativo de resolução de sistemas de equações lineares	127
MEPOPT	Variável inteira (I8) que define o módulo a utilizar no processo iterativo. Opções definidas na tabela INPUTmepopt	128
iphOSS	Variável inteira (I8) que define o número da fase correspondente à estratégia *One Step Springback*	129

Quadro B.22. Descrição dos campos da tabela INPUTnstart.

Campo	Descrição (DESC)		ORDER
NSTART	1	Inicia ou reinicia a simulação a partir de um ficheiro do tipo 'resume.ufo', 'resume.stage.ufo' e 'resume.trim.ufo'	10
	98	Reinicia a simulação a partir de um ficheiro do tipo 'resume.ufo' e armazena a informação relativa ao processo iterativo no ficheiro GID_DD3nst[NST].res	40
	99	Executa a estratégia *One Step Springback* a partir de um ficheiro do tipo 'resume.ufo'	50

Quadro B.23. Descrição dos campos da tabela INPUTmepopt.

Campo	Descrição (DESC)	
MEPOPT	1	Módulo elastoplástico consistente
	2	Módulo elastoplástico tangente
	3	Módulo elástico

Quadro B.24. Descrição dos campos da tabela 1INPUTfbp e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	160
IDline	Variável inteira, incremental, que define cada uma das saídas intermédias. O valor mínimo é 1 e o máximo é 50	161
FBP	Variável real (F10.4) que define o valor de deslocamento da ferramenta que controla a fase, associado à produção de um ficheiro de saída	162

Quadro B.25. Descrição dos campos da tabela 2BCON e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	200
NLIN	Variável inteira, incremental, associada a cada uma das condições de fronteira. O valor mínimo é 0	201
ERR	Variável real (E10.3) que define a tolerância admissível para impor a condição de fronteira num nó	202

Quadro B.25. Descrição dos campos da tabela 2BCON e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
A	Variável real (F10.4) que define o parâmetro A de acordo com a equação (2.1)	203
B	Variável real (F10.4) que define o parâmetro B de acordo com a equação (2.1)	204
C	Variável real (F10.4) que define o parâmetro C de acordo com a equação (2.1)	205
D	Variável real (F10.4) que define o parâmetro D de acordo com a equação (2.1)	206
COORD	Variável inteira (I4) que define a coordenada à qual é imposta a condição de fronteira. Opções definidas na tabela BCONcoord	207
NCR_ID	Variável inteira (I4) que define o tipo de condição de fronteira a ser imposta. Deslocamento nulo corresponde à opção 2	208
NRES	Variável inteira (I4) que define o número total de restrições a que a condição de fronteira NLIN está sujeita	209

Quadro B.26. Descrição dos campos da tabela BCONcoord.

Campo	Descrição (DESC)	
COORD	1	Coordenada Ox
	2	Coordenada Oy
	3	Coordenada Oz
	4	Todas as coordenada (Ox, Oy, Oz)

Quadro B.27. Descrição dos campos da tabela 2BCONRES e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	240
NLIN	CHAVE ESTRANGEIRA: variável inteira que associa a restrição à condição de fronteira imposta na tabela 2BCONRES	241
NLINRES	Variável inteira, incremental, que define cada uma das restrições a impor a cada um dos planos definidos na tabela 2BCONRES. O valor mínimo é 1 e o máximo é NRES	242

Quadro B.27. Descrição dos campos da tabela 2BCONRES e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
COORD	Variável inteira (I8) que define a coordenada à qual é imposta a condição de fronteira. Opções definidas na tabela BCONcoordres	243
COND	Variável inteira (I8) que define o tipo de condição associada à restrição. Opções definidas na tabela COMPARISON	244
VALUE	Variável real (F10.4) que define o valor de controlo para a restrição	245
ERR2	Variável real (E10.3) que define a tolerância admissível para impor a restrição num nó	246

Quadro B.28. Descrição dos campos da tabela BCONcoordres.

Campo	Descrição (DESC)	
COORD	1	Coordenada Ox
	2	Coordenada Oy
	3	Coordenada Oz
	5	Coordenada radial r

Quadro B.29. Descrição dos campos da tabela COMPARISON.

