2019

Especificação Técnica:

Formato P3D aplicado ao simulador SMH

Versão 1.0

1. ESPECIFICAÇÃO TÉNICA

A seguir apresentaremos a especificação técnica do P3D para uso com o SMH.

1.1.CONVENÇÕES

As seguintes convenções foram adotadas no detalhamento da especificação.

ITEM	DESCRIÇÃO	LAYOUT	FORMATO	EXEMPLO
Descrições	Descrição do objeto/atributo, bem a unidade, e intervalo admitido de seus atributos.	**– Cor Verde e Sublinhado**	**DESCRIÇÃO | UNIDADE | DEFAULT**	– **Contante gravitacional | m/s^2 | 9.806 |]0, ..[** – **Coordenada x do ponto da curva | m**
Tipo	Tipo que representa o atributo do objeto na API.	– *Cor Azul e Itálico*	*TIPO*	– *string* – *double* – *int*
Fim de Seção	Texto que marca o fim do conjunto de informações de um objeto.	– *Cor Verde Claro e itálico*	*end_NOME_DO_OBJETO*	– end_MODEL
Enumeráveis/ Intervalos	Representação de tipos enumeráveis e intervalos de valores de objetos.	– Cor Vermelho	– <Valor1 | Valor 2 | .,.> – (Início .. Fim)	– <SGO– DGN | DAT | MG> – (1..*)
Informação Opcional	Marcação em objetos/atributos opcionais.	- Fundo Verde		– TEXTURE_FILENAME
Elemento descontinuado	Marcação em objetos/atributos descontinuados.	~~– Elemento tachado~~		- ~~SIMPLIFIED_MASS_MATRIX~~

MODEL
{
WATER_DEPTH *double* **Tamanho da lâmina de água (m).**
~~WAMIT~~ < "YES" | "NO" > **Chave que controla o cálculo do WAMIT em tempo de simulação. Estado padrão é "NO".**
~~SIMPLIFIED_MASS_MATRIX~~ < "YES" | "NO" > **Chave que controla o uso da matriz de massa simplificada 6 x 6. Estado padrão é "NO".**
~~INITIAL_SUPER_DAMPING~~ < "YES" | "NO" > **Chave que controla o uso de amortecimento excessivamente grande durante o início da simulação. Estado padrão é "NO".**
EVAL_OF_1ST_ORDER_MOTION < 0 | 1 | 2> **Chave que controla o modo de cálculo das força de primeira ordem de onda. 0 –> RAO | 1 > RAO + Força | 2 -> Força.**
COORDINATE_SYSTEM
{
~~GEOREF~~ *Vector2D* **Chave que controla (lat | lon).**
} *end_COORDINATE_SYSTEM*
TIME_INTEGRATION
{
INITIAL *double* **Tempo inicial da simulação (s).**
FINAL *double* **Tempo final da simulação (s).**
STEP *double* **Passo de integração (s).**
STEP_BETWEEN_PRINTS *int* **Quantidade de passos de integração executados entre cada saída no relatório.**
RAMP *double* **Instante de tempo em que a rampa do integrador termina (s).**
~~RAMP_DURATION_STATIC~~ *int* **Quantidade de passos para execução da rampa do integrador da análise estática.**
} *end_TIME_INTEGRATION*
SEA_BOTTOM
{
P1 *Vector3D* **Ponto para definição do plano do fundo do mar (m | m | m).**
P2 *Vector3D* **Ponto para definição do plano do fundo do mar (m | m | m).**
P3 *Vector3D* **Ponto para definição do plano do fundo do mar (m | m | m).**
} *end_SEA_BOTTOM*
CURRENTS
{
CURRENT
{
WATER_DENSITY *double* **Densidade da água para a corrente (t/m³).**

PROFILE_DEPTH_VEL_DIR_FLUCT_PER_ANG *Vector6D* **Perfil de corrente: Velocidade (m/s), Profundidade (m), Direção de propagação NED (°), Flutuação (adm), Período (s), Direção de propagação ENU (°).**
GEOMETRIA_CANAL *MatrixNxM* **Dados para canais estreitos (???).**
VELOCITY_FIELD
{
SHADOW_CURRENT_POSITION_X *VectorND* **Pontos X do grid do mapa de corrente (m | m | ...).**
SHADOW_CURRENT_POSITION_Y *VectorMD* **Pontos Y do grid do mapa de corrente (m | m | ...).**
SHADOW_CURRENT_VELOCITY_X *MatrixNxM* **Componente X da velocidade da corrente (m/s | m/s | ...).**
SHADOW_CURRENT_VELOCITY_Y *MatrixNxM* **Componente Y da velocidade da corrente (m/s | m/s | ...).**
} *end_VELOCITY_FIELD*
} *end_CURRENT*
} *end_CURRENTS*
WINDS
{
WIND
{
AIR_DENSITY – *double* **Densidade do ar para o vento (t/m³).**
VELOCITY – *double* **Velocidade do vento a 10 m (m/s).**
ANGLE - *double* **Direção de propagação do vento ENU (°).**
SPECTRUM - < “API” | ”DAVENPORT” | “HARRIS” | “KAIMAL” | “NPD” | “OCHISHIN” | “QUEUFFEULOU” | “REGULAR” | “WILLS” > **Tipo do espectro do vento.**
} *end_WIND*
} *end_WINDS*
WAVES
{
WAVE
{
SPECTRUM - < “GAUSSIAN” | ”JONSWAP” | “PIERSON” | “REGULAR” > **Tipo do espectro da onda.**
PERIOD – *double* **Período de pico do espectro (s).**
HEIGHT – *double* **Altura significativa do espectro (m).**
ANGLE - *double* **Direção de propagação da onda ENU (°).**
SEED - *double* **Semente para geração das fases aleatórias do espectro (adm).**
GAMA - *double* **Parâmetro do espectro de JONSWAP (adm).**

