

## Bibliografia

ALMEIDA, L. F. **Otimização de Alternativas para Desenvolvimento de Campo de Petróleo utilizando Computação Evolucionária**, Dissertação de Mestrado, DEE-PUC/RJ, 2003.

ARPS, J. J., et al. **A Statistical Study of Recovery Efficiency**. In American Petroleum Institute Bulletin D14, 33 p, 1967.

ARNONDIN, M. **Integration of Production Analyst and Microsoft Excel's Solver for Production Analyst and Optimization**, SPE 27566, October 1995.

BADRU, O. **Well-Placement optimization using the Quality Map approach**, MSc. Dissertation, Stanford University, 2003.

BECKNER, B. L., SONG, X. **Field Development Planning Using Simulated annealing – Optimal Economic Well Scheduling and Placement**, SPE 30650, January 1995.

BEZERRA, R. A. M., et al. **Modelo Neuro-Fuzzy Hierárquico BSP Mamdani**, In Anais do VI SBAI - Simpósio Brasileiro de Automação Inteligente, Bauru, SP, Sociedade Brasileira de Automática, pp. 745-750, setembro 2003.

BISHOP C. **Neural Network for Pattern Recognition**, Oxford University Press Inc. 1995.

BITTENCOURT, A. C., HORNE, R. N. **Reservoir Development and Design Optimization**, SPE 38895 presented at the 1997 SPE Annual Technical Conference and Exhibition, San Antonio, Texas, October 5-8, 1997.

BITTENCOURT, A. C. **Optimizing Hydrocarbon Field Development Using a Genetic Algorithm Based Approach**, PhD thesis, Stanford University, 1997.

BOBAR, A. R. **Reservoir Engineering Concepts on Well Spacing**, SPE 15338. Janeiro 1985.

BREALEY, R., MYERS, S. **Principles of Corporate Finance**, McGraw-Hill series in finance, New York, 1981

BROOMHEAD, D. S., LOWE, D. **Multivariate functional interpolation and adaptive networks**, Complex Systems, 2:321-355, 1988

BRUCE, G. H. et al. **Calculations of Unsteady-state Gas Flow through Porous Media**, Trans. AIME, 198, 79, 1953

BURKHART, R.H., YOUNG, D. P. **GMRES acceleration and optimization codes**, TA-TR-88, Boeing Computer Services, May 1988.

CANTÚ-PAZ, E. **Designing efficient master-slave parallel genetic algorithms**, Technical report 95004, Illinois Genetic Algorithms Laboratory, University of Illinois and Urbana-Champaign, Urbana, IL, 1997.

CLARK, S. K., et al. **Well Spacing – Its Effects on Recoveries and Profits**, AAPG. Fevereiro de 1944.

CMG. **IMEX Advanced Oil/Gas Reservoir Simulator Version 2000 User's Guide**. Computer Modelling Group LTD., Calgary, Alberta, Canada, 746pp. 2000:a.

\_\_\_\_\_. **Results Visualization and Animation Post-Processing Modules Version 2000 User's Guide**, Computer Modelling Group LTD., Calgary, Alberta, Canada, 59pp. 2000:b.

COHOON, J. P., et al. **Punctuated equilibria: A parallel genetic algorithm**, in J. J. Grefenstette Editor, Proceedings of the Second International Conference of Genetic Algorithms, p. 148, Lawrence Erlbaum Associates, 1987.

CONTRERAS, R., et al. **Técnicas de Seleção de Características Aplicadas a Modelos Neuro-Fuzzy Hierárquicos BSP**, Anais do VI SBAI - Simpósio Brasileiro de Automação Inteligente, Bauru, SP, Sociedade Brasileira de Automática, pp. 403-408, setembro de 2003.

CRICHLOW, H. B. **Modern Reservoir Engineering - A Simulation Approach**, Prentice-Hall, Inc. Englewood Cliffs, New Jersey 07632, 1977.

CRUZ, P. S., HORNE, R. N., DEUTSCH, C., V. **The Quality Map: A Tool for Reservoir Uncertainty Quantification and Decision Making**, SPE 56578, October 1999

CRUZ, P. S. **Reservoir Management Decision-Making in the Presence of Geological Uncertainty**, PhD Thesis, Department of Petroleum engineering, Stanford University, 2000.

CUN, Y. L. **Une procédure d'apprentissage pour réseau a seuil assy métrique**. Proceedings of Cognitiva, 85, 199-604. 1985

CYBENCO, G. **Approximation by superpositions of a sigmoidal function**. Mathematics of Control, Signals and Systems, 2(4), 303-314. 1989.

DAMSLETH et al. **Maximum Information at Minimum Costs**, Journal of Petroleum Technology, pp 1349-1356, 1992

DAVIS, e F., SHEPLER, J. C. **Reservoir Pressure Data Used to Justify Infill Drilling in a Low Permeability Reservoir**, SPE 2260, march 1969.

DAVIS, L. **Handbook of genetic algorithms**, Van Nostrand Reinhold. New York, 44-47, 1991.

\_\_\_\_\_. **Adapting operator probabilities in genetic algorithms**, In J. David Schafer (ed.), Proceedings of the Third International Conference on Genetic Algorithms. San Mateo, Calif.: Morgan Kaufmann Publishers, 1989.

DE SOUZA, F. J. **Modelos Neuro-Fuzzy Hierárquicos**, Tese de Doutorado, DEE/PUC-Rio, 1999.

DE SOUZA, F. J., et al. **Load Forecasting with The Hierarchical Neuro-Fuzzy Binary Space Partitioning Model**, International Journal of Computers Systems and Signals, (ISSN 1608-5655), Vol. 3, No. 2, pp. 118-132, International Association for the Advancement of Methods for System Analysis and Design (IAAMSAD), South Africa, 2002.

DING, Z., STARTZMAN, R. A. **A Software to Oil Field Facility Investments Minimization**, SPE 28252, august 1994

DOUGLAS Jr, J., ARBOGAST, T. **Dual-porosity models for flow in naturally fractured reservoirs "Dynamics of fluids in hierarchical porous formations"**, J. Cushman, ed., Academic Press, London, 1990.

DRISCOLI, V. J. **Recovery Optimization Through Drilling – Concepts, Analysis and Field Results**, SPE 4977. Março de 1974.

EGGERS S. et al. **Simultaneous Multithreading: A Platform for Next-generation Processors**, IEEE Micro, pp. 12-18. September/October 1997.

EKERN, S. **An Option Pricing Approach to Evaluating Petroleum Projects**, Energy Economics 10, 1985, pp. 91-99.

ELMAN, J. L. **Finding Structure in Time**. Cognitive Science, 14, 179-211. 1990.

