ESCOLA POLITÉCNICA
DE PERNAMBUCO

Departamento de
Sistemas
Computacionais

Modelo Formal em Redes de Petri para Verificação de Simulações

Trabalho de Conclusão de Curso

Engenharia da Computação

Renata Wanderley Medeiros
Orientador: Prof. Dr. Ricardo Massa Ferreira Lima

Recife, junho de 2007

UNIVERSIDADE
DE PERNAMBUCO

ESCOLA POLITÉCNICA
DE PERNAMBUCO

Departamento de
Sistemas
Computacionais

Modelo Formal em Redes de Petri para Verificação de Simulações

Trabalho de Conclusão de Curso

Engenharia da Computação

Este Projeto é apresentado como requisito parcial para obtenção do diploma de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Renata Wanderley Medeiros
Orientador: Prof. Dr. Ricardo Massa Ferreira Lima

Recife, junho de 2007

UNIVERSIDADE
DE PERNAMBUCO

Renata Wanderley Medeiros

Modelo Formal em Redes de Petri para Verificação de Simulações

Resumo

Este trabalho trata de simuladores, gerados pelo MPhyScaS (*Multi Physics and Multi Scale Solver Environment*), que envolve a simulações de fenômenos multifísicos acoplados. Estes são fenômenos que transformam a matéria, mas não alteram sua composição química. Neste contexto, a simulação de um problema real pode ser bastante complexa. Desta forma, os simuladores devem auxiliar o usuário a fazer verificações sobre os elementos inseridos e validá-los antes de iniciar a simulação. Este trabalho propõe uma representação das relações entres os dados de entrada dos simuladores gerados pelo MPhyScaS, bem como um modelo formal, utilizando a técnica de redes de Petri, baseado na representação definida. Este modelo justificará o prosseguimento para a fase de simulação, removendo erros que podem ocorrer durante esta fase.

Abstract

This work is about simulators, generated by MPhyScaS (*Multi Physics and Multi Scale Solver Environment*), that involve the simulation of coupled multiphysics phenomena. These are phenomena that transform the material, but they don't modify its chemical composition. In this context, the simulation of a real problem may be very complex. In this way, simulators must help users making verifications on inserted elements and validating them before starting simulation. This work proposes a representation to the relations of entry data for MPhyScaS simulators, in addition to a formal model, using the Petri nets technique, based on defined representation. This model will justify the passage to simulation phase, removing mistakes that may occur.

Sumário

Índice de Figuras iv

Tabela de Símbolos e Siglas v

1 Introdução 7

1.1 MPhyScaS 8
1.2 Visão Geral da Monografia 9

2 Redes de Petri 11

***3* Representação das Relações entre os Dados de Entrada do Simulador** 14

4 Especificação Formal 21

4.1 O modelo formal 21
4.2 Análises sobre o Modelo 28
4.2.1 Dependência entre Quantias 29
4.2.2 Quantia ativada ou fenômeno criado não é utilizado 30
4.2.3 Bloco ou Grupo criado não é utilizado 32
4.2.4 Estado de um Grupo não é utilizado 33

5 Estudo de Caso 35

5.1 A Simulação 35
5.2 O Modelo Criado 37
5.3 As Análises 41

6 Conclusões e Trabalhos Futuros 42

6.1 Contribuições 42
6.2 Trabalhos Futuros 43

Índice de Figuras

Figura 1-1 Organização do ambiente MPhyScaS. 9
Figura 2-1 Elementos das redes de Petri. 11
Figura 2-2 Exemplo de rede de Petri. 12
Figura 2-3 Representação de arco valorado. 12
Figura 2-4 Exemplo de disparo de uma transição. 13
Figura 3-1 Relação entre *Kernel*, Bloco, Grupo e Fenômeno. 15
Figura 3-2 Representação das relações entre os dados de entrada. 16
Figura 3-3 Representação com as relações de *Kernel* possui Blocos e Bloco possui Grupos. 17
Figura 3-4 Representação com as relações de Grupo possui Fenômenos, *GroupTasks* e gerencia Estados. 18
Figura 3-5 Representação com as relações de Quantia contribui para um Estado e *GroupTasks* solicita ao Fenômeno o cálculo de uma Quantia. 19
Figura 3-6 Representação com as relações de acoplamento. 20
Figura 4-1 Exemplo de especificação para *Kernel*, Bloco e Grupo e suas relações. 22
Figura 4-2 Exemplo de especificação para Fenômenos e *GroupTasks*, e suas relações com os Grupos. 23
Figura 4-3 Exemplo de especificação para Quantias e Estados, suas relações entre si e com os Fenômenos. 24
Figura 4-4 Exemplo de especificação quando uma Quantia contribui para mais de um Estado. 25
Figura 4-5 Exemplo de especificação quando acontece acoplamento. 25
Figura 4-6 Exemplo de especificação para o relacionamento entre Grupo e Estado. 26
Figura 4-7 Exemplo de especificação para solicitação do cálculo de uma Quantia pelo *GroupTask*. 27
Figura 4-8 Exemplo de um modelo completo. 28
Figura 4-9 Exemplo de rede de Petri com os Estados que auxiliarão as análises. 29
Figura 4-10 Exemplo de dependência entre Quantias. 30
Figura 4-11 Estado inicial para a análise que verifica se existe Quantia ativa ou Fenômeno criado que não é utilizado. 31
Figura 4-12 Rede de Petri durante a análise, destacando as transições habilitadas. 32
Figura 4-13 Estado inicial da rede de Petri para a análise que verifica se há algum grupo ou bloco criado que não é utilizado. 33
Figura 4-14 Estado inicial da rede de Petri para a análise que verifica se algum Estado criado não é utilizado, destacando as transições que poderão disparar quando o *token* chegar ao Grupo1. 34
Figura 5-1 Geometria do problema. 36
Figura 5-2 Resultado da simulação quando a concentração do soluto varia no tempo. 37
Figura 5-3 Resultado da simulação quando a concentração do soluto varia no tempo e no espaço. 37
Figura 5-4 Parte do modelo que possui as relações entre *Kernel*, Bloco, Grupo e Fenômeno. 38
Figura 5-5 Parte I do modelo que possui as relações entre Fenômeno, Quantia e Estado. 38
Figura 5-6 Parte II do modelo que possui as relações entre Fenômeno, Quantia e Estado. 39
Figura 5-7 Parte III do modelo que possui as relações entre Fenômeno, Quantia e Estado. 39
Figura 5-8 Parte I do modelo que possui as relações entre Grupo, *GroupTask* e Quantia. 40
Figura 5-9 Parte II do modelo que possui as relações entre Grupo, *GroupTask* e Quantia. 40
Figura 5-10 Parte III do modelo que possui as relações entre Grupo, *GroupTask* e Quantia. 40