ALFA - *double* **Parâmetro do espectro de JONSWAP (adm).**
SMAX – *double* **Parâmetro para cálculo do espalhamento da onda (adm). Ativa o espalhamento quando presente.**
FORCE_FIELD
{
SHADOW_WAVE_POSITION_X *VectorND* **Pontos X da grade do mapa de onda (m | m | ...).**
SHADOW_WAVE_POSITION_Y *VectorMD* **Pontos Y da grade do mapa de onda (m | m | ...).**
SHADOW_WAVE_INTENSITY *MatrixNxM* **Multiplicador da altura significativa em cada ponto da grade (adm | adm | ...).**
SHADOW_WAVE_ANGLE *MatrixNxM* **Direção de propagação da onda em cada ponto da grade ENU (° | ° | ...).**
}
} *end_WAVE*
} *end_WAVES*
SWELLS
{
SWELL
{
SPECTRUM - < "GAUSSIAN" | "JONSWAP" | "PIERSON" | "REGULAR" > **Tipo do espectro do swell.**
PERIOD – *double* **Período de pico do espectro (s).**
HEIGHT – *double* **Altura significativa do espectro (m).**
ANGLE - *double* **Direção de propagação do swell ENU (°).**
SEED - *double* **Semente para geração das fases aleatórias do espectro (adm).**
GAMA - *double* **Parâmetro do espectro de JONSWAP (adm).**
ALFA - *double* **Parâmetro do espectro de JONSWAP (adm).**
SMAX – *double* **Parâmetro para cálculo do espalhamento do swell (adm). Ativa o espalhamento quando presente.**
FORCE_FIELD
{
SHADOW_SWELL_POSITION_X *VectorND* **Pontos X da grade do mapa de onda (m | m | ...).**
SHADOW_SWELL_POSITION_Y *VectorMD* **Pontos Y da grade do mapa de onda (m | m | ...).**
SHADOW_SWELL_INTENSITY *MatrixNxM* **Multiplicador da altura significativa em cada ponto da grade (adm | adm | ...).**
SHADOW_SWELL_ANGLE *MatrixNxM* **Direção de propagação do swell em cada ponto da grade ENU (° | ° | ...).**

}
} *end_SWELL*
} *end_SWELLS*

ENVIRONMENT_CONDITIONS
{
GRAVITY *double* **Aceleração da gravidade (m/s²).**
COMBINATION
{
COMBINATION_ID < 1 > **ID da condição ambiental. No caso do SMH deve ser sempre 1.**
LINE_NO *int* **ID local da linha que será rompida. Zero caso não haja linha para romper.**
TIME_BREAK *double* **Instante de tempo em que a linha será rompida (s).**
} *end_COMBINATION*
} *end_ENVIRONMENT_CONDITIONS*
VESSEL (1..*)
{
TYPE < MONOBUOY | MOONPOOL | PLATFORM | SHIP > **Rótulo correspondente ao tipo do corpo. MOONPOOL e PLATFORM causam lógicas diferentes no simulador.**
NAME *string* **Nome do corpo.**
VESSEL_ID *int* **ID local do corpo. Único entre todos os corpos.**
GLOBAL_ID *int* **ID global do corpo. Único em todo o P3D.**
OWNER *int* **ID global do corpo ao qual este está ligado caso ele seja um Moonpool. "0" caso contrário.**
BEAM *double* **Boca. Comprimento ao longo do eixo y (m).**
HEIGHT *double* **Altura. Comprimento ao longo do eixo z medido a partir do nível da água (m).**
LENGTH *double* **Comprimento. Comprimento ao longo do eixo x (m).**
LOCAL_SYSTEM *Vector6D* **Posição inicial da quilha à meia nau do corpo no sistema de coordenadas inercial (global) (m | m | m | ° | ° | °). As posições verticais (heave, roll e pitch) não devem ser alteradas.**
K1 *double* **Constante para o cálculo de shear force.**
K2 *double* **Constante para o cálculo de shear force.**
K3 *double* **Constante para o cálculo de shear force.**
SPD
{
PID_STANDARD_GAINS
{
Pxyyaw *Vector3D* **Ganho proporcional do controlador PID (kN/m | kN/m | kN/m).**
Ixyyaw *Vector3D* **Ganho integral do controlador PID (kN/(m.s) | kN/(m.s) | kN/(m.s)).**
Dxyyaw *Vector3D* **Ganho derivativo do controlador PID (kN.s/m | kN.s/m | kN.s/m).**