FANCHI, J. R. **Integrated Flow Modeling**, Elsevier Science, ISBN: 0-444-50500-8, pp 304, 2000.

FIGUEIREDO, K., T., et al. **Reinforcement Learning\_Hierarchical Neuro-Fuzzy Polintree Model for Control of Autonomous Agents**, In Fourth International Conference on Hybrid Intelligent Systems (HIS'04) proceedings, 6 pp, Kitakyushu, Japan, December 2004.

FRASCONI, P., GORI, M. and SODA G., **Local Feedback Multilayered Networks**. Neural computation, 4(1), pp120-130, 1992.

GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization, and Machine Learning**, Addison-Wesley, Reading, 1989.

GONÇALVES, L., et al. **Inverted Hierarchical Neuro-Fuzzy BSP System: A Novel Neuro-Fuzzy Model for Pattern Classification and Rule Extraction in Databases**, aceito para publicação em IEEE Transactions on Systems, Man & Cybernetics, Part C, (ISSN: 1094-6977), 2005.

GRANT, E. L., IRESON, W. G., LEAVENWORTH, R. S. **Principles of Engineering Economy**, 7 Ed. - Wiley, New York, ISBN 047106436X, 687p., 1982

GÜYAGÜLER, B., HORNE, R. **Optimization of Well Placement**, Presented at the Petroleum Production Technology Symposium for ETCE 2000, New Orleans, Louisiana. Also accepted for publication in Journal of Energy Resources Technology.

GÜYAGÜLER, B., et al. **Optimization of Well Placement in a Gulf of Mexico Waterflooding Project**, in 2000 SPE Annual Technical Conference and Exhibition, Dallas, Texas, 1–4 October 2000, SPE 63221.

HALTON J. H. **On the efficiency of certain quasirandom sequences of points in evaluating multidimensional integrals**. Numerische Mathematik, 2:84-90. 1960

HAYKIN, S. **Neural Networks - A Comprehensive Foundation**, Macmillan College Publishing Company, Inc.. pp. 401-402, 1999.

HENNING, M., VINOSKI S. **Advanced CORBA® Programming with C++**, Addison-Wesley professional computing series. USA, 1999.

HOLLAND, J. H. **Adaptation in Natural and Artificial Systems**. MIT Press, Boston, MA, 1992.

HOPFIELD, J. J. **Neural networks and physical systems with emergent collective computational abilities**. Proceedings of the National Academy of Sciences of the U.S.A., 79, pp.2554-2558, 1982.

HORNIK K., et al. **Multilayer feedforward networks are universal approximators**. Neural Networks, 2:359-366, 1989.

HWANG, K. **Advanced Computer Architecture**. Mc Graw-Hill, New York, 1993.

INGBER, L. **Very fast simulated re-annealing**, Math. Comp. Mod., v.12, 967-973. 1989.

JANG J., SUN C. **Neuro-Fuzzy Modeling and Control**, in Proceedings of the IEEE, 83 No 3, pp 378-406. March 1995.

JANG J. S. R. **ANFIS: Adaptive-Network-Based Fuzzy Inference System**, IEEE Transactions on Systems, Man, and Cybernetics, may/june 1993, Vol.23, No.3, pp. 665-685.

JOSUTTIS, N. M. **The C++ Standard Library: A Tutorial and Reference**. Addison Wesley Longman, Inc. 1999

KIRKPATRICK, S. et al, **Optimization by Simulated Annealing**, Science, Vol 220, 671-680, 1983.

KOZA, J. R. **Genetic Programming: On the Programming of Computers by Means os Natural Selection**, MIT Press, USA, 1992.

LAWRENCE, D. **Handbook of Genetic Algorithms**, Van Nostrand Reinhold, USA, 1991.

LEXICON. **The 'Lectric Law Library's Lexicon**. Quite simply the best legal resource that we have come across on the Web. - CNNfn Wonders of the Web. Disponível em , <<http://www.lectlaw.com/def2/p077.htm>> . Acesso em 30 jul. 2003.

LJUNG, L. **System Identification: Theory for the user**. Prentice-Hall, Englewood Cliffs, NJ. 1987.

MANDERICK, B., SPIESSENS, P. **Fine-Grained Parallel Genetic Algorithms**, In J. D. Schaffer editor, Proceedings of the Third International Conference on Genetic Algorithms, p. 428, Morgan Kauffman, 1989.

MELANIE, M. **An Introduction to Genetic Algorithms**, The MIT Press, USA, 1994.

MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**, Springer-Verlag, USA, 1996.

MILANI, N. V. R. **Determinação do Espaçamento Ótimo entre poços considerando a heterogeneidade do reservatório**, Dissertação de Mestrado, Universidade Federal de Ouro Preto, Departamento de Mineração da Escola de Minas, 1988.

MINSKY, M., PAPERT, S. **Perceptrons: An Introduction to Computational Geometry**. The MIT Press. 1969

NAKAJIMA, L., SCHIOZER, D. J. **Automated Methodology for Field Performance Optimization Developed with Horizontal Wells**. Latin American and Caribbean Petroleum Engineering Conference, Port-of-Spain, Trinidad, West Indies. SPE 81031, 2003.

NIEDERREITER H. **Random Number Generation and Quasi-Monte Carlo Methods**, Society for Industrial & Applied Mathematics, 1992

NYSTAD, A. N. **Reservoir Economic Optimization**, SPE 13775, 1985.

OMG. **CORBA Services: Common Object Services Specification**, disponível em <ftp://ftp.omg.org/pub/docs/formal/98-07-05.pdf>. Framingham, MA Object Management Group, ultimo acesso em 23 julho 2003

ORR, J. L. **Introduction to Radial Basis Function Networks**, Centre for Cognitive Science, University of Edinburgh, 2, Buccleuch Place, Edinburgh EH89LW, Scotland, April, 1996.

\_\_\_\_\_. **Recent Advances in Radial Basis Function Networks**, Institute for Adaptive and Neural Computation, Division of Informatics, Edinburgh University, Edinburgh EH89LW, Scotland, June, 1999.

PARI, C. **Redes Neurais Temporais para o Tratamento de Sistemas Variantes no Tempo**, Dissertação de Mestrado, DEE-PUC/RJ, 1999.

PARKER, D. B. **Learning Logic** (Tech. Report, Nos TR-47). Cambridge, MA: Massachusetts Institute of Technology, Center for Computational Research in Economics and Management Science, 1985.

PEDROSO, C. J. **Otimização de Locações de Poços Usando Simulação Numérica de Reservatórios**. Dissertação de Mestrado, Faculdade de Engenharia Mecânica, Unicamp, 1999. 123 p.