Tabela de Símbolos e Siglas

MPhyScaS – *Multi Physics and Multi Scale Solver Environment.*
A.D.R. - *Applied Data Research Inc.*
FEM - *Finite Element Method.*
INA – *Integrated Net Analyzer.*
CRA – *Corrosion Resistant Alloys.*

Agradecimentos

Antes de tudo a Deus, com quem falo diariamente e, apesar de não obter respostas, sinto-o do meu lado, como força, luz e amor.

Ao meu pai Olavio, minha mãe Ione, minha tia Orquídia, minha irmã Roberta e meu cunhado Adriano, presença constantes em meu cotidiano, pela alegria, compreensão, estímulo e ajuda que me deram, permitindo-me concretizar esta tarefa que expressa um instante singular de minha vida.

Ao meu namorado William, por ter estado ao meu lado durante esta trajetória fazendo as figuras dessa monografia e me dando apoio e carinho sempre que precisei. E a seus familiares, pelo carinho e compreensão recebidos de cada um.

Aos meus amigos Tássia e Fernando, e aos outros queridos amigos e colegas, pelo companheirismo e espírito de luta, pois não sei, sinceramente, se, sem eles, teria terminado este trabalho.

Por fim, ao professor Ricardo Massa que, com eficácia de seus ensinamentos, contribui, de modo objetivo, para a minha formação.

E, como não posso esquecer de ninguém, a todos os outros que contribuíram, direta ou indiretamente, para que este trabalho fosse finalizado.

Capítulo 1

Introdução

Em problemas do mundo real, desenvolver sistemas e a partir deles auferir informações quantitativas e qualitativas sobre suas características pode acarretar em custos e riscos excessivos. Construir e implementar um modelo de simulação, por sua vez, é uma alternativa que pode economizar recursos financeiros e de tempo, bem como minimizar o acontecimento de riscos. De posse desse modelo implementado, poderá ser possível realizar simulações próximas ou equivalentes às funcionalidades do sistema real. Desta forma, a construção do sistema pode ser realizada ao final, tomando como base as informações que forem extraídas dessas simulações.

Desenvolver e implementar um simulador é uma tarefa complexa. Os simuladores, para serem eficientes, devem ser capazes de acompanhar o crescimento da complexidade dos sistemas a serem simulados. O MphyScaS (*Multi Physics and Multi Scale Solver Environment*) propõe uma metodologia para dar assistência à construção automatizada de simuladores que envolvem fenômenos físicos. Estes são fenômenos que transformam a matéria, mas não alteram sua composição química.

Como os simuladores gerados pelo MPhyScaS analisam problemas reais, uma simulação que envolva vários fenômenos que interagem durante sua evolução pode ser bastante custosa, levando horas ou dias para ser realizada. Isto acontece porque há uma forte dependência e compartilhamento de dados nas leis de comportamento nos quais os fenômenos são definidos [1]. Por isso, é extremamente importante, para o usuário do simulador, verificar e validar se os dados inseridos estão corretos, ou seja, verificar sua simulação antes de iniciá-la.

Uma forma de facilitar o entendimento do funcionamento e comportamento, dos dados inseridos no simulador pelo usuário, é criar um modelo formal. Este modelo deve justificar o prosseguimento para a fase de simulação, removendo erros que podem ocorrer durante esta fase.

Antes que esse modelo seja definido, é necessário que uma representação para as relações entre os dados inseridos no simulador pelo usuário seja criada. Ela deve representar fielmente a estrutura e organização dos dados de entrada do simulador e relações existentes, constituindo uma abstração do sistema [2]. É esta representação que guiará a criação do modelo, que permitirá responder questões prévias sobre a simulação.