} *end_PID_STANDARD_GAINS*
NOTCH_GAINS
{
ZETA *double* **Parâmetro zeta do filtro Notch (adm).**
W1 *double* **Primeira frequência natural do filtro Notch (rad/s).**
W2 *double* **Segunda frequência natural do filtro Notch (rad/s).**
W3 *double* **Terceira frequência natural do filtro Notch (rad/s).**
}
AUTOMATIC_CONTROL_GAINS < "YES" | "NO" > **Chave que controla o cálculo automático do ganho do PID. Caso "NO" utiliza os que são fornecidos no P3D.**
AUTOMATIC_FILTER_GAINS < "YES" | "NO" > **Chave que controla o cálculo automático do filtro. Caso "NO" utiliza as frequências que são fornecidas no P3D.**
PROPULSION
{
PROP (1..*)
{
ID *int* **ID do propulsor. No caso do SMH deve seguir a convenção de acordo com o tipo de propulsor.**
NAME *string* **Nome do propulsor.**
POSITION_TYPE < "AZIMUTE_FIXED" | "AZIMUTE_FREE" > **Chave que define se o propulsor é azimutal ou fixo.**
CONTROL_TYPE < "VARIABLE_PITCH" | "VARIABLE_PITCH_ROTATION" | "VARIABLE_ROTATION" > **Chave que define se o propulsor e rotação controlada ou passo controlado ou ambos.**
MAX_PERP_VEL *double* **Velocidade máxima perpendicular da água no propulsor (m/s).**
X_PROP *double* **Posição do propulsor no eixo x local do corpo (em relação à quilha a meia nau) (m).**
Y_PROP *double* **Posição do propulsor no eixo y local do corpo (em relação à quilha a meia nau) (m).**
Z_PROP *double* **Posição do propulsor no eixo z local do corpo (em relação à quilha a meia nau) (m).**
ALPHA_PROP *double* **Ângulo do propulsor em relação ao eixo local do corpo (°). Deve ser fornecido para propulsores fixos.**
DIAMETER *double* **Diâmetro do propulsor (m).**
ROTATION **Rotação máxima do propulsor caso o propulsor seja rotação controlada ou rotação de trabalho caso o propulsor seja passo controlado (rps).**
P_D *double* **Passo da hélice do propulsor caso o propulsor seja rotação controlada ou passo máximo caso o propulsor seja passo controlado (adm).**
POT_MAX *double* **Potencia máxima do propulsor (kW).**

EFFIC *double* [0, 1] **Eficiência total do propulsor (adm).**
TMAX_THROTTLE *double* **Tempo que o propulsor leva para acelerar do 0 até a rotação máxima (s).**
TIME_TO_FULL_PITCH *double* **Tempo que o propulsor leva para ir de passo zero até passo máximo (s). Deve ser fornecido para propulsores do tipo passo controlado.**
TMAX_TURN *double* **Tempo que o propulsor leva para dar uma volta de 360° (s). Deve ser fornecido para propulsores azimutais.**
INVERSION_KEY < "YES" | "NO" > **Chave que controla se o propulsor pode ser revertido.**
DEAD_ZONES *MatrixNx2* **Matriz que define a(s) zona(s) proibidas. Cada linha da matriz é uma zona proibida com ângulo de início e ângulo de término da zona (° | °).**
KRE *double* [0, 1] **Multiplicador da força produzida pelo propulsor quando acionado a ré (adm).**
TABLE_POLY < "TABLE" | "POLI" > **Chave que indica o uso do polinômio ou da tabela.**
PROP_TABLE *MatrixNx3* **Tabela que relaciona J com Kq e Kt. Deve ser fornecido caso a chave esteja selecionando TABLE.**
PROP_POLY *MatrixNx4* **Coeficientes do polinômio que relaciona J com Kq e Kt. Deve ser fornecido caso a chave esteja selecionando POLY.**

} *end_PROP*

RUDDER (1..*)

{

ID *int* **ID do leme. No caso do SMH deve seguir a convenção que adota 72 como ID inicial para os lemes.**
NAME *string* **Nome do leme.**
X_RUDDER *double* **Posição do leme no eixo x local do corpo (em relação à quilha a meia nau) (m).**
Y_RUDDER *double* **Posição do leme no eixo y local do corpo (em relação à quilha a meia nau) (m).**
Z_RUDDER *double* **Posição do lema no eixo z local do corpo (em relação à quilha a meia nau) (m).**
AREA_SUB *double* **Área do leme que está em contato com a água (m²).**
ALPHA_MAX *double* **Fator de esteira do leme (°).**
TMAX_TURN *double* **Tempo que o leme leva para ir de -ALPHA_MAX até ALPHA_MAX (s).**
PROP_ASSOCIADO *int* **ID do propulsor ao qual o leme está associado.**
WAKE_FACTOR *double* **Fator de esteira do leme (adm).**
REDUCTION_FACTOR_INVERSE_THRUST *double* **Fator de redução de transmissão de fluxo do propulsor para o leme a ré. Valor padrão: 0.0 (adm).**
RUDDER_TABLE *MatrixNx3* **Relaciona um ângulo do leme com os coeficientes de *drag* e *lift*. (° | adm | adm).**

