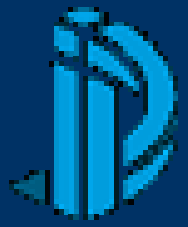
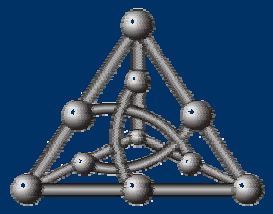




Running Highly-Coupled Parallel Applications in a Computational Grid

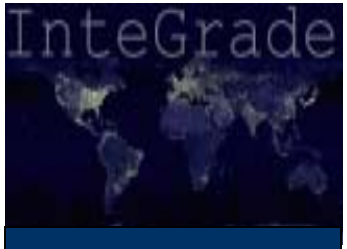
Andrei Goldchleger, Carlos Alexandre Queiroz,

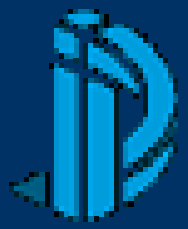
Fabio Kon, Alfredo Goldman

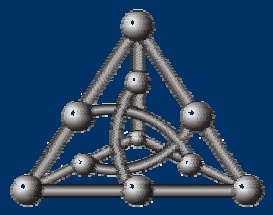
{andgold,carlosq,kon,gold}@ime.usp.br

http://gsd.ime.usp.br/integrade

IME - USP

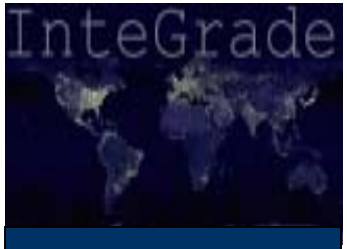


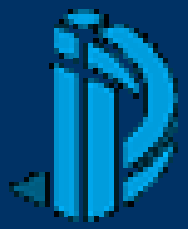


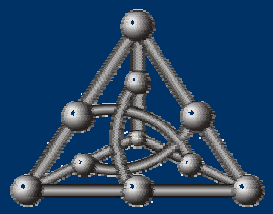


O Projeto InteGrade

- InteGrade é um sistema de Computação em Grade que visa utilizar a capacidade ociosa de máquinas compartilhadas. Suas principais características incluem:
  - Arquitetura Orientada a Objetos
  - Utiliza CORBA como infra-estrutura de comunicação
  - Oferece suporte a aplicações de paralelismo não trivial
  - Planeja coletar e utilizar padrões de uso das máquinas de maneira a melhorar o escalonamento
- Todo o software disponível na Incubadora da FAPESP
  - http://incubadora.fapesp.br/projects/integrade

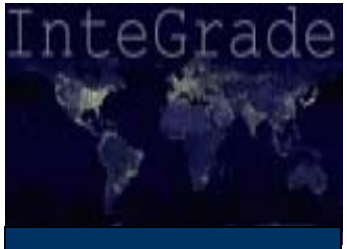


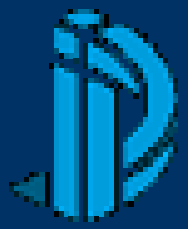


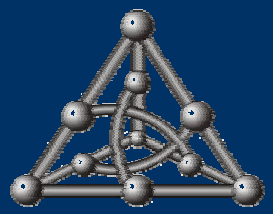


Objetivos do Trabalho

- Implementar suporte a aplicações paralelas sobre o InteGrade
- Utilizar um modelo paralelo já existente, e permitir que aplicações pré-existentes executem sobre o InteGrade sem alterações
- Modificar o mínimo possível as interfaces pré-existentes do InteGrade, mantendo a implementação da biblioteca paralela bem independente do restante do sistema

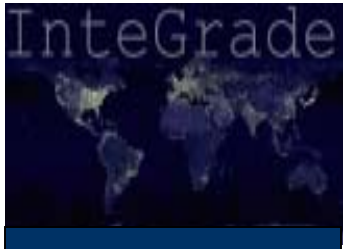


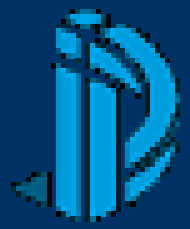


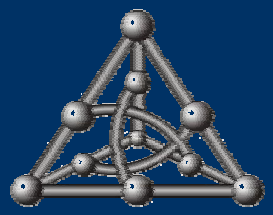


O Modelo BSP

- 1990 (Valiant, *A bridging model for parallel computation*)
- BSP foi o modelo escolhido, devido à:
  - Elegância
  - Simplicidade (A implementação de referência, Oxford BSPlib, possui apenas 20 funções)
- BSP organiza a computação em **superpassos**
  - 1º) Cada tarefa trabalha com valores disponíveis localmente
  - 2º) Cada tarefa comunica-se com as demais, se necessário
  - 3º) Ocorre uma barreira de sincronização entre as tarefas
- As comunicações só são efetivadas **no final** do superpasso