PEDROSO, C. J., SCHIOZER, D. J. **Otimização de Locações de Poços usando Simulação Numérica de Reservatórios e Computação Paralela (PVM)**, Rio Oil&Gas Expo and Conference, Rio de Janeiro, October, 2000.

PRESS, W. H. et al. **Numerical Recipes: The Art of Scientific Computing**. Cambridge: Cambridge University Press, 1986

REYNOLDS R. G. **An Introduction to Cultural Algorithms**, 2002 World Congress on Computational Intelligence.

ROGERS, L.L., DOWLA, F. U. **Optimal Groundwater Remediation Using Artificial Neural Networks with Parallel Solute Transport**, Water Resources Research, 30(2): 458-481. 1994.

RUMELHART, D. E., et al. **Learning representations by back-propagation errors**. Nature 323, 533-536, 1986.

SAAD Y., SCHULTZ M. H. **GMRES: Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems**, SIAM Journal of Scientific and Statistical Computing, 7, pp. 856-869, 1986.

SALAZAR, V. M., et al. **External Parallelization of Reservoir Simulators Using a Network of Workstations and PWM**. Fourth Latin American and Caribbean Petroleum Engineering Conference, Trinidad and Tobago, 23-26. Society of Petroleum Engineers SPE 36109. 1996.

SALEM ZEBULUM, R. **Síntese de Circuitos Eletrônicos por Computação Evolutiva**, Tese Doutorado, DEE-PUC/RJ, 1999

SCHILTHUIS, R. J. **Active Oil and Reservoir Energy**, Trans. AIME, 118, 33, 1936

SCHIOZER, D. J., MEZZOMO, C. C. **Methodology for Field Development Optimization with Water Injection**, SPE Hydrocarbon Economics and Evaluation Symposium, Dallas, Texas, U.S.A., SPE 82021, 2003.

SEBER G. A. F. **Linear Regression Analysis**, Wiley Series in probability and mathematics, New York: J Wiley, 1977.

SOBOL, I. M. **On the distribution of points in a cube and the approximate evaluation of integrals**. U.S.S.R. Computational Mathematics and Mathematical Physics, 7(4):86-112, 1967.

STONE, M. **Cross-validatory choice and assessment of statistical predictions**, I Journal of the Royal Statistical Society, vol. B36, pp. 111-133, 1974.

\_\_\_\_\_. **Cross-validation: A review**, In **Mathematische Operationsforschung Statistischen**, Serie Statistics, vol.9, pp. 127-139. 1978

STROUSTRUP, B. **The C++ Programming Language**, Third Edition, Addison Wesley Longman Inc. 1999.

TOMASSINI, M. **Parallel and Distributed Evolutionary Algorithms: A Review**, In **Evolutionary Algorithms in engineering and Computer Science**, J Wiley and Sons, K. Miettinen, M. Mäkelä, 1999.

\_\_\_\_\_. **The parallel genetic cellular automata: Application to global function optimization**. In R. F. Albrecht, C. R. Reeves, and N. C. Steele, editors, **Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms**, pages 385--391. Springer-Verlag, 1993.

TÚPAC, Y. J. et al. **Selection of Alternatives for Oil Field Development by Genetic Algorithms**, Revista de Engenharia Térmica RETERM, Ed. Especial n. 2 ISSN 1676-1790 - Curitiba – PR, 2002.

WERBOS, P. **Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences**. Unpublished doctoral dissertation, Harvard University, 1974.

WASSERMAN, W., Neter J. **Applied linear statistical models: regression, analysis of variance, and experimental designs**. Homewood, Iii, R. D. Irwin, 1976.

WILLIAMS, R. J., ZIPSER, D. **A learning algorithm for continually running fully recurrent neural networks**, Neural Computation, 1(2), pp.270-280, 1989.

## Apêndice A – Simulador de Reservatórios IMEX

Neste apêndice são mostradas algumas informações referidas ao simulador de reservatórios IMEX utilizado neste trabalho de tese

### O Simulador IMEX

A corporação CMG: *Computing Modelling Group* (CMG 2000) desenvolveu um pacote de aplicações para simulação de reservatório: simulador *black-oil* IMEX, simulador de injeção de gás, editor de malhas para simuladores CMG, gerador de relatórios, visualizador da malha de reservatório em 2D e 3D, e graficador de curvas de resultados. O mais importante destes produtos para este trabalho é o IMEX.

IMEX é um simulador *black-oil* (Fanchi 2000) de três fases que leva em consideração a gravidade e efeitos capilares. O IMEX suporta configurações de *grid* cartesiana, cilíndrica e com profundidade e espessura variáveis. Desta forma é possível mapear as configurações de campos em 2D e 3D.

O IMEX foi programado em FORTRAN 77 e pode ser executado sob diferentes plataformas: mainframes IBM, CDC, CRAY, Honeywell, DEC, Prime, Gould, Unisys, Apollo, SUN, HP, FPS, IBM-PC 486, e família Intel Pentium.

### Características do IMEX

O simulador IMEX pode ser executado em três modos: explícito, implícito ou adaptativo. Estes modos, referem-se à forma como são resolvidas as equações não-lineares do modelo de fluxo. No caso de modo explícito, o passo de tempo utilizado deve ser sempre menor que o máximo valor de passo que mantenha a estabilidade do sistema. No modo implícito, é necessário resolver um sistema de equações não lineares a cada passo, podendo ser escolhidos passos maiores. O modo explícito é vantajoso quando as variações dos estados são lentas e não são feitos muitos cálculos, enquanto que o modo implícito é necessário quando existem muitas variações nos estados e fica necessário resolver sistemas de equações. O modo adaptativo permite escolher entre a

solução implícita ou explícita segundo a característica das variações dos estados do sistema.

Geralmente, na malha, só um pequeno número de blocos deve ser tratado de forma totalmente implícita, pois a maioria dos blocos pode ser tratado de forma explícita. Este procedimento é bastante proveitoso para problemas onde só existem grandes fluxos nas proximidades dos poços (efeito *cone*) ou em reservatórios estratificados com camadas bem finas. Utilizando o modo adaptativo o tratamento de forma totalmente implícita é feito apenas onde for necessário, obtendo-se uma grande economia no esforço computacional requerido.

O IMEX suporta também as opções de dupla porosidade e dupla permeabilidade (Douglas, 1990), as quais permitem o tratamento de reservatórios com fraturas, onde as fraturas são modeladas como duas partes com porosidade primária e secundária. Já pela dupla permeabilidade é possível modelar a comunicação de fluidos entre as duas partes definidas na fratura.