} *end_PROP*

} *end_PROPULSION*

} *end_SPD*

FAIR_LEADS
{
    FAIR_LEAD (1..*)
    {
        POSITION *Vector3D* **Posição do *fairlead* em relação à quilha à meia nau do navio (m | m | m).**
        TURRET < 0 | 1 > **Chave que controla se o *fairlead* é um Turret.**
        CONNECTED_LINES *VectorND* **Vetor com os ID globais das linha conectadas neste *fairlead*.**
    } *end_FAIR_LEAD*
} *end_FAIR_LEADS*
INIT_POS *Vector6D* **Delta de posição aplicado ao corpo no primeiro passo de integração. (m | m | m | ° | ° | °).**
INIT_VEL *Vector6D* **Velocidade inicial do corpo (m/s | m/s | m/s | °/s | °/s | °/s).**
FORCE_DIRECTION < 1 | 2 > **Chave que controla se o a força aplicada ao corpo é dada no sistema global (1) ou local (2).**
TOWING_POINT *Vector3D* **Ponto de aplicação da força em relação a quilha à meia nau. (m | m | m).**
TOWING_FORCE *Vector3D* **Força aplicada ao corpo (kN | kN | kN).**
CG_WAMIT *Vector3D* **Posição do centro de gravidade no sistema de coordenadas do WAMIT (m | m | m).**
~~BODY_WAMIT~~ *Vector3D* **??? (m | m | m).**
BODY_WAMIT_PHI *double* **Rotação em z da do sistema de coordenadas do WAMIT (°).**
CG_WRT_LOCAL *Vector3D* **Posição do centro de gravidade em relação ao sistema de coordenadas local (m | m | m).**
WAMITAXIS_WRT_LOCAL *Vector3D* **Posição do sistema de coordenadas do WAMIT em relação ao sistema de coordenadas local (m | m | m).**
DISPLACEMENT_VOLUME *double* **Volume deslocado pelo corpo (m^3).**
FLOAT_CENTER *Vector3D* **Centro de flutuação. Ponto onde é aplicada a força de restauração. (m | m | m).**
~~MDL_FILE~~ *string* **Caminho para o arquivo de modelo .mdl. Utilizado pelas interfaces PreA3D e PreTPN. Era utilizado pelo TPN Anal.**
~~DFX_FILE~~ *string* **Caminho para o arquivo de modelo .dfx. Era utilizado por versões antigas do visualizador.**
~~GDF_FILE~~ *string* **Caminho para o arquivo .gdf de malha do WAMIT.**
TURRET < 0 | 1 > **Chave que indica se há Turret no modelo.**
TURRET_LOCAL_COORDINATES *Vector3D* **Posição do Turret em relação ao sistema de coordenadas local. (m | m | m).**
DRAFT *double* **Calado do corpo (m).**
EXPONENTIAL_LINEAR_DRAG *double* **Fator exponencial de multiplicação da velocidade (adm).**
KEY_SQUATTING < “YES” | “NO” > **Chave que controla se a força de *squat* será calculada. Estado padrão é “NO”.**
KEY_STRAIT_CANAL < “YES” | “NO” > **Chave que controla se será considerado o uso de canais estritos. Estado padrão é “NO”.**
KEY_ADDED_MASS < “YES” | “NO” > **Chave que controla se será considerado o incremento na massa adicional devido a águas rasas. Estado padrão é “NO”.**
~~KEY_HYDRO_DERIVED~~ < “YES” | “NO” > **Chave obsoleta. Uso não documentado.**

~~KEY_CY_CR~~ < “YES” | “NO” > **Chave obsoleta. Uso não documentado.**
SHALLOW_WATERS_MAGNIFICATION *MatrixNxM* **Dados para o cálculo dos esforços devido a águas rasas (???).**
MATRIX_FY_BARRA *MatrixNxM* **Dados para o cálculo dos esforços devido a canais estritos (???).**
ALPHA_X_BARRA_STRAIT_CANAL *MatrixNxM* **Dados para o cálculo dos esforços devido a canais estritos (???).**
ADDEDMASS *Matrix6x6* **Matriz de massa adicional simplificada. Diagonal principal (t | t | t | t.m² | t.m² | t.m²).**
POTENTIAL_DAMPING_COEFFICIENTS *Matrix6x6* **Matriz de amortecimento potencial simplificada. Diagonal principal (t/s | t/s | t/s | t.m²/s | t.m²/s | t.m²/s).**
GLOBAL_MASS *Matrix6x6* **Matriz de massa. Diagonal principal (t | t | t | t.m² | t.m² | t.m²).**
RESTCOEFS *Matrix6x6* **Matriz de restauração. Coeficientes da diagonal principal (t/s² | t.m²/s² | t.m²/s²).**
EXTERNAL_DAMPING *Matrix6x6* **Matriz de amortecimento externo introduzido para o cálculo do WAMIT. Diagonal principal (t/s | t/s | t/s | t.m²/s | t.m²/s | t.m²/s).**
LINEAR_DAMPING *Matrix6x6* **Matriz de amortecimento linear. Diagonal principal (t/s | t/s | t/s | t.m²/s | t.m²/s | t.m²/s).**
QUADRATIC_DAMPING *Matrix6x6* **Matriz de amortecimento quadrático. Diagonal principal (t/m | t/m | t/m | t.m² | t.m² | t.m²).**
MULTIPLIERS *Matrix8x6* **Matriz de multiplicadores de forças. LINHA: 1 – Corrente; 2 – Vento; 3 – Onda primeira ordem; 4 – Onda deriva média; 5 – Onda deriva lenta; 6 – Onda amortecimento; 7 – Amortecimento de linha; 8 – Arrasto de linha.**
WIND_COEFFICIENTS
{
    XCV *double* **Distância do ponto de medição até a meia nau (m).**
    FRONTAL_PRESSURE_CENTER *double* **Ponto de aplicação da força frontal de vento (m).**
    LATERAL_PRESSURE_CENTER *double* **Ponto de aplicação da força lateral de vento (m).**
    MATRIX *MatrixNx4* **Coeficientes de arrasto para vento (CDs) (° | adm | adm | adm).**
} *end_WIND_COEFFICIENTS*
CURRENT_COEFFICIENTS
{
    MODEL < “ARANHA” | “OBOKATA” | “REGULAR” | “SDIN” | “TAKASHINA” > **Chave que controla modelo de força de corrente.**
    XCD *double* **Distância do ponto de medição até a meia nau (m). Utilizado pelo método de Obokata.**
    WET_SURFACE *double* **Superfície molhada (m²). Utilizado pelos métodos de Asa Curta (ARANHA) e SDIN.**
    BLOCK_COEFFICIENT *double* **Coeficiente de arrasto (adm). Utilizado pelo método de Asa Curta (ARANHA).**
    CDX *double* **Coeficiente de arrasto (adm). Utilizado pelo método de TAKASHINA.**
    CY *double* **Coeficiente de arrasto transversal (adm). Utilizado pelo método de Asa Curta (ARANHA).**
    LONGITUDINAL_PRESSURE_CENTER *double* **Centro de pressão longitudinal (m). Utilizado pelo método de Asa Curta (ARANHA).**
    FRONTAL_PRESSURE_CENTER *double* **Centro de pressão frontal (m).**
    LATERAL_PRESSURE_CENTER *double* **Centro de pressão lateral (m).**

