# VirD-GM: Modelagem e Funcionalidades do Construtor Iterativo Paralelo

**Marcelo Würdig[1], Felipe Munhoz, Wagner Al-Alan[2], Renata Reiser, Adenauer Yamin**

NAPI - Universidade Católica de Pelotas (UCPel)
Rua Félix da Cunha, 420 - 96075690 – Pelotas – RS – Brasil

{wurdig,fmunhoz,wagnerg,reiser,adenauer}@ucpel.tche.br

***Abstract*** *This paper introduces the parallel iteration process constructor and presents a case study in order to analyze the behavior and related features implemented in the D-GM environment.*

***Resumo.*** *Este trabalho introduz o construtor de iteração paralelo e apresenta um estudo de caso para avaliação do comportamento e das funcionalidades implementados no ambiente D-GM.*

## 1. Introdução

Este trabalho descreve a modelagem do construtor de iterações, que viabiliza a repetição sequencial e/ou paralela de operações, e sua integração ao ambiente de execução do Projeto D-GM (*Distributed Geometric Machine*) [Reiser et al. 2004], denominado VirD-GM (*Virtual Distributed Geometric Machine Model*) [Munhoz et al. 2008], incluindo um estudo de caso para avaliação do comportamento da versão paralela deste construtor denominado Construtor Iterativo Paralelo.

No contexto deste trabalho, compreende-se a integração entre o ambiente de desenvolvimento VPE-GM (*Visual Programming Environment for the Geometric Machine Model*) e o ambiente de execução VirD-GM, onde as aplicações geradas no VPE-GM podem ser executados com suporte a distribuição no VirD-GM. A especificação das possibilidades de paralelismo deve ser feita pelo programador no ambiente VPE-GM. Por sua vez o ambiente de execução analisa a disponibilidade atual das unidades computacionais e assim distribui os processos, considerando o estado atual de ocupação dos nodos do sistema paralelo onde está ocorrendo o processamento.

A organização do texto está distribuída da seguinte forma: Na Seção 2, apresentam-se a especificação em UML (diagrama de sequência) dos principais componentes de software da VirD-GM e seus relacionamentos. A descrição do Construtor Iterativo e principais características são apresentadas Seção 3. A seguir, na Seção 4., descreve-se a avaliação do comportamento do construtor iterativo paralelo incluindo os resultados obtidos e as considerações finais.

## 2. Principais Componentes de Software VirD-GM

Os componentes funcionais e lógicos da VirD-GM foram concebidos considerando uma visão de maior abstração quando do processo de execução e dão

[1] Bolsista BIC/UCPel

[2] Bolsista CAPES

suporte a execução dos atuais construtores que integram o ambiente: produtos sequencial e paralelo, somas determinísticas e não determinísticas, iteração e iteração paralela, construtor de macro, entre outros. As aplicações concorrentes desenvolvidas no ambiente de programação visual VPE-GM tem suas especificações exportadas para o ambiente VirD-GM. Por sua vez, pelo componente VirD-GM ocorre a análise da estrutura de processos (construtores produto paralelo e/ou de iteração paralela, operações) e de memória (dimensão, posições, tipos) e a verificação das dependências de dados no arquivo que descreve a aplicação modelada. Na sequência, tem-se a execução paralela/distribuída destes processos, de acordo com a disponibilidade atual do ambiente computacional. As principais funcionalidades dos componentes desenvolvidos para alcançar a execução concorrente dos processos no modelo D-GM são brevemente descritos.

O modulo *Loader VirD-GM* recebe os arquivos descritores de processos e de memórias da aplicação corrente, exportados pelo ambiente VPE-GM. Este módulo, além da leitura das estruturas de processos e de memória, também lê os parâmetros para execução. Consideram-se também as atividades de preparação da execução, assim como a criação da matriz de adjacência utilizada no controle do fluxo de execução.