Campo	Descrição (DESC)	
COND	-1	Menor ou igual a VALUE
	0	Igual a VALUE
	1	Maior ou igual a VALUE

Quadro B.30. Descrição dos campos da tabela 2BCONOSS e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	220
NLINOSS	Variável inteira, incremental, que define cada uma das restrições a impor. O valor mínimo é 1	221
X	Variável real (F10.4) que define a coordenada Ox para determinar o nó mais próximo ao qual deve ser imposta a condição de fronteira	222

Quadro B.30. Descrição dos campos da tabela 2BCONOSS e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
Y	Variável real (F10.4) que define a coordenada Oy para determinar o nó mais próximo ao qual deve ser imposta a condição de fronteira	223
Z	Variável real (F10.4) que define a coordenada Oz para determinar o nó mais próximo ao qual deve ser imposta a condição de fronteira	224
COORD	Variável inteira (I8) que define a coordenada à qual é imposta a condição de fronteira. Opções definidas na tabela COORD	225
XINIT	Variável inteira (I4) que define se a condição é imposta em função das coordenadas inicias ou finais. Opções definidas na tabela XINIT	226

Quadro B.31. Descrição dos campos da tabela COORD.

Campo	Descrição (DESC)	
COORD	1	Coordenada Ox
	2	Coordenada Oy
	3	Coordenada Oz

Quadro B.32. Descrição dos campos da tabela XINIT.

Campo	Descrição (DESC)	
XINIT	0	Coordenadas finais
	1	Coordenadas iniciais

Quadro B.33. Descrição dos campos da tabela 3PHASE e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	300
ENTI	Variável inteira (I4) que define o número associado a cada ferramenta. Esta definição deve estar de acordo com o TOOLnum definido no Quadro B.7	301
DIS0X	Variável real (G8) que define o deslocamento inicial a impor à ferramenta na direcção Ox	302
DIS0Y	Variável real (G8) que define o deslocamento inicial a impor à ferramenta na direcção Oy	303

Quadro B.33. Descrição dos campos da tabela 3PHASE e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
DIS0Z	Variável real (G8) que define o deslocamento inicial a impor à ferramenta na direcção Oz	304
TOOLname	Variável caracter (A16) que permite associar um nome a cada uma das ferramentas. Este campo só deve ser preenchido se não for utilizada a tabela 0SIMUL_Tools(ver Quadro B.7)	305
BCID	Variável inteira (I4) utilizada para impor condições de fronteira adicionais	306
iSET	Variável inteira (I4) utilizada para associar a cada ferramenta o *set* de contacto	307
lim-	Variável real (F8.4) utilizada para impor um valor mínimo para o deslocamento da ferramenta	308
lim+	Variável real (F8.4) utilizada para impor um valor máximo para o deslocamento da ferramenta	309

Quadro B.34. Descrição dos campos da tabela 3PHASEheader e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	320
IPH	Variável inteira (I4), incremental, que define o número da fase. O valor mínimo é 1 e o máximo é NPH (de acordo com o Quadro B.13)	321
DELT	Variável real (F10.4) que define a dimensão do tamanho do incremento para a fase de previsão, da fase IPH	322
NOUT	Variável inteira (I4) que define o número da ferramenta que controla a fase IPH. O valor mínimo é 1 e o máximo é o número de ferramentas	323
JD	Variável inteira (I4) que define a direcção de controlo da fase IPH. Opções definidas de acordo com a tabela COORD (ver Quadro B.31)	324
NTYP	Variável inteira (I4) que define o tipo de controlo da fase IPH. Opções definidas de acordo com a tabela PHASEntyp	325
NOPR	Variável inteira (I4) que define a condição de controlo para a validação do final da fase IPH. Opções definidas de acordo com a tabela PHASEnopr	326

Quadro B.35. Descrição dos campos da tabela PHASEntyp.

Campo	**Descrição (DESC)**	
NTYP	1	Fase controlada por deslocamento
	2	Fase controlada por força
	3	Fase controlada por perda de contacto

Quadro B.36. Descrição dos campos da tabela PHASEnopr.