FRONTAL_PROJECTED_AREA *double* **Área frontal projetada (m^2).**
LATERAL_PROJECTED_AREA *double* **Área lateral projetada (m^2).**
MATRIX *MatrixNx4* **Coeficientes de arrasto para corrente (CDs) (° | adm | adm | adm).**
XMAX *double* **Valor do eixo X no qual ocorre a mudança dos coeficientes: de MATRIX para MATRIX2 (m). Utilizado pelo método de Obokata.**
MATRIX2 *MatrixNx4* **Coeficientes de arrasto para corrente (CDs) (° | adm | adm | adm). Utilizado pelo método de Obokata, deve ser fornecido caso XMAX esteja definido.**
SDIN
{

XDU *double* **Coeficiente para SDIN.**
PRH1 *double* **Coeficiente para SDIN.**
PRH2 *double* **Coeficiente para SDIN.**
CWN1 *double* **Coeficiente para SDIN.**
CWN2 *double* **Coeficiente para SDIN.**
ALF *double* **Coeficiente para SDIN.**
XVR *double* **Coeficiente para SDIN.**
XRR *double* **Coeficiente para SDIN.**
YV *double* **Coeficiente para SDIN.**
YVV0 *double* **Coeficiente para SDIN.**
YVV1 *double* **Coeficiente para SDIN.**
YVVV *double* **Coeficiente para SDIN.**
YUV *double* **Coeficiente para SDIN.**
YDV *double* **Coeficiente para SDIN.**
YR *double* **Coeficiente para SDIN.**
YRR0 *double* **Coeficiente para SDIN.**
YRR1 *double* **Coeficiente para SDIN.**
YRRR *double* **Coeficiente para SDIN.**
YDR *double* **Coeficiente para SDIN.**
YRV *double* **Coeficiente para SDIN.**
YVR *double* **Coeficiente para SDIN.**
YRRV *double* **Coeficiente para SDIN.**
YVVR *double* **Coeficiente para SDIN.**
YUR *double* **Coeficiente para SDIN.**
NV *double* **Coeficiente para SDIN.**

NVV0 *double* **Coeficiente para SDIN.**
NVV1 *double* **Coeficiente para SDIN.**
NVVV *double* **Coeficiente para SDIN.**
NDV *double* **Coeficiente para SDIN.**
NR *double* **Coeficiente para SDIN.**
NRR0 *double* **Coeficiente para SDIN.**
NRR1 *double* **Coeficiente para SDIN.**
NRRR *double* **Coeficiente para SDIN.**
NVVR *double* **Coeficiente para SDIN.**
NRRV *double* **Coeficiente para SDIN.**
NUV *double* **Coeficiente para SDIN.**
NUVR *double* **Coeficiente para SDIN.**
NUVRR *double* **Coeficiente para SDIN.**
NUR *double* **Coeficiente para SDIN.**
NVR *double* **Coeficiente para SDIN.**
NRV *double* **Coeficiente para SDIN.**
VISC *double* **Coeficiente para SDIN.**
VREF *double* **Coeficiente para SDIN.**
XLREF *double* **Coeficiente para SDIN.**

} *end_SDIN*

} *end_CURRENT_COEFFICIENTS*

FREQUENCIES *MatrixNFreqx1* **Frequências de discretização do WAMIT (rad/s).**

ANGLES *MatrixNAngx1* **Ângulos de discretização do WAMIT (°).**

COUPLINGS

{

COUPLING

{

BODY *int* **Referência ao ID global do corpo ao qual as informações de massa adicional e amortecimento potencial estão acopladas.**

ADDED_MASS_FREQ

{

INDEXES *MatrixNIndx1* **Índices da matriz de massa adicional para os quais serão fornecidos os dados.**