Esta monografia define uma proposta de uma representação das relações entre os dados inseridos no simulador. Define também um modelo formal, utilizando a técnica de redes de Petri, baseado na representação definida. Este modelo permitirá que verificações quanto à corretude dos dados sejam feitas, informando ao usuário seus resultados para que possam ser validados antes de iniciar a simulação.

1.1 MPhyScaS

Simulação é um método bastante relevante para resolver vários problemas. É usada para estudar o comportamento e reações de um determinado sistema através de modelos que imitam, na totalidade ou em partes, propriedades e comportamentos desse sistema, permitindo sua manipulação e estudo detalhado [3].

Quando o desenvolvimento e implementação de um sistema real é completamente inviável, devido a incertezas quanto aos custos e riscos que o sistema poderia causar, esta técnica pode ser aplicada. Sistemas como a manutenção de dutos de cargas químicas podem levar a perda de vidas se feitas de maneira inadequada, ou mesmo se não forem feitas em tempo estimado. Para este sistema, é necessário verificar a evolução do desgaste do material no tempo, a fim de que danos não venham a ocorrer. Cálculos como este são muito complexos, pois envolvem muitos fenômenos físicos atuando em uma geometria contínua [4], tornando este um importante problema computacional. Isto se deve ao fato de haver uma forte dependência e compartilhamento de dados nas leis de comportamento onde os fenômenos físicos são definidos.

O projeto e implementação de simuladores para Fenômenos físicos é um problema computacionalmente impraticável se não houver poderosas ferramentas para auxílio [5]. A utilização de técnicas de reuso [6] e a utilização de componentes de *software* [7] tornariam o desenvolvimento desses simuladores mais eficiente.

O MPhyScaS (*Multi Physics and Multi Scale Solver Environment*) é uma proposta de ambiente que dá assistência à construção automatizada de simuladores multi-físicos. Para resolver problemas desta natureza, ele se baseia no Método do Elemento Finito (FEM – *Finite Element Method* [8]). É uma extensão do trabalho proposto por Cruz [9]. Os simuladores criados pelo MPhyScaS poderão ser de várias classes, sendo estipulado pelo engenheiro especialista no assunto, e não somente de uma única espécie.

O MPhyScaS é composto por três sistemas distintos:

i. Gerador/Configurador de Simuladores: responsável por integrar os componentes de *software* ao núcleo do simulador a fim de produzir simuladores específicos para propósitos diferentes, além de gerenciar os componentes de *software* que fazem parte de um simulador existente;
ii. Simulador: produto gerado pelo Gerador/Configurador de Simuladores, responsável por fazer simulações utilizando os componentes de *software* disponíveis.
iii. Sistema de Gerenciamento de Componentes: funciona como um banco de dados e é responsável por armazenar e gerenciar componentes de *software*, diferenciando suas utilidades e verificando a corretude de cada um deles.

A Figura 1-1 ilustra a organização desses sistemas.

Existem dois tipos de usuários para o sistema: (i) um especialista, que utiliza o Gerador/Configurador de Simuladores para gerar um simulador específico solicitado por um usuário; (ii) um usuário, que faz diversas simulações, no simulador solicitado e criado pelo especialista, podendo configurar algumas partes desse simulador a partir de componentes disponíveis.

O MPhyScaS é construído sobre uma linguagem de padrões, permitindo assim a utilização de diversos componentes de *software* [5], ou seja, se esse componentes forem construídos seguindo os padrões definidos, eles são capazes de serem reconhecidos e utilizados tanto pelo Gerador/Configurador de Simuladores como pelo Simulador.

O Mphyscas possui um conjunto de abstrações, que são capazes de representar os Fenômenos e suas interações (dependência e compartilhamento de dados) e os algoritmos de solução empregados pelo Simulador (cálculo de Quantias e solução de problemas auxiliares).