Campo	**Descrição (DESC)**	
NOPR	1	Menor ou igual ao valor imposto para o final de fase
	2	Maior ou igual ao valor imposto para o final de fase

Quadro B.37. Descrição dos campos da tabela 3PHASElines e respectiva correspondência na tabela Zfields.

Campo	**Descrição**	**Tabela Zfields (ATn)**
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	350
IPH	CHAVE ESTRANGEIRA: Variável inteira (I4), incremental, que define o número da fase. O valor mínimo é 1 e o máximo é NPH (de acordo com o Quadro B.13)	351
ENTI	CHAVE ESTRANGEIRA: Variável inteira (I4) que define o número associado a cada ferramenta. Esta definição deve estar de acordo com o TOOLnum definido no Quadro B.7	352
INDOUTx	Variável inteira (I4) que define o estatuto da ferramenta na fase IPH, na direcção Ox. Opções definidas de acordo com a tabela PHASEindout	353
INDOUTy	Variável inteira (I4) que define o estatuto da ferramenta na fase IPH, na direcção Oy. Opções definidas de acordo com a tabela PHASEindout	354
INDOUTz	Variável inteira (I4) que define o estatuto da ferramenta na fase IPH, na direcção Oz. Opções definidas de acordo com a tabela PHASEindout	355
DISINTx	Variável real (G8) que define o valor do deslocamento total imposto à ferramenta na fase IPH, na direcção Ox	356
DISINTy	Variável real (G8) que define o valor do deslocamento total imposto à ferramenta na fase IPH, na direcção Oy	357

Quadro B.37. Descrição dos campos da tabela 3PHASElines e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
DISINTz	Variável real (G8) que define o valor do deslocamento total imposto à ferramenta na fase IPH, na direcção Oz	358
EFFIMPx	Variável real (F8.2) que define o valor da força total imposta à ferramenta na fase IPH, na direcção Ox	359
EFFIMPy	Variável real (F8.2) que define o valor da força total imposta à ferramenta na fase IPH, na direcção Oy	360
EFFIMPz	Variável real (F8.2) que define o valor da força total imposta à ferramenta na fase IPH, na direcção Oz	361
D	Variável inteira (I4)	362
P	Variável inteira (I4)	363

Quadro B.38. Descrição dos campos da tabela PHASEindout.

Campo	Descrição (DESC)	
INDOUT	0	Ferramenta inactiva
	1	Ferramenta parada
	2	Ferramenta com deslocamento imposto
	3	Ferramenta a força imposta (constante)
	4	Ferramenta a força imposta (linear, proporcional ao deslocamento da ferramenta que controla a fase)
	10	Ferramenta *Glue*: parada
	20	Ferramenta *Glue*: com deslocamento imposto
	30	Ferramenta *Glue*: com força imposta

Quadro B.39. Descrição dos campos da tabela 6CONTACTSET e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	700
NCONTSET	Variável inteira (I5), incremental, que define o número do *set* de contacto	701
A	Variável real (F10.4) que define o parâmetro *A* de acordo com a equação (2.1), (2.2) ou (2.3)	702
B	Variável real (F10.4) que define o parâmetro *B* de acordo com a equação (2.1), (2.2) ou (2.3)	703

Quadro B.39. Descrição dos campos da tabela 6CONTACTSET e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
C	Variável real (F10.4) que define o parâmetro C de acordo com a equação (2.1), (2.2) ou (2.3)	704
D	Variável real (G10.4) que define o parâmetro D de acordo com a equação (2.1), (2.2) ou (2.3)	705
STYPE	Variável caracter (A4) que define o tipo de superfície. Opções definidas de acordo com a tabela CONTACTstype	706

Quadro B.40. Descrição dos campos da tabela CONTACTstype.