MATRIX *MatrixNFreqxNInd* **Coeficientes da matriz de massa adicional (t).**

} *end_ ADDED_MASS_FREQ*

POTENTIAL_DAMPING_COEFFICIENTS_FREQ

{

INDEXES *MatrixNIndx1* **Índices da matriz de amortecimento potencial para os quais serão fornecidos os dados.**

MATRIX *MatrixNFreqxNInd* **Coeficientes da matriz de amortecimento potencial Diagonal principal (t/s | t/s | t/s | t.m²/s | t.m²/s | t.m²/s).**

} *end_ POTENTIAL_DAMPING_COEFFICIENTS_FREQ*

} *end_ COUPLING*

} *end_ COUPLINGS*

NUMBER_FREQUENCIES_SUBDIVISIONS *int* **Número de frequências para interpolação entre cada frequência do WAMIT.**

RAO_SURGE_AMPLITUDE *MatrixNFreqxNAng* **Módulo da resposta ao impulso do casco para *surge*. (m/m).**

RAO_SURGE_PHASE *MatrixNFreqxNAng* **Fase da resposta ao impulso do casco para *surge*. (°).**

RAO_SWAY_AMPLITUDE *MatrixNFreqxNAng* **Módulo da resposta ao impulso do casco para *sway*. (m/m).**

RAO_SWAY_PHASE *MatrixNFreqxNAng* **Fase da resposta ao impulso do casco para *sway*. (°).**

RAO_HEAVE_AMPLITUDE *MatrixNFreqxNAng* **Módulo da resposta ao impulso do casco para *heave*. (m/m).**

RAO_HEAVE_PHASE *MatrixNFreqxNAng* **Fase da resposta ao impulso do casco para *heave*. (°).**

RAO_ROLL_AMPLITUDE *MatrixNFreqxNAng* **Módulo da resposta ao impulso do casco para *roll*. (°/m).**

RAO_ROLL_PHASE *MatrixNFreqxNAng* **Fase da resposta ao impulso do casco para *roll*. (°).**

RAO_PITCH_AMPLITUDE *MatrixNFreqxNAng* **Módulo da resposta ao impulso do casco para *pitch*. (°/m).**

RAO_PITCH_PHASE *MatrixNFreqxNAng* **Fase da resposta ao impulso do casco para *pitch*. (°).**

RAO_YAW_AMPLITUDE *MatrixNFreqxNAng* **Módulo da resposta ao impulso do casco para *yaw*. (°/m).**

RAO_YAW_PHASE *MatrixNFreqxNAng* **Fase da resposta ao impulso do casco para *yaw*. (°).**

EXCITING_WAVE_FORCE_SURGE_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência de força de primeira ordem para *surge*. (kN/m).**

EXCITING_WAVE_FORCE_SURGE_PHASE *MatrixNFreqxNAng* **Fase da função de transferência de força de primeira ordem para *surge*. (°).**

EXCITING_WAVE_FORCE_SWAY_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência de força de primeira ordem para *sway*. (kN/m).**

EXCITING_WAVE_FORCE_SWAY_PHASE *MatrixNFreqxNAng* **Fase da função de transferência de força de primeira ordem para *sway*. (°).**

EXCITING_WAVE_FORCE_HEAVE_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência de força de primeira ordem para *heave*. (kN/m).**

EXCITING_WAVE_FORCE_HEAVE_PHASE *MatrixNFreqxNAng* **Fase da função de transferência de força de primeira ordem para *heave*. (°).**

EXCITING_WAVE_FORCE_ROLL_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência de força de primeira ordem para *roll*. (kN.m/m).**
EXCITING_WAVE_FORCE_ROLL_PHASE *MatrixNFreqxNAng* **Fase da função de transferência de força de primeira ordem para *roll*. (°).**
EXCITING_WAVE_FORCE_PITCH_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência de força de primeira ordem para *pitch*. (kN.m/m).**
EXCITING_WAVE_FORCE_PITCH_PHASE *MatrixNFreqxNAng* **Fase da função de transferência de força de primeira ordem para *pitch*. (°).**
EXCITING_WAVE_FORCE_YAW_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência de força de primeira ordem para *yaw*. (kN.m/m).**
EXCITING_WAVE_FORCE_YAW_PHASE *MatrixNFreqxNAng* **Fase da função de transferência de força de primeira ordem para *yaw*. (°).**
SLOW_DRIFT_FORCE_SURGE_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência quadrática de força de segunda ordem para *surge*. (kN/m).**
SLOW_DRIFT_FORCE_SURGE_PHASE *MatrixNFreqxNAng* **Fase da função de transferência quadrática de força de segunda ordem para *surge* (°). Como a função é quadrática a fase se restringe a 0° ou 180°.**
SLOW_DRIFT_FORCE_SWAY_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência quadrática de força de segunda ordem para *sway*. (kN/m).**
SLOW_DRIFT_FORCE_SWAY_PHASE *MatrixNFreqxNAng* **Fase da função de transferência quadrática de força de segunda ordem para *sway* (°). Como a função é quadrática a fase se restringe a 0° ou 180°.**
SLOW_DRIFT_FORCE_YAW_AMPLITUDE *MatrixNFreqxNAng* **Módulo da função de transferência quadrática de força de segunda ordem para *surge*. (kN/m).**
SLOW_DRIFT_FORCE_YAW_PHASE *MatrixNFreqxNAng* **Fase da função de transferência quadrática de força de segunda ordem para *surge* (°). Como a função é quadrática a fase se restringe a 0° ou 180°.**
MI_INTERPOLATION *int*
SLOW_DRIFT_FORCE_HEAVE_FREQUENCY *Matrix*
SLOW_DRIFT_FORCE_HEAVE_MI *Matrix*
SLOW_DRIFT_FORCE_HEAVE_REAL *Matrix*
SLOW_DRIFT_FORCE_HEAVE_IMAGINARY *Matrix*
SLOW_DRIFT_FORCE_ROLL_FREQUENCY *Matrix*
SLOW_DRIFT_FORCE_ROLL_MI *Matrix*
SLOW_DRIFT_FORCE_ROLL_REAL *Matrix*
SLOW_DRIFT_FORCE_ROLL_IMAGINARY *Matrix*
SLOW_DRIFT_FORCE_PITCH_FREQUENCY *Matrix*