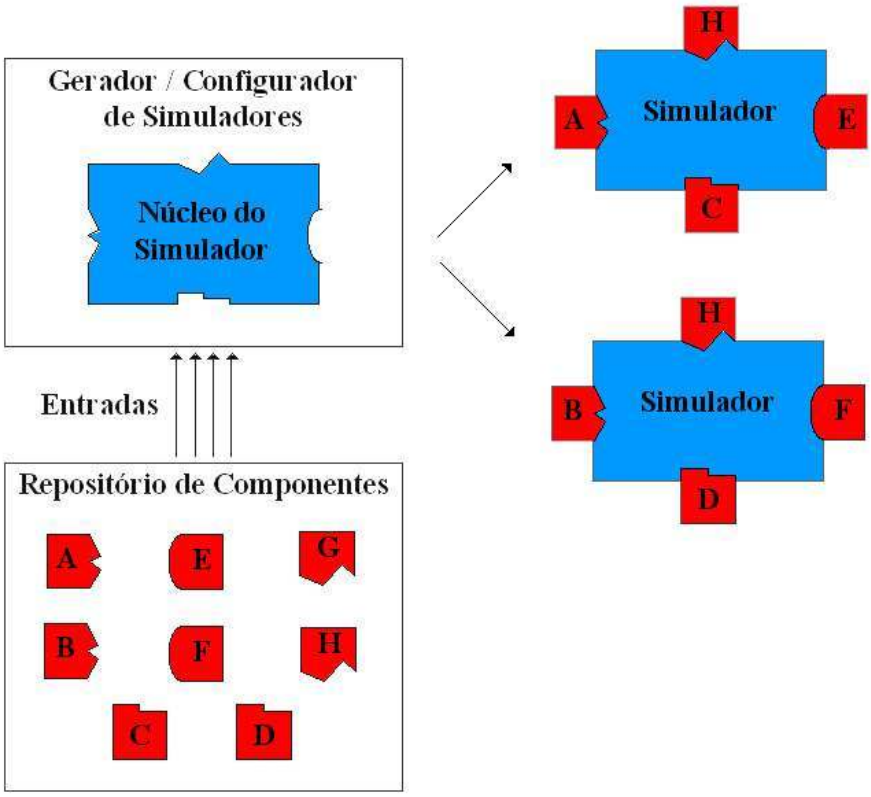


Figura 1-1 Organização do ambiente MPhyScaS.

Para facilitar o entendimento sobre Quantias, será feito uma analogia entre um Simulador e um sistema de equações. Se estes fossem equivalentes, um Fenômeno seria equivalente a uma equação do sistema, e uma Quantia, a uma variável de uma equação. O cálculo de uma Quantia seria o mesmo que achar o valor da variável, mas poderia ser necessário que o valor de outra variável já tivesse sido encontrado, o que representa o acoplamento entre Fenômenos.

1.2 Visão Geral da Monografia

O Capítulo 2 apresenta a conceituação de redes de Petri, técnica que foi utilizada para criar o modelo formal para verificações dos dados de entrada. Identifica seus elementos, suas propriedades, limitações e extensões existentes.

O Capítulo 3 contextualiza o sistema MPhyScaS, ressaltando sua importância, os sistemas que o compõem e seus modos de utilização. O capítulo apresenta também a arquitetura de *software* criada para representar as interações entre os dados de entrada do simulador.

O Capítulo 4 explica como o modelo formal foi definido, detalhando como o mesmo deve ser criado de acordo com os dados de entrada do simulador inseridos pelo usuário. Define como cada componente deve ser especificado e como suas relações devem ser modeladas para que seja possível fazer as análises que darão informações sobre os dados. Mostra ainda como as análises devem ser feitas no modelo em redes de Petri e como devem ser interpretados os resultados obtidos para fornecer informações essenciais ao usuário.

O Capítulo 5 exemplifica a utilização deste trabalho em um estudo de caso realizado em um simulador real para simular a difusão de hidrogênio em dutos de petróleo.

O Capítulo 6 conclui o trabalho, ressaltando as contribuições obtidas, e identifica trabalhos futuros que contribuirão para extensão deste trabalho em outras partes do MPhyScaS.

Capítulo 2

Redes de Petri

As redes de Petri foram criadas a partir da tese de doutorado de Carl Adam Petri, intitulada *Kommunication mit Automaten* (Comunicação com Autômatos), apresentada, em 1962, à faculdade de Matemática e Física de Darmstadt na então Alemanha Ocidental.

As primeiras aplicações de redes de Petri aconteceram em 1968, no projeto norte-americano *Information System Theory*, da A.D.R. (*Applied Data Research Inc.*). Muito da teoria inicial, da notação e da representação de redes de Petri foram desenvolvidas nesse projeto e publicadas em seu relatório final. Esse trabalho ressaltou como as redes de Petri poderiam ser aplicadas na análise e na modelagem de sistemas com componentes concorrentes [10].

Redes de Petri é uma técnica de especificação formal bem estabelecida, largamente difundida e adequada para a modelagem de sistemas que tenham atividades paralelas, concorrentes, assíncronas e não-determinísticas [11]. Hoje, é possível verificar sua abrangência e aplicabilidade em diversas áreas, tais como: na ciência da computação, engenharias eletrônica e química, administração de empresas. A técnica tem sido mais explorada para especificação de sistemas de hardware ou *software*, avaliação de desempenho, especificação de protocolos de comunicação, diagnóstico de falhas e no projeto de *software*/*hardware*. Isto ocorre devido ao conjunto de elementos que possui ser capaz de descrever partes de sistemas com características de concorrência, controle, conflito, sincronização e compartilhamento [12].

A representação gráfica das redes de Petri tem sido muito útil por permitir a visualização dos processos e a comunicação entre eles. As redes de Petri são compostas por dois elementos: um passivo denominado de lugar; e um ativo denominado de transição. Os lugares representam uma condição, uma atividade ou um recurso e são representados graficamente por círculos. As transições representam um evento (ação) realizado pelo sistema; graficamente, são representados por um traço ou uma barra. A Figura 2-1 mostra a representação gráfica dos elementos das redes de Petri.