O *Launcher VirD-GM* tem como principal funcionalidade o gerenciamento do controle e do disparo da execução, após mapeamento do arquivo XML para as estruturas internas que definem o módulo *Loader VirD-GM*. Este controle está baseado em políticas de escalonamento, previamente definidas quando da especificação da aplicação e compatíveis com a arquitetura do modelo D-GM.

A funcionalidade do módulo *Exec VirD-GM* consiste na execução das computações paralelas nos nodos remotos da célula, pela utilização de serviços do *middleware* EXEHDA. Esta execução tem suporte em bibliotecas auxiliares, previamente definidas pela aplicação em desenvolvimento. Para tal, faz-se necessário o uso de operações de acesso à memória compartilhada.

A Figura 1 descreve, a sequencialização das etapas com a participação de cada um dos componentes do VirD-GM visando a execução dos processos no modelo D-GM.

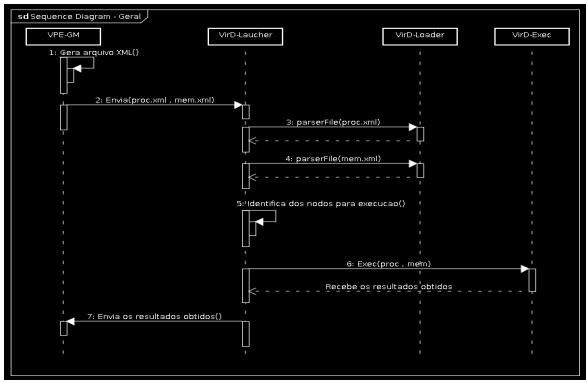


**Figura1. Diagrama de comunicação entre componentes do VirD-GM**

## 3. Construtor Iterativo

A implementação dos construtores está baseada na definição recursiva da estrutura de envelope e de processo elementar [Munhoz et al. 2008]. De acordo com as abstrações do modelo D-GM, o *Construtor Iterativo* provê a iteração sequencial de processos possivelmente pelo tipo envelope, incluindo a composição com outros construtores. Este construtor tem como principal característica a repetição de operações. O número de repetições do *Construtor Iterativo* é variável e definido por um parâmetro de controle no momento de sua geração no editor de processos VPE-GM.

A implementação do *Construtor Iterativo* foi realizada no módulo *VirD-Loader*. Durante a leitura do arquivo XML, caso encontrado um *Construtor Iterativo*, o processo descendente é replicado N vezes, onde N é o parâmetro definido o número de iterações do *Construtor Iterativo*. Além disso, o sistema verifica se o *Construtor Iterativo* é sequencial ou paralelo.

No caso de aplicação do *Construtor Iterativo Sequencial (*for_dgm*)*, além de replicar os processo e inserir na lista, é necessário resolver as dependências de dados entre os processos gerados.

O *Construtor Iterativo Paralelo* (pfor_dgm) tem o funcionamento análogo ao *Construtor Iterativo Sequencial*, exceto pela etapa de verificação de dependências internas. Na Figura 2, mostra-se um diagrama que descreve o funcionamento do C*onstrutor Iterativo Sequencial* e do *Construtor Iterativo Paralelo* respectivamente, sendo, este último caracterizado pela não dependência de dados entre as iterações, ou seja, cada iteração pode ser executada concorrentemente com as demais iterações.

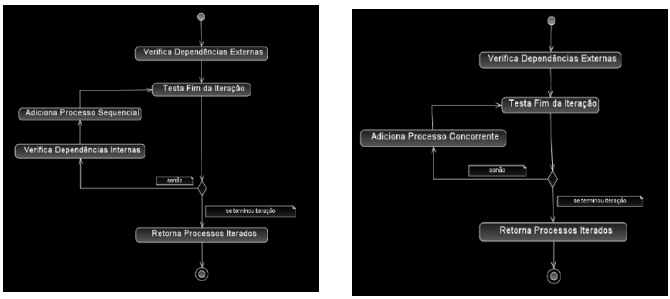


**Figura 2. Diagrama dos construtores de iteração: sequencial e paralelo**

## 4. Estudo de Caso para Avaliação do Construtor Iterativo Paralelo

Considera-se um estudo de caso, avaliando o comportamento do *Construtor Iterativo Paralelo*, apresentando as funcionalidades disponibilizadas no ambiente D-GM.