SLOW_DRIFT_FORCE_PITCH_MI *Matrix*
SLOW_DRIFT_FORCE_PITCH_REAL *Matrix*
SLOW_DRIFT_FORCE_PITCH_IMAGINARY *Matrix*

} *end_VESSEL*

LINE (1..*)

{

NAME *string* **Nome de identificação da linha.**

LINE_ID *int* **ID local da linha. Único entre as linhas.**

GLOBAL_ID *int* **ID global da linha. Único no documento.**

ACTIVE < "YES" | "NO" > **Chave que indica se a força desta linha será considerada na integração do corpo. Estado padrão é "YES".**

~~POSITION1~~ *Vector3* **Posição local do *fairlead* conectado a extremidade 1 da linha (m | m | m). Caso esteja zerado indica conexão a um ponto fixo ou âncora.**

OBJECT1 *int* **Referência ao ID global do objeto que está conectado na extremidade 1 da linha.**

OBJECT2 *int* **Referência ao ID global do objeto que está conectado na extremidade 2 da linha.**

FAIRLEAD1 *int* **Referência ao ID local do *fairlead* que está conectado na extremidade 1 da linha. Caso seja zero indica conexão a um ponto fixo ou âncora.**

FAIRLEAD2 *int* **Referência ao ID local do *fairlead* que está conectado na extremidade 2 da linha. Caso seja zero indica conexão a um ponto fixo ou âncora.**

~~TYPE~~ < "MOORING" | "RISER" > **Tipo da linha: ancoragem ou *riser*.**

FLAG_DYNAMIC_EA < 0 | 1 > **Chave que indica se será utilizado o EA dinâmico nesta linha. Valor padrão 0.**

SEGMENTS

{

SEGMENT (1..*)

{

PAID_LENGTH *double* **Comprimento do segmento considerado para o cálculo (m).**

DIV *int* **Número de elementos deste segmento.**

LI *double* **Comprimento do primeiro elemento do segmento (m).**

LF *double* **Comprimento do último elemento do segmento (m).**

~~Q~~ *double* **Razão LI/LF (adm).**

~~R~~ *double* **Razão (LI-LF)/(DIV-1) (m).**

SEGMENT_GROUP_ID *int* **Referência ao ID global do SEGMENT_GROUP deste segmento.**

SUMMARIZED_EXTERNAL_DIAMETER *double* **Valor do diâmetro externo da linha caso o segmento seja do tipo *Flexible Joint* (m).**

SUMMARIZED_MBL *double* **Menor carga para rompimento da linha (kN).**
DYNAMIC_EA *double* **Valor do EA dinâmico da linha (kN). Caso seja fornecido como zero o cálculo é feito internamente. Caso não seja fornecido o EA dinâmico, este não é considerado neste segmento.**
BOUNDARY_CONDITIONS
{
FLOAT
{
MAXLOAD *double*
LENGTH *double*
LPEND *double*
} *end_ FLOAT*
LANK
{
WEIGHT *double*
} *end_LANK*
} *end_BOUNDARY_CONDITIONS*
} *end_SEGMENT*
} *end_SEGMENTS*
INDIVIDUAL_SLOPE < 0 | 1 > **Chave que ativa o plano de fundo individual por linha. Valor padrão 0.**
PLANE *Vector4* **Definição do plano para fundo da individual (m | m | m | m).**
FLEX_ANCHOR *int* **Índice da âncora no vessel.**
FLEX_VESSEL *int* **Local ID do vessel que possui a âncora.**
FLEX_VEL *double* **Velocidade de pagamento da âncora (m/s).**
FLEX_PANC *double* **Peso da âncora (kg).**
FLEX_HOLDING *double* **Capacidade de tração da âncora (kN).**
DYNAMIC_ANALYSIS_DATA
{
COMPUTATION_METHOD < “CATENARY_EQUATION” | “CHARACTERISTIC_CURVES” | “CUSHION” | “FENDER” | “~~PREADYN~~” | “SPRING” > **Indica o método de cálculo da linha.**
CHARACTERISTIC_CURVES
{
NUM_POINTS *int* **Número de pontos da curva de restauração.**
NUM_DEPTHS *int* **Número de curvas de restauração.**