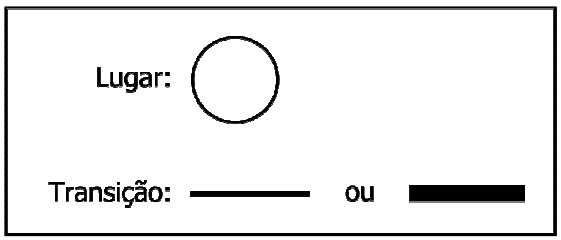


Figura 2-1 Elementos das redes de Petri.

Esses dois elementos são os vértices do grafo associado às redes de Petri. Os vértices são interligados por arcos dirigidos, como pode ser visto na Figura 2-2. Os arcos que interligam lugares às transições correspondem à relação entre as condições verdadeiras, que em um dado momento, possibilitam a execução das ações, enquanto os arcos que interligam transições aos lugares representam a relação entre as ações e as condições que se tornam verdadeiras com a execução das ações. Desta forma, uma rede de Petri é um grafo direcionado com dois tipos de nós, não havendo arcos entre nós do mesmo tipo [13].

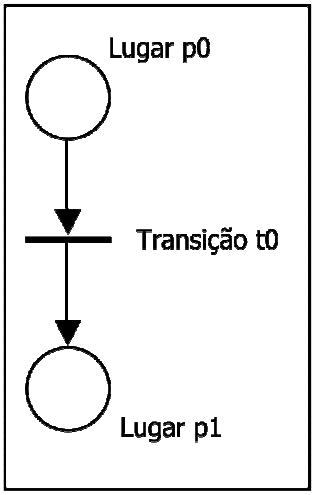


Figura 2-2 Exemplo de rede de Petri.

Os lugares podem armazenar marcas ou *tokens*, representados graficamente por pontos pretos. Uma distribuição de *tokens* na rede é chamada de marcação e corresponde ao Estado da rede de Petri. Além dos *tokens*, ainda é possível que os vértices de um grafo sejam interligados por múltiplos arcos. A Figura 2-3 ilustra um exemplo, com que um lugar pode ser conectado a uma transição através de diversos arcos ou vice-versa. Por conveniência, os múltiplos arcos são substituídos por um único arco valorado, onde o numeral associado ao arco corresponde ao número de arcos que interligam os vértices.

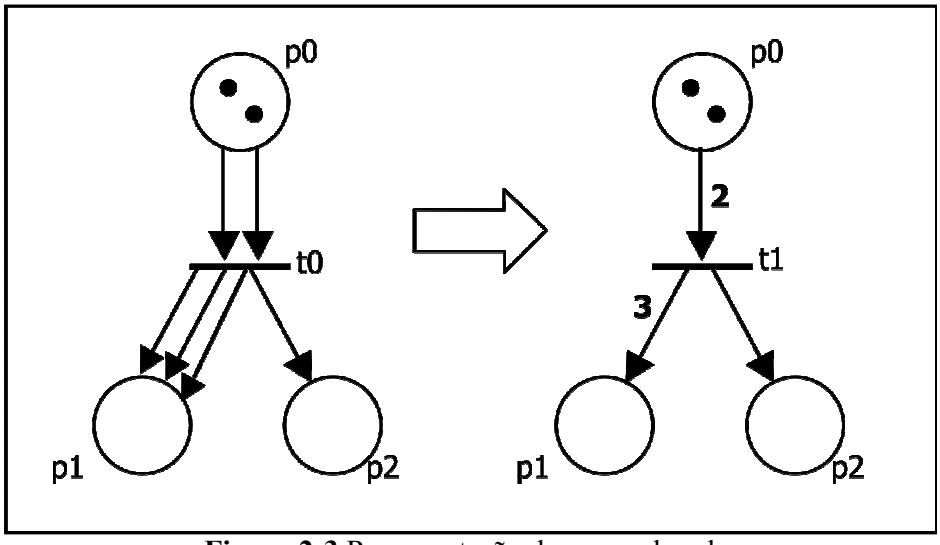


Figura 2-3 Representação de arco valorado.

Pode-se dizer que uma transição está habilitada se todos os lugares de entrada, ou seja, lugares de onde partem arcos em direção a esta transição têm ao menos o número de *tokens* igual ou maior que o número de arcos que ligam o lugar a esta transição. Uma transição habilitada pode disparar e seu disparo, ou realização de uma ação, está associado a algumas pré-condições, ou seja, existe uma relação entre os lugares e as transições, que possibilita a realização de uma ação. De forma semelhante, após a realização de uma ação, alguns lugares terão suas informações alteradas (pós-condições). A Figura 2-4(a) mostra uma rede de Petri com uma transição disponível, que, quando disparada, causa mudanças na marcação, deixando-a como a Figura 2-4(b).