Primeiramente, o *Construtor Iterativo Paralelo* verifica as dependências externas, identificando os processos que precisam ser finalizados para início da corrente iteração. Após, testa a finalização da iteração pela verificação do número de iterações do processo. Assim, enquanto o número de iterações não está satisfeito, adiciona o processo a iteração corrente usando uma estrutura de lista de armazenamento temporário, já com as dependências de dados resolvidas. E quando o número de

iterações do processo finaliza, os processos gerados na iteração são enviados para a lista de processos de execução.

As aplicações executadas no VirD-GM utilizaram as ações do *Construtor Iterativo Paralelo.* Neste estudo de caso, tem-se um conjunto de nodos selecionados para a realização dos testes do tipo homogêneo quanto a sua capacidade de processamento, em grade de computadores *dual-core*. Pela definição da arquitetura da VirD-GM, um dos nodos distingue-se dos demais, sendo caracterizado como nodo VirD-base, responsável tanto pelo escalonamento das tarefas quanto pelo controle do fluxo de dados. O gráfico na Figura 3, tem como principal objetivo resumir os resultados em termos do *speedup* da aplicação. A diferenciação na execução de cada teste deste estudo de caso se dá pelo número de nodos passados como parâmetros de entrada, entretanto, salienta-se que os parâmetros de memória e dos processos, ambos gerados nas interfaces da VPE-GM, são idênticos para todos os casos de testes. A modelagem do VirD-GM possibilita que a alocação dinâmica do números de nodos seja passada como parâmetro de entrada.

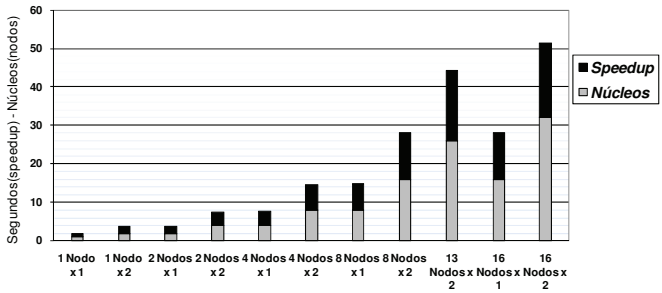


**Figura 3: Gráfico de *speedup* do estudo de caso**

Pela análise dos resultados, observa-se que com uma distribuição 16 (nodos) x 2 o desempenho atingiu um *speedup* de 19,30''. Entretanto, com 3 nodos a menos, 13 (nodos) x 2, o *speedup* já atinge 18,33''. Ou seja, o aumento no número de nodos em função de não ser múltiplo do número de processos, não provocou um aumento no desempenho. Sendo assim, na continuidade do trabalho, propõe-se o desenvolvimento de um módulo de análise, baseado na correlação entre processos concorrentes em execução e números de nodos disponíveis, para melhor gerenciamento da distribuição na grade computacional, visando uma contribuição no desempenho aliada à proposta de otimização de recursos.

## Referências

R. Reiser, A. C. R. Costa, and G. Dimuro. Distributed approach for the geometric machine model. In PARA 2004 - Workshop on State-of-the-art in Scientific Computing Valideted, number 1, pages 106-114, Ligby, 2004

F. N. Munhoz, G. Vivan, V. S. Fonseca, R. H. S. Reiser, M. L. Pilla, and A. C. Yamin. Arquitetura de software da vird-gm: Modelagem e funcionalidades. In ERAD, pages 1-4, 2008.

F.N. Munhoz. Expandindo o VirD-GM para Suporte as Demandas do Projeto D-GM. Ciência da Computação – UCPel. (Monografia de Projeto de Graduação).