} *end_CHARACTERISTIC_CURVES*

FENDER

{

NUM_POINTS *int* **Número de pontos da curva de restauração.**

STATFRIC *double* **Coeficiente de atrito estático (???).**

DYNAFRIC *double* **Coeficiente de atrito dinâmico (???).**

STIFFRIC *double* **Relação entre atrito e rigidez (???).**

PLANE_NORMAL *Vector3* **Vetor normal ao plano de colisão (m | m | m).**

PLANE_RADIUS *double* **Raio que define o tamanho do plano de colisão (m).**

DAMP_COEF *double* **Coeficiente de amortecimento (???).**

EXP_COEF *double* **Coeficiente exponencial (???).**

RESTORING_CURVE *double* **Curva de restauração (???).**

} *end_FENDER*

CUSHION

{

C1 *double* **Coeficiente de curva de força 1 (???).**

C2 *double* **Coeficiente de curva de força 2 (???).**

PLANE_RADIUS_A *double* **Raio que define o tamanho do plano de ação A do amortecedor (m).**

PLANE_NORMAL_A *Vector3* **Vetor normal ao plano A (m | m | m).**

PLANE_RADIUS_B *double* **Raio que define o tamanho do plano de ação B do amortecedor (m).**

PLANE_NORMAL_B *Vector3* **Vetor normal ao plano B (m | m | m).**

} *end_FENDER*

} *end_DYNAMIC_ANALYSIS_DATA*

RESTORING_CURVES

{

CURVE (1..*)

{

DEPTH *double* **Profundidade a partir do nível das águas calmas da curva de restauração (m).**

POINTS *MatrixNx3* **Curva de restauração da linha (???).**

} *end_CURVE*

} *end_RESTORING_CURVES*

} *end_LINE*

ANCHOR (1..*)

{
GLOBAL_ID *int* **ID global da âncora. Único no documento.**
POSITION *Vector3* **Posição da âncora no sistema de coordenadas global (m | m | m).**
} *end_ANCHOR*
CONNECTION (1..*)
{
GLOBAL_ID *int* **ID global da conexão. Único no documento.**
P1 *Vector3* **Posição do ponto de conexão no sistema de coordenadas global (m | m | m).**
} *end_CONNECTION*
SEGMENT_GROUPS
{
SEGMENT_GROUP (1..*)
{
SEGMENT_TYPE < 1 | ~~2~~ | 3 | **4** | 5 | 6 | 7 | 8 > **Tipo do segmento: 1 – Morring Line; 2 – Flexible Line; 3 – Stiffener; 4 – Connector; 5 – Rigid Tube; 6 – Rigid Joint; 7 – Stress Joint; 8 – Flexible Joint.**
SEGMENT_GROUP_ID *int* **ID local do SEGMENT_GROUP. Único entre os SEGMENT_GROUPS.**
FE_TYPE < 1 | 2 | 5 > **Tipo do elemento: 1 – Barra; 2 – Treliça; 5 – Escalar.**
HYDRODYNAMIC_DIAMETER *double* **Diâmetro hidrodinâmico (m).**
NOMINAL_DIAMETER *double* **Diâmetro nominal (m).**
EXTERNAL_DIAMETER *double* **Diâmetro externo (m).**
INTERNAL_DIAMETER *double* **Diâmetro interno (m).**
STRUCTURAL_EXTERNAL_DIAMETER *double* **Diâmetro externo estrutural (m).**
STRUCTURAL_WALL_THICKNESS *double* **Espessura da parede estrutural (m).**
MIDDLE_EXTERNAL_DIAMETER *double* **Diâmetro hidrodinâmico (m).**
SUMMARIZED_EXTERNAL_DIAMETER *double* **Diâmetro externo nominal (m).**
ELASTIC_MODULUS *double* **Módulo elástico (???).**
SPECIFIC_WEIGHT *double* **Peso específico (kN/m).**
DRY_WEIGHT *double* **Peso específico seco (kN/m).**
WET_WEIGHT *double* **Peso específico molhado (kN/m).**
AXIAL_STIFFNESS *double* **Rigidez axial (kN).**
BENDING_STIFFNESS *double* **Rigidez a flexão (???).**
TORSIONAL_STIFFNESS *double* **Rigidez a torsão (???).**
MORISON_DRAG_COEFFICIENT *double* **Coeficiente de arrasto de Morison (???).**

MORISON_INERTIA_COEFFICIENT **Coeficiente de inércia de Morison (???).**

} *end_SEGMENT_GROUP*

} *end_SEGMENT_GROUPS*

~~ONDA_ENSAIO~~

{

WNPT *int* **Número de pontos da série da onda de ensaio.**

NWVS < 1 | 2 > **Número de picos do espectro de onda de ensaio.**

NHM *int* **Número de harmônicos para cálculo da transformada de Fourier. Não deve ser maior que WNPT.**

WDT *double* **Intervalo de amostragem da série da onda de ensaio (s).**

EXPERIMENT_PERIOD *Vecor2D* **Período de pico teórico utilizado na geração da onda (s).**

EXPERIMENT_HEIGHT *Vecor2D* **Altura significativa teórica utilizada na geração da onda (m).**

EXPERIMENT_GAMA *Vecor2D* **Gama teórico utilizada na geração da onda (adm).**

EXPERIMENT_ALFA *Vecor2D* **Alfa teórico utilizada na geração da onda (adm).**

WDIR *Vecor2D* **Direção de propagação da onda (°).**

WPT *Matrix2x2* **Ponto de medição da onda (m | m).**

WSIG *MatrixNx2* **Série temporal da onda de ensaio (m | m).**

} *end_ONDA_ENSAIO*

} *end_MODEL*