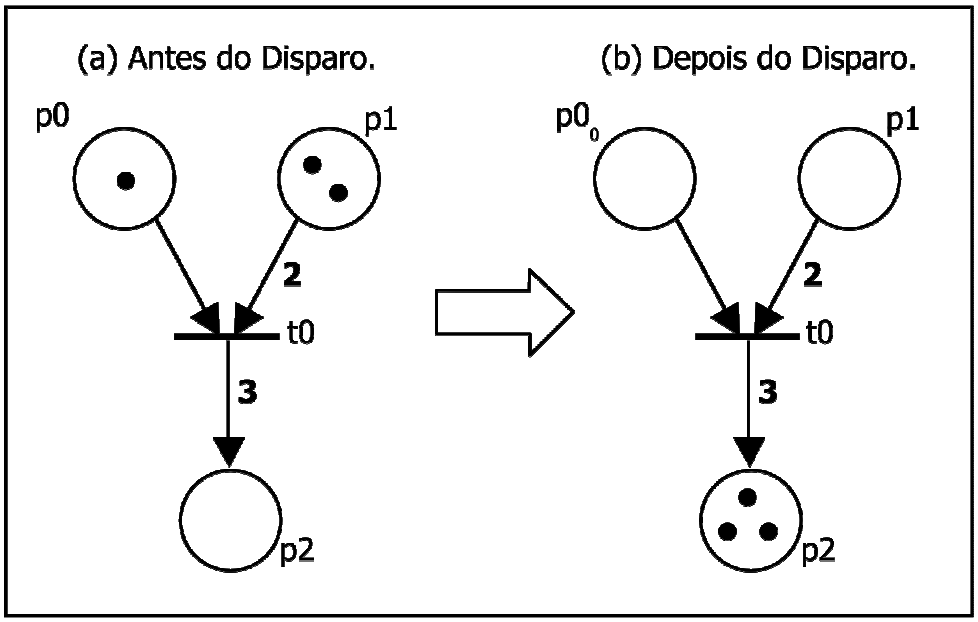


Figura 2-4 Exemplo de disparo de uma transição.

O Estado de uma rede, em um determinado instante, é representado pela quantidade de *tokens* em cada lugar da rede de Petri. O Estado da rede é alterado quando ocorre ao menos um disparo. Como exemplo dessa representação, tem-se que o Estado da rede da Figura 2-4(a) é M(1, 2, 0), que representa a quantidade de *tokens* dos lugares p0, p1 e p2, respectivamente. Ao acontecer o disparo da transição t0, esse Estado é alterado para M(0, 0, 3), que é o Estado da Figura 2-4(b).

Desde o trabalho original de Carl Adam Petri têm surgido diversas variantes ao seu modelo de redes de Petri. Pode-se afirmar que a maior parte destas variantes nasceu da necessidade de adaptação à especificidade da aplicação para as quais a sua utilização era desejada. O modelo original das redes de Petri falha na representação de duas importantes características: aspectos funcionais complexos, tais como: condições que determinam o fluxo de controle, e os aspectos de temporização [11]. Para enfrentar essas limitações, extensões das redes de Petri foram desenvolvidas. Alguns exemplos delas são: as Redes de Petri Temporizadas [14], as Redes e Petri Coloridas [15], as Redes de Petri Estocásticas [16], as Redes de Petri Estocásticas Generalizadas [17].

Capítulo *3*

Representação das Relações entre os Dados de Entrada do Simulador

Criar um modelo que represente as relações entre os dados inseridos no simulador guiará a definição do modelo formal. Este modelo é importante, pois:

i. facilita a comunicação entre as partes interessadas no projeto, pois oferece uma representação com alto nível de abstração;
ii. permite tomada de decisões antes de começar a implementação da criação do modelo formal e das análises;
iii. possibilita que partes dessa representação sejam reutilizadas para outro sistema.

Estas razões influenciam diretamente na antecipação de decisões em relação a estratégias utilizadas para o desenvolvimento do projeto. Isto acontece porque a tomada dessas decisões é um processo que requer experiência de quem o faz. Essas decisões podem, se inadequadas, impossibilitar o projeto devido a gastos excessivos e ao tempo perdido para corrigir o que foi feito. Com a representação feita, essas decisões podem ser tomadas de forma mais segura.

Para isto, a representação deve identificar os componentes e suas interações. Além disso, deve ser possível entender os requisitos do sistema, a fim de que desenvolvedores e testadores entendam o trabalho que lhes foi atribuído e o gestor, as implicações do planejamento estratégico. Os desenvolvedores devem se restringir às estruturas definidas na representação e os arquitetos devem supervisionar e verificar a adequação desse modelo, registrando impactos, riscos e dificuldades. As informações obtidas com esse registro contribuem para a evolução do modelo [1].

Para o melhor entendimento das dependências entre os componentes *Kernel*, Bloco, Grupo e Fenômenos, que estão presentes na representação criada para o MPhyScaS, a Figura 3-1 representa a relação entre esses componentes, onde o de maior nível possui componentes do nível imediatamente inferior. Os níveis estão representados, na Figura 3-1, de forma que o componente mais acima da figura é o de maior nível.

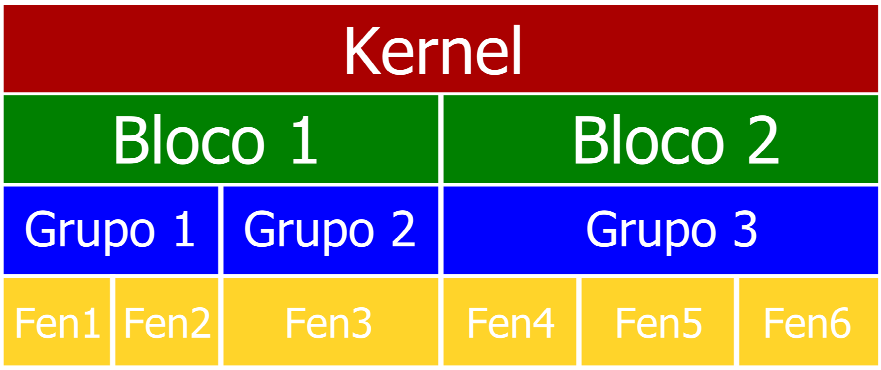


Figura 3-1 Relação entre *Kernel*, Bloco, Grupo e Fenômeno.