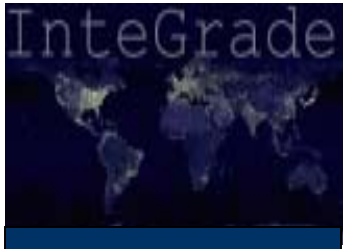


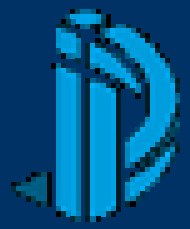


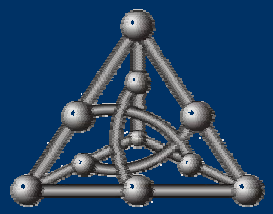


BSPlib no InteGrade

- A biblioteca BSP do InteGrade implementa a **mesma** interface da Oxford BSPlib para C, porém nossa implementação ainda é parcial
  - Porte de aplicações é direto: inclua cabeçalho, recompile e religue a aplicação com a biblioteca do InteGrade
- Oxford BSPlib possui dois tipos de comunicação entre processos:
  - DRMA (*Direct Remote Memory Access*): disponível no InteGrade
  - BSMP (*Bulk Synchronous Message Passing*): ainda não disponível no InteGrade, mas em processo de implementação

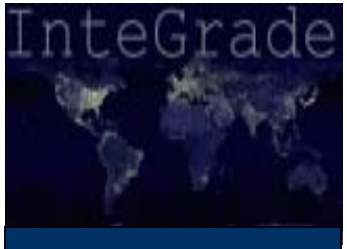


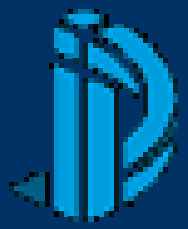


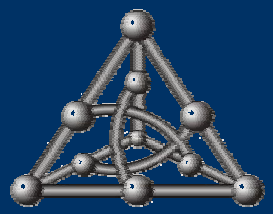


Principais Componentes do InteGrade

- LRM (*Local Resource Manager*)
  - Módulo executado em cada máquina que cede recursos para a Grade. Responsável por informar ao GRM a quantidade de recursos disponíveis na máquina, assim como receber requisições para execução de aplicações
- GRM (*Global Resource Manager*)
  - Recebe informações sobre disponibilidade de recursos enviadas pelos diversos LRM. Também é responsável pelo escalonamento das aplicações submetidas à Grade
- ASCT (*Application Submission and Control Tool*)
  - Ferramenta que permite que o usuário da Grade registre aplicações no Repositório de Aplicações, solicite execuções para o GRM, acompanhe o estado da execução e colete resultados das execuções