O simulador IMEX usa uma rotina de solução baseada em eliminação de *Gauss* incompleta, isto serve como um acondicionamento dos dados antes de aplicar o método GMRES (*Generalized Minimum Residual*) (Saad e Shultz, 1986; Burkhart, 1988) que é um método iterativo que permite minimizar a norma do resíduo baseado na geração dos espaços de *Krilov*. Em conjunto, o acondicionamento dos dados e o método GMRES permitem diminuir o custo computacional e os requerimentos de memória quando o número de equações a resolver é muito grande.

## **Uso do IMEX**

Neste simulador a comunicação com o usuário é efetuada através de arquivos de entrada e saída. O usuário deve criar um arquivo texto de entrada (formato .DAT) que contém: a configuração da malha, informação geológica, tipo de fluxo de óleo, condições iniciais, métodos numéricos, lista de poços, entre outras coisas. O simulador, após a execução, fornece os valores de saída em um grupo de arquivos. Esse fluxo de informações é apresentado no seguinte gráfico:

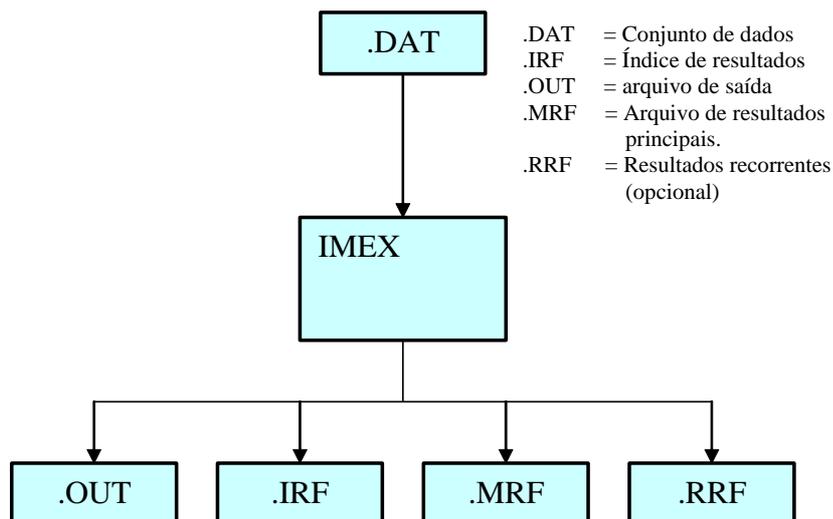


Figura 74. Arquivos de entrada e saída do IMEX

O Arquivo .DAT é imprescindível, uma vez que este arquivo contém todas as informações que definem a configuração completa da reserva a ser simulada. A informação contida no arquivo .DAT está classificada em 7 tipos de dados:

<b>Tipos de dados usados no arquivo .DAT</b>
1. Dados de controle de Entrada e Saída
2. Dados de descrição do reservatório
3. Propriedades das componentes
4. Dados de Rochas e Fluidos
5. Condições Iniciais
6. Informação sobre os métodos numéricos
7. Informação dos poços e dados recorrentes.

Tabela 37. Tipos de dados existentes no arquivo .DAT

Se a simulação for com recorrência, deverá existir o arquivo .RRF e a entrada seguinte deve estar composta pelo conjunto de arquivos .DAT, .IRF, .MRF e .RRF da última execução.

Os arquivos .OUT e .IRF contêm os resultados globais da simulação, isto é, valores de óleo, gás, água etc. O arquivo .IRF contém o índice dos resultados e o arquivo .OUT armazena os resultados numéricos em si.

## Comando INCLUDE

Neste trabalho, os únicos dados a serem alterados do ponto de vista do simulador de reservatório são os relacionados aos poços (informação dos poços e dados descritos na Tabela 37).

O comando `*INCLUDE` permite ao simulador quebrar a leitura seqüencial do arquivo de entrada `.DAT` primário para começar a leitura de um arquivo secundário. Uma vez finalizada a leitura do segundo arquivo (*EOF is reached*), continua-se com a leitura a partir da linha seguinte a que continha o comando `*INCLUDE`.

O formato do comando é o seguinte: `*INCLUDE 'arquivosec'`

Este comando facilita a alteração de algumas partes do arquivo `.DAT` sem afetar o resto. Como mencionado acima, no trabalho proposto precisam ser alterados apenas os dados relacionados aos poços; portanto o arquivo `.DAT` deve conter um comando `*INCLUDE xxxxxx.inc` no momento que se começa a definição dos poços no campo. Deste modo, os dados de definição de poços (tipo de poços, locações, geometria, parâmetros físicos, restrições, cronograma) estarão definidos no arquivo secundário `xxxxxx.inc`. Com esta estratégia, as mudanças dadas pelo processo de otimização serão fornecidas em um arquivo pequeno, ao invés de se precisar re-criar arquivos `.DAT` completos.

## O Aplicativo *Results Report*

Todos os resultados da simulação são armazenados em arquivos `.OUT` e `.IRF`, sendo que a maioria destes resultados não é necessária para se obter a curva de produção do campo. O aplicativo *results report* (CMG 2000:2) permite escolher apenas os dados necessários de acordo com a aplicação, servindo como um gerador de relatórios.

No problema em questão precisa-se dos seguintes valores:

- Óleo acumulado em superfície no grupo de poços produtores do campo;
- Gás acumulado em superfície no grupo de poços produtores do campo;
- Afluência de água em superfície no grupo de poços produtores do campo;
- Taxa de Produção de óleo, gás e água em condições de superfície do campo.

Portanto, ao se usar o *results report* com um *script* adequado às necessidades do trabalho, obtém-se os valores acima de forma resumida e legível pelo módulo que calcula o Valor Presente Líquido.

## Apêndice B – Arquivos dos reservatórios utilizados

Neste apêndice mostram-se os arquivos .DAT dos modelos de reservatórios IMEX que foram utilizados nos experimentos desta tese:

### Reservatório homogêneo 30x30x1

```
*****
**   Arquivo gerado pelo programa AVALHOR
**   Ricardo C. M. Portella - CENPES/DIGER/SIGER
**   portella@cenpes.petrobras.com.br
*****
*TITLE1 'Campo 30 x 30 testes ANEPI'
*INUNIT *MODSI
*OUTPRN *WELL *BRIEF
*OUTPRN *GRID *NONE
*OUTPRN *TABLES *NONE
*OUTPRN *RES *NONE
*WSRF *WELL 1
*WSRF *GRID 1
*OUTSRF *WELL *ALL
*OUTSRF *GRID *PRES *SW
*GRID *CART 30 30 1
*KDIR *DOWN
*DI *CON 100.000000
*DJ *CON 100.000000
*DK *CON 90.000000
*DEPTH 1 1 1 1000.000000
*POR *CON 0.200000
*PRPOR 1.000000
*CPOR 0.0
*PERMI *CON 1000.000000
*PERMJ *CON 1000.000000
*PERMK *CON 1000.000000
*MODEL *BLACKOIL
*PVT *BG 1
1.033512 0.691811 1.060850 0.351997 8.992717 0.005464 0.000000
4.145154 1.692682 1.063113 0.087764 8.521406 0.009372 0.000000
7.256797 2.812482 1.065657 0.050131 8.047870 0.013086 0.000000
10.368439 4.015568 1.068402 0.034467 7.593067 0.017240 0.000000
13.480081 5.284059 1.071312 0.026370 7.164903 0.021652 0.000000
16.591722 6.606956 1.074363 0.021311 6.766004 0.026501 0.000000
19.703363 7.976727 1.077538 0.017850 6.396483 0.031845 0.000000
22.815006 9.387853 1.080827 0.015334 6.055212 0.037734 0.000000
25.926649 10.836096 1.084220 0.013422 5.740470 0.044220 0.000000
29.038292 12.318088 1.087711 0.011921 5.450296 0.051357 0.000000
32.149937 13.831082 1.091293 0.010710 5.182682 0.059201 0.000000
35.261578 15.372792 1.094963 0.009714 4.935676 0.067809 0.000000
38.373219 16.941277 1.098715 0.008880 4.707439 0.077242 0.000000
41.484859 18.534872 1.102548 0.008172 4.496269 0.087568 0.000000
44.596500 20.152128 1.106457 0.007563 4.300607 0.098854 0.000000
47.708141 21.791771 1.110441 0.007033 4.119039 0.111174 0.000000
136.559021 75.179390 1.249259 0.002216 1.769343 1.352840 0.000000
225.409882 136.663757 1.425471 0.001374 1.104641 5.967701 0.000000
314.260773 203.405060 1.630924 0.001087 0.804184 14.028974 0.000000
403.111633 274.151703 1.861210 0.000950 0.634863 23.863567 0.000000
*DENSITY *OIL 915.857605
```

```

*CO 0.000379
*CVO 0.007112
*DENSITY *WATER 1071.860840
*GRAVITY *GAS 0.800000
*BWI 1.033549
*CW 0.000039
*REFPW 44.599998
*VWI 0.433018
*CVW 0.000254
*ROCKFLUID
*RPT 1
*SWT
 0.200000 0.000000 0.800000 0.000000
 0.225000 0.000750 0.722000 0.000000
 0.250000 0.003000 0.648000 0.000000
 0.275000 0.006750 0.578000 0.000000
 0.300000 0.012000 0.512000 0.000000
 0.325000 0.018750 0.450000 0.000000
 0.350000 0.027000 0.392000 0.000000
 0.375000 0.036750 0.338000 0.000000
 0.400000 0.048000 0.288000 0.000000
 0.425000 0.060750 0.242000 0.000000
 0.450000 0.075000 0.200000 0.000000
 0.475000 0.090750 0.162000 0.000000
 0.500000 0.108000 0.128000 0.000000
 0.525000 0.126750 0.098000 0.000000
 0.550000 0.147000 0.072000 0.000000
 0.575000 0.168750 0.050000 0.000000
 0.600000 0.192000 0.032000 0.000000
 0.625000 0.216750 0.018000 0.000000
 0.650000 0.243000 0.008000 0.000000
 0.700000 0.300000 0.000000 0.000000
*SLT
 0.300000 0.900000 0.000000 0.000000
 0.370000 0.238864 0.008000 0.000000
 0.405000 0.210906 0.018000 0.000000
 0.440000 0.184687 0.032000 0.000000
 0.475000 0.160207 0.050000 0.000000
 0.510000 0.137468 0.072000 0.000000
 0.545000 0.116468 0.098000 0.000000
 0.580000 0.097207 0.128000 0.000000
 0.615000 0.079687 0.162000 0.000000
 0.650000 0.063905 0.200000 0.000000
 0.685000 0.049864 0.242000 0.000000
 0.720000 0.037562 0.288000 0.000000
 0.755000 0.027000 0.338000 0.000000
 0.790000 0.018178 0.392000 0.000000
 0.825000 0.011095 0.450000 0.000000
 0.860000 0.005751 0.512000 0.000000
 0.895000 0.002148 0.578000 0.000000
 0.930000 0.000284 0.648000 0.000000
 0.965000 0.000000 0.722000 0.000000
 1.000000 0.000000 0.800000 0.000000
*INITIAL
*VERTICAL *BLOCK_CENTER *WATER_OIL
*PB *CON 44.599998
*REFDEPTH 990.000000
*REFPRES 100.000000
*DWOC 1100.000000
*NUMERICAL
** Para resolver o problema totalmente EXPLICITO (IMPES)
** troque a linha de baixo para *AIM *OFF. Para a resol-
** ver o problema totalmente IMPLICITO troque a linha
** abaixo para *AIM *THRESH 0.0 0.0
*AIM *STAB
*NCUTS 10
*DTMIN 5.0
**DTMAX 365.

```

```
*RUN
*DATE 2001 1 1
*DTWELL 5.0
*GROUP 'ANEPIGroup' *ATTACHTO 'Field'
*INCLUDE include.inc
*TIME 7201.0
```

## Reservatório Heterogêneo 57x33x3

```
RESULTS SECTION INOUT
*INTERRUPT *RESTART-STOP
*RANGECHECK *ON
*XDR *ON
*WPRN *WELL 100
*WPRN *SECTOR 100
*WPRN *GRID 100
*WPRN *ITER *NONE
*OUTPRN *WELL *BRIEF
*OUTPRN *TABLES *NONE
*OUTPRN *GRID *NONE
*OUTPRN *RES PV HCPV

*WSRF *WELL 1
*WSRF *SECTOR 1
*WSRF *GRID 1
*OUTSRF *GRID *PRES *Sw *SG *DATUMPRES *OILPOT *BPP

*OUTSRF *RES *ALL

*OUTSRF *WELL *LAYER *NONE
*OUTDIARY *BRIEF *PRESAQ *HEADER 20
*INUNIT *MODSI
*OUTUNIT *MODSI

**
** Export from SimGrid Version 2.5
** Grid regGrid made by OLINTO GOMES DE SOUZA JUNIOR
** 1710413 on Wed Jan 16 10:18:52 2002

**
** UNITS meters

*GRID *CORNER 33 57 3

*COORD
** VALORES OMITIDOS PARA DIMINUIR NÚMERO DE PÁGINAS

*ZCORN
** VALORES OMITIDOS PARA DIMINUIR NÚMERO DE PÁGINAS

RESULTS SECTION NETPAY

RESULTS SECTION NETGROSS
*NETGROSS *CON 1.00

*POR *CON 0.229

*PERMI *CON 575.000000
*PERMJ *CON 575.000000
*PERMK *CON 57.500000

RESULTS SECTION TRANS

RESULTS SECTION FRACS
```