O *Kernel*, componente de maior nível na Figura 3-1, é único, pois representa o programa principal, que inicia a execução do simulador e o controla, verificando o seu término. Ele possui um ou mais Blocos, que por sua vez, cada um deles possui um ou mais Grupos. Cada Grupo possui um ou mais Fenômenos. Não faz sentido haver um componente de um nível se este não pertencer a um componente do nível imediatamente superior. Como exemplo disso, tem-se que não faz sentido haver um Fenômeno que não pertença a um Grupo.

A Figura 3-2 mostra a representação das relações dos dados inseridos no simulador do MPhyScaS, um dos objetivos deste trabalho. Nela estão representados seus componentes (*Kernel*, Bloco, Grupo, Fenômeno, Quantia, Estado, *GroupTask* – tarefa do Grupo), suas interações e dependências.

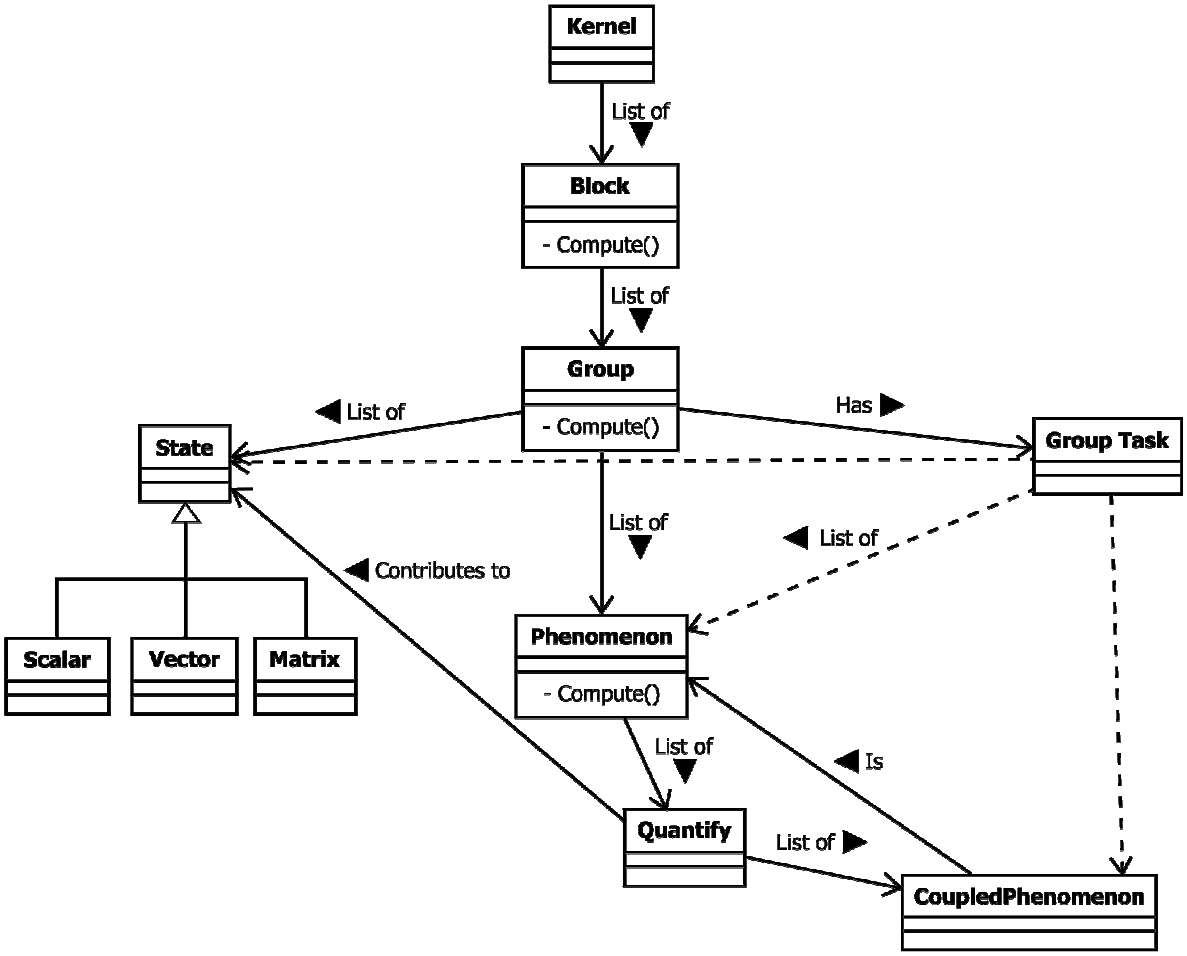


Figura 3-2 Representação das relações entre os dados de entrada.