Campo	Descrição (DESC)		ORDER
STYPE	E	Cilindro de base elíptica	40
	P	Plano	1
	R	Cilindro	10

Quadro B.41. Descrição dos campos da tabela 6FRICTION e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
caseID	CHAVE PRIMÁRIA: variável inteira, incremental, que deve ser definida para cada nova simulação	750
NLIN	Variável inteira, incremental, que define o número de condições de contacto com atrito	751
GSP	Variável caracter (A4) que identifica o tipo de condição. Opções definidas de acordo com a tabela FRICTIONgsp	752
SetNUM	Variável inteira (I4) que define o número do *set* de contacto, para o qual a condição é válida. O valor mínimo é 1 e o máximo corresponde ao máximo de NCONTSET (ver Quadro B.39)	753
FPatch	Variável inteira (I4) que define o número mínimo da superfície, para o qual a condição é válida	754
LPatch	Variável inteira (I4) que define o número máximo da superfície, para o qual a condição é válida	755
FricCoef	Variável real (F10.4) que define o valor constante para o coeficiente de atrito de Coulomb	756
CPen	Variável real (F10.4) que define o valor constante do parâmetro de penalidade	757

Quadro B.41. Descrição dos campos da tabela 6FRICTION e respectiva correspondência na tabela Zfields.

Campo	Descrição	Tabela Zfields (ATn)
IPenvar	Variável inteira (I10) que selecciona a lei de variação para o(s) parâmetro(s) de penalidade. Opções definidas de acordo com a tabela FRICTIONgsp	758
CPenD	Variável real (F10.4) que define o valor constante do parâmetro de penalidade para o contacto, caso seja activada a opção IPenvar	759

Quadro B.42. Descrição dos campos da tabela FRICTIONipenvar.

Campo	Descrição (DESC)	
IPenvar	0	Valor por defeito
	1	Valor variável para CPen de acordo com a expressão proposta por P. Alart
	2	Valor variável para CPen de acordo com a expressão proposta por P. Alart para o lagrangiano aumentado
	3	Valor constante diferente na direcção normal (CPenD) e tangencial (CPen)
	4	Valor variável diferente na direcção normal (CPenD) e tangencial (CPen) de acordo com a expressão proposta por P. Alart
	5	Valor variável diferente na direcção normal (CPenD) e tangencial (CPen) de acordo com a expressão proposta por P. Alart para o lagrangiano aumentado

Quadro B.43. Descrição dos campos da tabela FRICTIONgsp.

Campo	Descrição (DESC)		ORDER
GSP	G	Condições globais	1
	S	Condições para o *set* de contacto identificado em SetNUM	10
	P	Condições para o conjunto de superfícies do domínio [FPatch, LPatch]	40

Quadro B.44. Descrição dos campos da tabela _MASKfiles.

Campo	Descrição
key	Campo utilizado para identificar zonas distintas do ficheiro de entrada
DD3_bcon	Campo utilizado para definir a máscara de cada zona do ficheiro DD3_bcon
DD3_input	Campo utilizado para definir a máscara de cada zona do ficheiro DD3_input

Quadro B.44. Descrição dos campos da tabela _MASKfiles.

Campo	Descrição
DD3_materx	Campo utilizado para definir a máscara de cada zona do ficheiro DD3_materx
DD3_mesh	Campo utilizado para definir a máscara de cada zona do ficheiro DD3_mesh.dat, necessário para o gerador interno
DD3_mesh_bin	Campo utilizado para definir a máscara do ficheiro DD3_mesh.bin
DD3_phase	Campo utilizado para definir a máscara de cada zona do ficheiro DD3_phase
DD3_tool	Campo utilizado para definir diferentes tipos de ficheiros de entrada para a definição da geometria das ferramentas
DD3_contact	Campo utilizado para definir a máscara de cada zona do ficheiro DD3_contact

Quadro B.45. Descrição dos campos da tabela ZFIELDS.

Campo	Descrição
ATn	Variável inteira, incremental, que permite associar cada campo de uma tabela a uma posição única nesta tabela
FIELD	Variável caracter que associa o nome da tabela com o campo da tabela
TABLE	Variável caracter que associa o nome da tabela
DIM	Variável inteira que define a dimensão do campo
FORMAT	Variável caracter que associa a cada campo um determinado formato
GUIname	Variável caracter que associa o nome a ser utilizado na aplicação
DESCfield	Variável caracter que permite a introdução de uma descrição para o campo
VALUEmax	Variável real para a definição do valor máximo do campo
VALUEmin	Variável real para a definição do valor mínimo do campo
TABLEfield	Variável caracter que associa o nome campo da tabela