RESULTS SECTION VOLMOD

RESULTS SECTION GRIDNONARRAYS

\*CPOR 0.000010

\*PRPOR 390.000000

RESULTS SECTION GRIDOTHER

RESULTS SECTION ROCKCOMPACTION

RESULTS SECTION SECTORLEASE

MODEL \*BLACKOIL

\*\*\$ OilGas Table 'Table A'

\*PVT \*BG 1

**	P	Rs	Bo	BG	VisO	VisG
0.1		0.0	1.0466	1.000	39.56	0.0123
19.614	8.3	1.0707	0.036	14.36	0.0125	
39.228	13.2	1.0807	0.026	11.13	0.0129	
54.370	16.4	1.0858	0.016	9.51	0.0139	
88.263	24.	1.10000		0.010	7.2	0.0150
588.42	136.	1.33000		0.002	1.2	0.0432

\*DENSITY\*OIL 940.8

\*GRAVITY\*GAS 0.7

\*CO 7.75E-05

\*CVO 0.045

\*BWI 1.013

\*CW 4.08E-05

\*REFPW 277.2

\*VWI 0.53

\*CVW 0

\*DENSITY\*WATER 1055.

RESULTS SECTION MODELARRAYS

\*ROCKFLUID

\*RPT 1

\*SWT

0.250000	0.000000	0.800000	0.000000
0.275000	0.003053	0.685900	0.000000
0.300000	0.007518	0.583200	0.000000
0.325000	0.012735	0.491300	0.000000
0.350000	0.018511	0.409600	0.000000
0.375000	0.024741	0.337500	0.000000
0.400000	0.031358	0.274400	0.000000
0.425000	0.038316	0.219700	0.000000
0.450000	0.045579	0.172800	0.000000
0.475000	0.053121	0.133100	0.000000
0.500000	0.060919	0.100000	0.000000
0.525000	0.068955	0.072900	0.000000
0.550000	0.077213	0.051200	0.000000
0.575000	0.085680	0.034300	0.000000
0.600000	0.094345	0.021600	0.000000
0.625000	0.103198	0.012500	0.000000
0.650000	0.112230	0.006400	0.000000
0.675000	0.121433	0.002700	0.000000
0.700000	0.130800	0.000800	0.000000
0.750000	0.150000	0.000000	0.000000

\*SLT

0.300000	0.900000	0.000000	0.000000
0.370000	0.238864	0.008000	0.000000
0.405000	0.210906	0.018000	0.000000
0.440000	0.184687	0.032000	0.000000
0.475000	0.160207	0.050000	0.000000
0.510000	0.137468	0.072000	0.000000
0.545000	0.116468	0.098000	0.000000

```

0.580000 0.097207 0.128000 0.000000
0.615000 0.079687 0.162000 0.000000
0.650000 0.063905 0.200000 0.000000
0.685000 0.049864 0.242000 0.000000
0.720000 0.037562 0.288000 0.000000
0.755000 0.027000 0.338000 0.000000
0.790000 0.018178 0.392000 0.000000
0.825000 0.011095 0.450000 0.000000
0.860000 0.005751 0.512000 0.000000
0.895000 0.002148 0.578000 0.000000
0.930000 0.000284 0.648000 0.000000
0.965000 0.000000 0.722000 0.000000
1.000000 0.000000 0.800000 0.000000

```

## RESULTS SECTION ROCKARRAYS

```

*INITIAL
*VERTICAL *BLOCK_CENTER *WATER_OIL
*REFDEPTH 3957.000000
*REFPRES 413.800000
*DWOC 3957.000000
*DGOC 0.000000

```

## RESULTS SECTION INITARRAYS

```

**$ RESULTS PROP PB Units: kg/cm2
**$ RESULTS PROP Minimum Value: 54.370 Maximum Value: 54.370
*PB *CON 54.370

```

## \*NUMERICAL

```

** Para resolver o problema totalmente EXPLICITO (IMPES)
** troque a linha de baixo para *AIM *OFF. Para a resol-
** ver o problema totalmente IMPLICITO troque a linha
** abaixo para *AIM *THRESH 0.0 0.0
*AIM *STAB
*NCUTS 10
*DTMIN 0.001000
*DTMAX 365.000000

```

## RESULTS SECTION NUMARRAYS

```

*RUN
*DATE 2002 1 18
*GROUP 'ANEPIGroup' *ATTACHTO 'Field'

```

```

** CORTEI AS RESTRIÇÕES DE CURTE D'ÁGUA MÍNIMO E
** PRESSÃO DE ÓLEO MÍNIMA
** PARA PODER REALIZAR O GA DE CRONOGRAMA E O
** MAPA DE QUALIDADE
** GCONM 'Field'
** *WCUT 0.9 *STOP
** *MINOIL 100. *STOP

```

```

*INCLUDE include.inc
*DATE 2022 1 18

```

## Apendice C – Arquitetura Corba

CORBA (Common Object Request Broker Architecture) (OMG, 1997) é uma arquitetura que permite ter modelos e interfaces para programação de aplicações orientadas a objeto portáteis e distribuídas. Permite ter independência de linguagem, de plataforma e de protocolos de comunicação em rede. Isto faz com que CORBA seja adequado para o desenvolvimento de aplicações novas e a integração destas com os sistemas distribuídos já existentes.

### Glossário CORBA

Segue uma explicação breve dos termos mais importantes do CORBA:

**Objeto CORBA:** é uma entidade virtual capaz de ser alocada em um ORB e receber requisições de clientes para fazer alguma coisa. Pode-se dizer que é virtual no sentido que não existe até ser concretizado por uma implementação feita em uma linguagem de programação.

**Objeto destino:** refere-se ao objeto CORBA que é o destino de uma requisição.

**Cliente CORBA:** é uma entidade que faz uma requisição a um objeto CORBA, este cliente pode estar localizado em uma rede distante do objeto CORBA ou podem formar parte de uma mesma aplicação.

**Servidor CORBA:** é uma aplicação na qual podem existir um ou mais objetos CORBA instanciados. Cabe ressaltar que o conceito de servidor e cliente são válidos apenas para as requisições e não para aplicações, pois em um dado momento, uma aplicação servidor, pode tornar-se cliente.