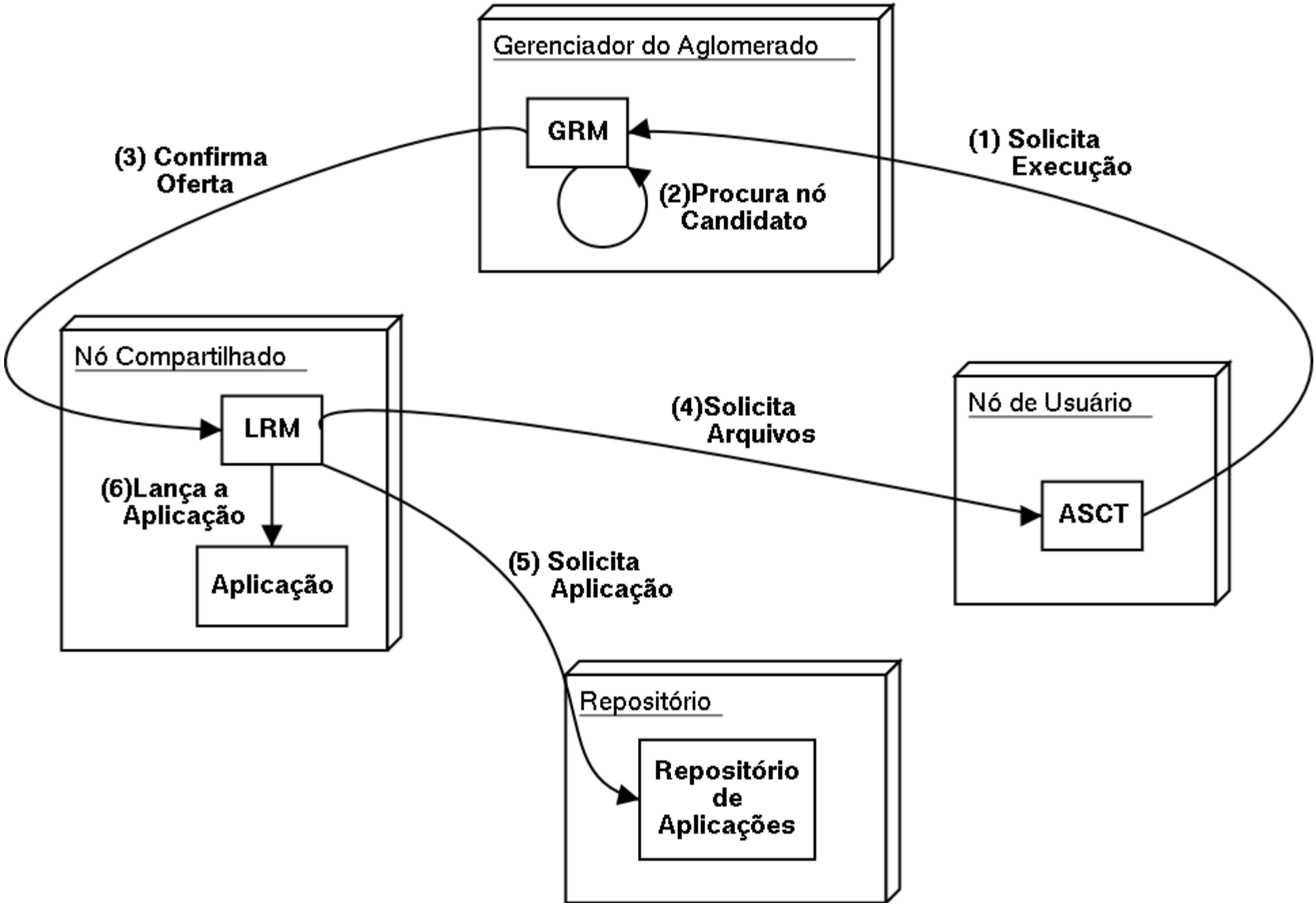


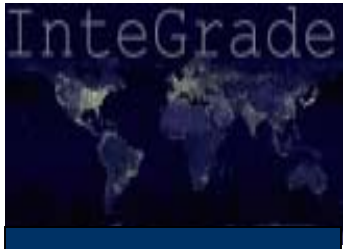


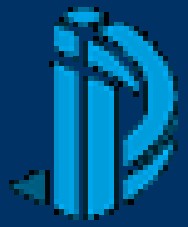


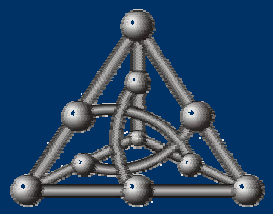
Execução de Aplicações no InteGrade

- (1) O usuário, por meio do ASCT, solicita a execução de uma aplicação previamente registrada no repositório
- (2) O GRM procura uma máquina apta a receber a aplicação
- (3) O GRM encaminha a requisição para o LRM de tal máquina
- (4) O LRM solicita ao ASCT que solicitou a execução os eventuais arquivos de entrada da aplicação
- (5) O LRM solicita a própria aplicação ao Repositório
- (6) O LRM executa a aplicação


Gerenciador do Aglomerado
GRM
(2)Procura nó Candidato
(1) Solicita Execução
(3) Confirma Oferta
Nó Compartilhado
LRM
(6)Lança a Aplicação
Aplicação
(4)Solicita Arquivos
Nó de Usuário
ASCT
(5) Solicita Aplicação
Repositório
Repositório de Aplicações