**Requisição:** é uma invocação feita desde um cliente para um objeto CORBA. Existe um fluxo de informações desta requisição que saem do cliente até o objeto destino no servidor para, depois ter um fluxo de retorno com a resposta dada para esta requisição.

**Referência a objeto:** é um *handle* utilizado para identificar, localizar e endereçar um objeto CORBA específico. O cliente utiliza a referência de objeto para poder encaminhar suas requisições até os objetos desejados, porém, o cliente não pode criar nem alterar referências.

**Servente:** que é uma entidade implementada em linguagem de programação (C++, java) implementando um ou vários objetos CORBA. Pode-se dizer que o

servente é quem “faz existir” os objetos CORBA. Este conceito existe no contexto de aplicações servidoras. No caso particular do C++, um servente é uma instância real de uma classe específica com implementação.

## Fluxo no CORBA

Como mostra a Figura 75, a aplicação cliente realiza uma requisição e a aplicação do servidor a recebe e retorna uma resposta. A requisição do cliente flui através do ORB e chega ao servidor da seguinte forma:

- O cliente pode escolher usar requisições estáticas desde os *stubs* já compilados do IDL ou usando o DII (*Dynamic Invocation Interface*). Em qualquer dos casos, o cliente encaminha a requisição ao núcleo ORB ao qual pertence;
- O ORB do cliente transmite a requisição até o ORB onde é localizada a aplicação servidora, podendo ser o mesmo ORB;
- O ORB do servidor entrega a requisição ao adaptador de objeto criado pelo objeto destino;
- O adaptador de objeto repassa a requisição ao servente que implementa o objeto destino. Como no caso do cliente, o servidor pode escolher entre mecanismo de entrega estáticos e dinâmicos de seus serventes, ora usando o *skeleton* compilado do IDL, ora usando o DSI (*Dynamic Skeleton Interface*).
- O servente recebe a requisição e executa o código definido na implementação
- Depois do servente finalizar a execução, ele retorna a resposta até a aplicação do cliente.

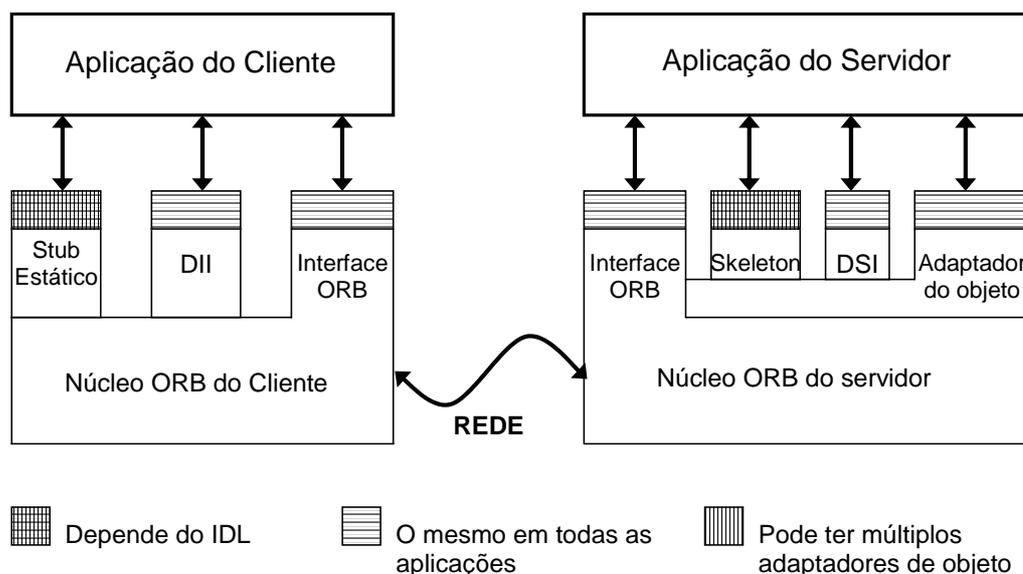


Figura 75. Fluxo de requisições e respostas na arquitetura CORBA

## Linguagem de definição de Interface (IDL)

Para poder realizar requisições, o cliente há de conhecer a interface dada pelo objeto CORBA. Define-se uma interface de objeto como um conjunto de operações e tipos de dados e estruturas de dados que podem ser enviadas e recebidas nestas operações. O cliente deve conhecer também o propósito e a semântica das operações que vai solicitar.

No CORBA, as interfaces de objetos são definidas através de uma metalinguagem de definição de interface da OMG (*OMG Interface Definition Language IDL*). Nesta linguagem podem ser implementadas as interfaces, os tipos de dados, estruturas de dados, de forma tal que fiquem independentes de quaisquer linguagens de programação, deixando para estas a implementação.

IDL já suporta muitos tipos de dados simples como inteiros com o sinal, *strings*, ponto flutuante, booleanos e tipos enumerados. Também permite criar estruturas de tipo *vector*, *union*, *sequences* (vetores unidimensionais) e *exceptions*. Todos estes tipos servem para definir os tipos de parâmetros de entrada e saída de qualquer operação dentro da interface. IDL também suporta a criação de módulos com o intuito de ter um melhor controle de espaços de nomes (namespaces).

## Mapeamento em Linguagem de Programação

Dado que o IDL permite apenas realizar as definições de tipos de dados, estruturas de dados e interfaces, não há como compilar e executar diretamente

desde aqui. O mapeamento em linguagem permite traduzir os tipos de dados, estruturas e declarações de objetos para dados de linguagens de programação conhecidas,. Existe mapeamentos normalizados pela OMG para C, C++, SmallTalk, COBOL, ADA e Java.

No caso do C++, que foi empregado neste trabalho de tese, uma interface é traduzida como uma classe (class), e as operações dadas na interface são traduzidas como métodos da classe.

Segue nas Tabela 38 e Tabela 39 alguns mapeamentos feitos do IDL para C++

IDL	C++
module	namespace ou class
interface	class
const type = value	const CORBA::type = value
enum etype {a, b, c}	enum etype {a, b, c}

Tabela 38. Mapeamentos IDL para C++ para entidades

IDL	C++
short	CORBA::Short
Long	CORBA::Long
long long	CORBA::LongLong
unsigned short	CORBA::UShort
unsigned long long	CORBA::ULongLong
float	CORBA::Float
double	CORBA::Double
Long double	CORBA::LongDouble
char	CORBA::Char
string	char*
boolean	CORBA::Boolean

Tabela 39. Mapeamentos IDL para C++ para tipos conhecidos

## Facilidades para requisição e envio

As aplicações que usam CORBA funcionam pelo envio e recepção de requisições nos objetos CORBA. Existem duas abordagens para as requisições:

### Requisição e envio estáticos

O IDL é traduzido em *stubs* e *skeletons* em uma linguagem de programação específica, que são compilados nas aplicações. Nesta abordagem

as aplicações possuem conhecimento estático dos tipos de dados e operações dos objetos CORBA que foram mapeadas a partir do IDL. O *stub* é o lado do cliente e o *skeleton* é o lado do servidor.