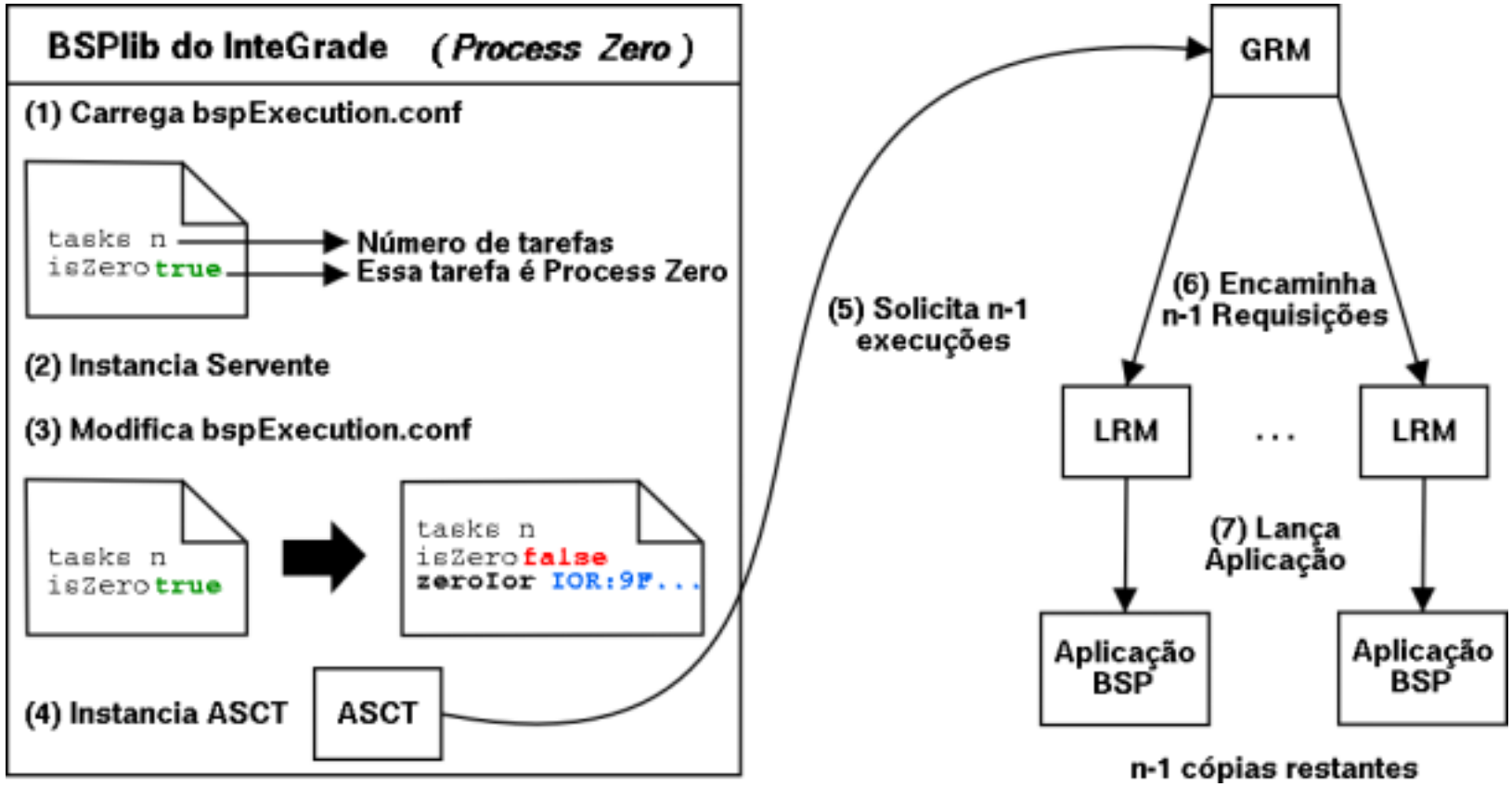




Execução de aplicações BSP

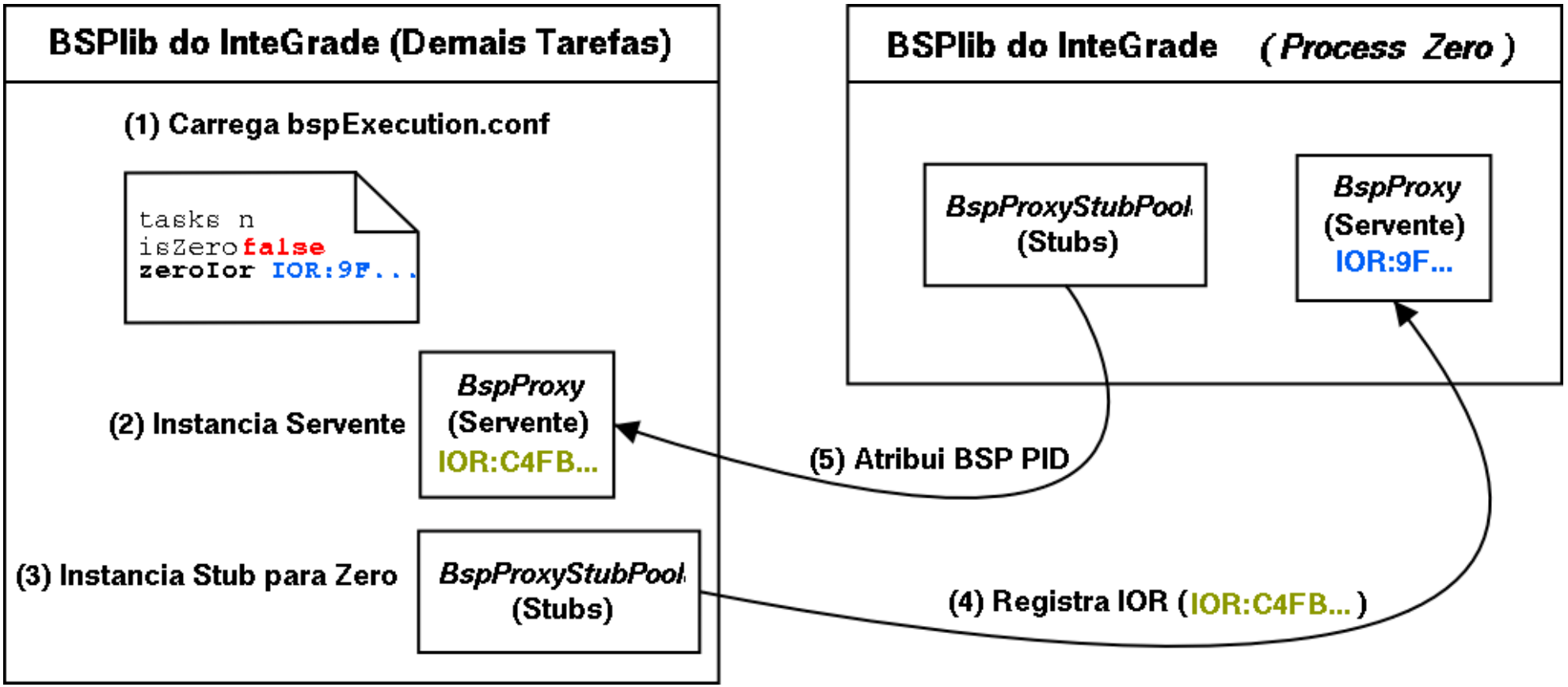
- Aplicações BSP são executadas de maneira diferente, porém não demandam alterações na interface do GRM. A execução ocorre em 2 estágios:
  - 1º) Lança-se uma cópia da aplicação. Esta cópia é chamada de *Process Zero* (Possui BSP PID 0). É responsável por:
    - Distribuir identificadores de processo BSP
    - Distribuir IORs das tarefas, permitindo que cada processo instancie *stubs* para os demais
    - Coordenar barreiras de sincronização
  - 2º) O *Process Zero* lança as cópias restantes
- Internamente, cada nó da aplicação BSP cria na inicialização:
  - *BspProxy*: servente CORBA responsável por receber mensagens
  - *BspProxyStubPool*: *pool* de *stubs* que permite que um nó da aplicação se comunique com os demais

Inicialização da aplicação BSP: *Process Zero*



Na inicialização de uma aplicação BSP, a biblioteca **(1)** carrega o arquivo bspExecution.conf e determina se tal cópia da aplicação é *Process Zero*. Caso seja, **(2)** a biblioteca instancia um *BspProxy*, **(3)** modifica o `bspExecution.conf` (a nova versão será solicitada pelas demais cópias) e **(4)** instancia um ASCT, utilizado para **(5)** solicitar a execução das tarefas restantes. O GRM procede da maneira usual, **(6)** encaminhando as requisições para LRMs, que **(7)** lançam as cópias restantes.

Inicialização da aplicação BSP: Cópias Restantes



Quando uma cópia da aplicação é lançada, a biblioteca BSPlib **(1)** carrega o arquivo `bspExecution.conf`, **(2)** instancia um *BspProxy* e **(3)** um *stub* para o *Process Zero*, o qual é utilizado para **(4)** registrar a IOR da tarefa no *Process Zero*. Este, por sua vez, **(5)** atribui um PID para a tarefa.

Para Saber Mais:
http://gsd.ime.usp.br/integrade