### Requisição e envio dinâmicos:

Neste caso a construção e envio de requisições CORBA é feita em tempo de execução. Como não existe informação disponível em tempo de compilação, a criação e interpretação de requisições em tempo de execução requer acessar serviços que possam fornecer informação sobre as interfaces e os tipos. Isto pode ser feito desde um Repositório de Interfaces que permite acessar a definições IDL em tempo de execução ou mesmo através de intervenção do usuário através de uma interface GUI.

## Interfaces empregadas no trabalho de tese

### Código IDL da interface – cenário

Tipo de dados para envio de arquivos .DAT

```
typedef sequence<string> TFileLines;
```

Estrutura com informações do campo petrolífero

```
struct TOilfield
{
    string            oilfieldName;
    string            oilfieldPath;
    unsigned long    iSize;
    unsigned long    jSize;
    unsigned long    kSize;
    string            gridType;
    string            wellsIncludeFile;
    sequence<string> includeFilesList;
    double            reserveSize;

    unsigned long    maxWellsNumber;
    double            maxHoriLength;
    double            minWellsDistance;
    unsigned long    geometrywell; // define tipo de poços H, V ou HV
    // Parameters to use in .INC generation (wells)
    double            RAD;
    double            SKIN;
    double            stw_vert_inj;
    double            bhp_vert_inj;
    double            stw_hori_inj;
    double            bhp_hori_inj;
    double            stl_vert_pro;
    double            bhp_vert_pro;
    double            stl_hori_pro;
    double            bhp_hori_pro;
};
```

### Estrutura que abrange toda a informação enviada no cenário

```

struct TDataware
{
    TOilfield          oilfield;
    String             hostname;
    String             IPaddress;
    ACB::TOilCommodity oilCommodity;
    ACB::TOperationalCost operationalCost;
    ACB::TDevelopmentCost developmentCost;
};

```

### Estrutura de dados dos parâmetros empregados para o cálculo do VPL

```

struct TNPVParams
{
    string             hostname;
    string             IPaddress;
    unsigned long     FTPPort;
    ACB::TOilCommodity oilCommodity;
    ACB::TOperationalCost operationalCost;
    ACB::TDevelopmentCost developmentCost;
};

```

### Definição da interface de envio do Cenário

```

//=====
//                               interface TDataCorba
//=====
interface TCorbaDataware: ACB::TCorbaBase
{
    long sendDataaware (in TDataware dataware);
    long sendDATFile   (in TFileLines DATFile);
    long sendNPVParams (in TNPVParams NPVParams);
    long sendFTPRequest ();
};
};

```

### Código IDL da interface – avaliação

#### Estrutura que define os estados do módulo avaliador

```

enum TStatus
{
    ready,      // host can compute any score from genome
    busy,       // host is running objective function
    error,      // internal error was raised (repair if possible)
    enabled,    // host can't evaluate, must be initialized
    disabled    // host can't evaluate and can't be initialized
};

```

#### Estrutura de dados com os valores retornados após avaliação

```

struct TVPLWells
{
    double VPL;      // Net Present Value
    double D;        // Development Cost
    double B;        // Reserve Size
    double q;        // Economic Quality
};

```

```

double    Rt;          // Incomes
double    COPt;       // Operational Cost
double    nPlatform; // Platforms Number
TStatus   status;     // Status after simulation
};

```

A seguir, as estruturas com os valores dos pontos de curva para uma alternativa, empregada na geração dos padrões para o modelo de pontos de curva de óleo acumulado da seção (4.5.2.1)

```

struct TValues
{
    double    p0090;
    double    p0180;
    double    p0360;
    double    p0720;
    double    p1800;
    double    p3600;
    double    p5400;
    double    p7200;
};
struct TSamplePoints
{
    TValues   oilPoints;
    TValues   gasPoints;
    TValues   waterPoints;
};

```

Estrutura com valores de óleo acumulado, gás e água, empregadas na geração dos mapas de qualidade e aquífero.

```

struct TCumuData
{
    double    cOil;      // Cumulative Oil value
    double    cWater;   // Cumulative Water
    double    cGas;     // Cumulative Gas
    TSamplePoints points;
};

```

Estrutura que contém as definições necessárias de um poço petrolífero, e a estrutura que contém a lista de poços. Esta estrutura é empregada para o envio de alternativas na otimização, na geração de amostras para os modelos de aprendizado e para a geração dos mapas de qualidade

```

struct TWell
{
    unsigned long wellOrder;
    unsigned long iAxis;
    unsigned long jAxis;
    unsigned long kAxis;
    unsigned long azimuthAngle; // 0 i-dir, 1 j-dir
    unsigned long wellGeometry; // 0 vert, 1 horiz, 2 directional
    long          length;       // may be negative values
    boolean       wellType;     // false : produc, true : injetor
};
typedef sequence<TWell>TWells;

```

Estrutura com informações relacionadas a uma alternativa e lista de alternativas

```

struct TAlternative
{
    string    alternativeName;
    double   alternativeCost;
    double   NPV;
    double   cumulativeOil;
    double   economicQuality;

    double   schAlternativeCost;
    double   schNPV;
    double   schCumulativeOil;
    boolean  isScheduled;
    boolean  isSimulated;
    TSamplePoints samplePoints;

    unsigned long platformsNumber;
    ACB::TOilCommodity  oilCommodities;
    ACB::TDevelopmentCost developmentCosts;
    ACB::TOperationalCost operationalCosts;
    TWells  wells;
};
typedef sequence<TAlternative> TAlternatives;

```

Estrutura relacionada à a ordem de entrada dos poços

```

typedef sequence<unsigned long> TOrderWellsGenes;

```

Interface definida para a comunicação entre o *master* e os diferentes *slaves* para o envio de alternativas e sua respectiva avaliação ou simulação. Note-se os vários serviços definidos nesta interface e os tipos de dados que são enviados e retornados.

```

// Interface used in communications of
// distributed GA and standalone parallel simulations
interface TCorbaEngine
{
    TStatus    checkStatus();
    void       sendAlternative    (in TAlternative alternative);
    TVPLWells  evaluateAlternative();
    TVPLWells  evaluateNPV       (in TWells wells);
    TVPLWells  evaluateNPVOrder  (in TOrderWellsGenes orderWellsGenes);
    TSamplePoints getSamplePoints (in TWells wells);
};

```