As relações que indicam que o *Kernel* possui Blocos e cada Bloco possui Grupos já foram mencionadas. Estas relações estão presentes na arquitetura de *software* e podem ser identificadas mais facilmente na Figura 3-3, que destaca as relações mencionadas.

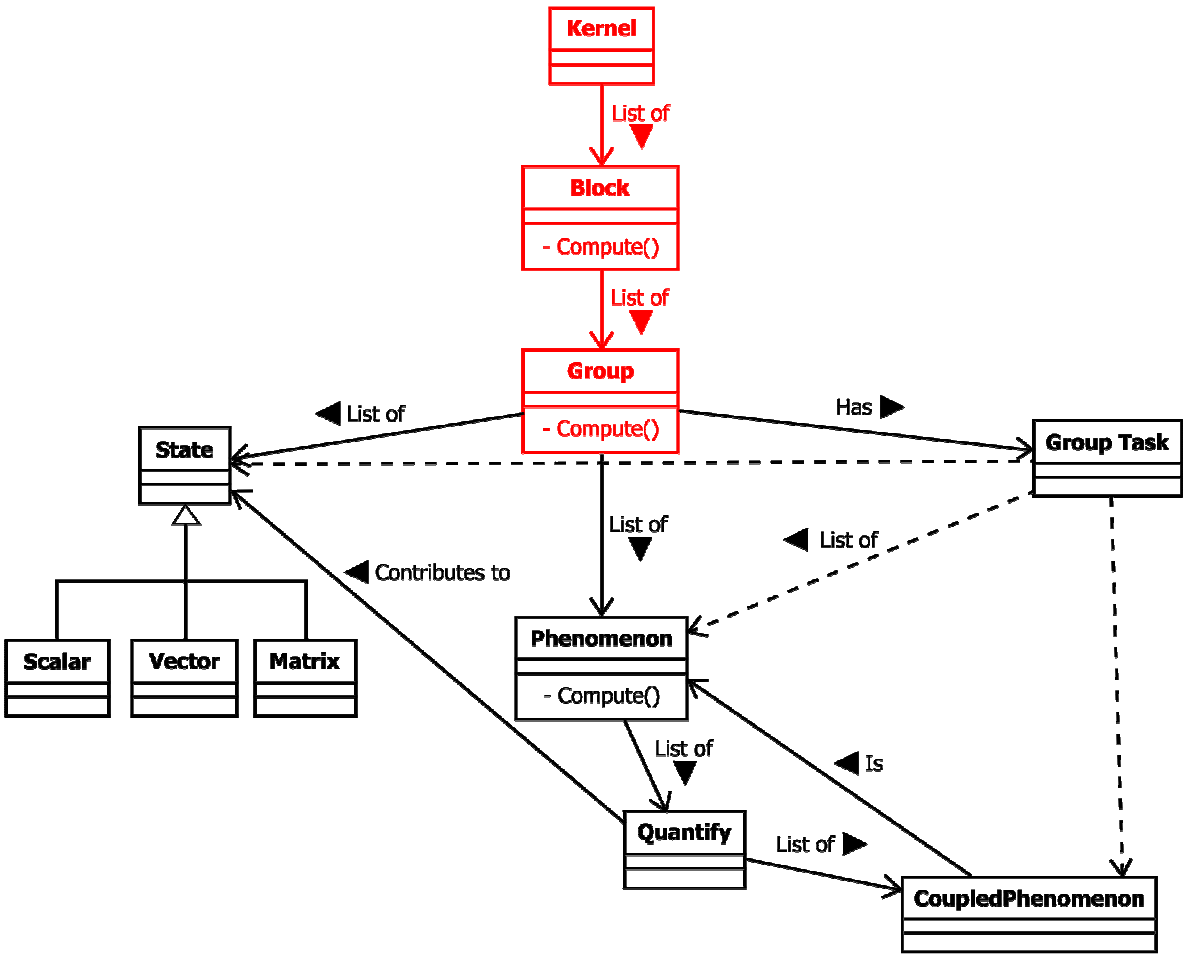


Figura 3-3 Representação com as relações de *Kernel* possui Blocos e Bloco possui Grupos.

Os Estados são estruturas de dados, como matriz, vetor ou escalar, onde serão armazenados os resultados obtidos nos cálculos das Quantias. Uma Quantia pode contribuir com valores em mais de uma estrutura, ou seja, uma Quantia pode contribuir com mais de um Estado. Os *GroupTasks* são tarefas que serão executadas pelo Grupo. Uma tarefa do Grupo é responsável por solicitar a um Fenômeno desse Grupo que calcule uma de suas Quantias.

Como dito anteriormente, o *Kernel* representa o programa principal, que inicia a execução do simulador e o controla, verificando o seu término. O Bloco está associado a um algoritmo de resolução, o que significa que cada Bloco seguirá esse algoritmo para resolver questões sobre os seus Grupos. O Grupo é responsável por gerenciar seus Fenômenos, Estados e *GroupTasks*. As relações que indicam que um Grupo possui Fenômenos, *GroupTasks* e é responsável por gerenciar um conjunto de Estados estão destacadas na Figura 3-4.

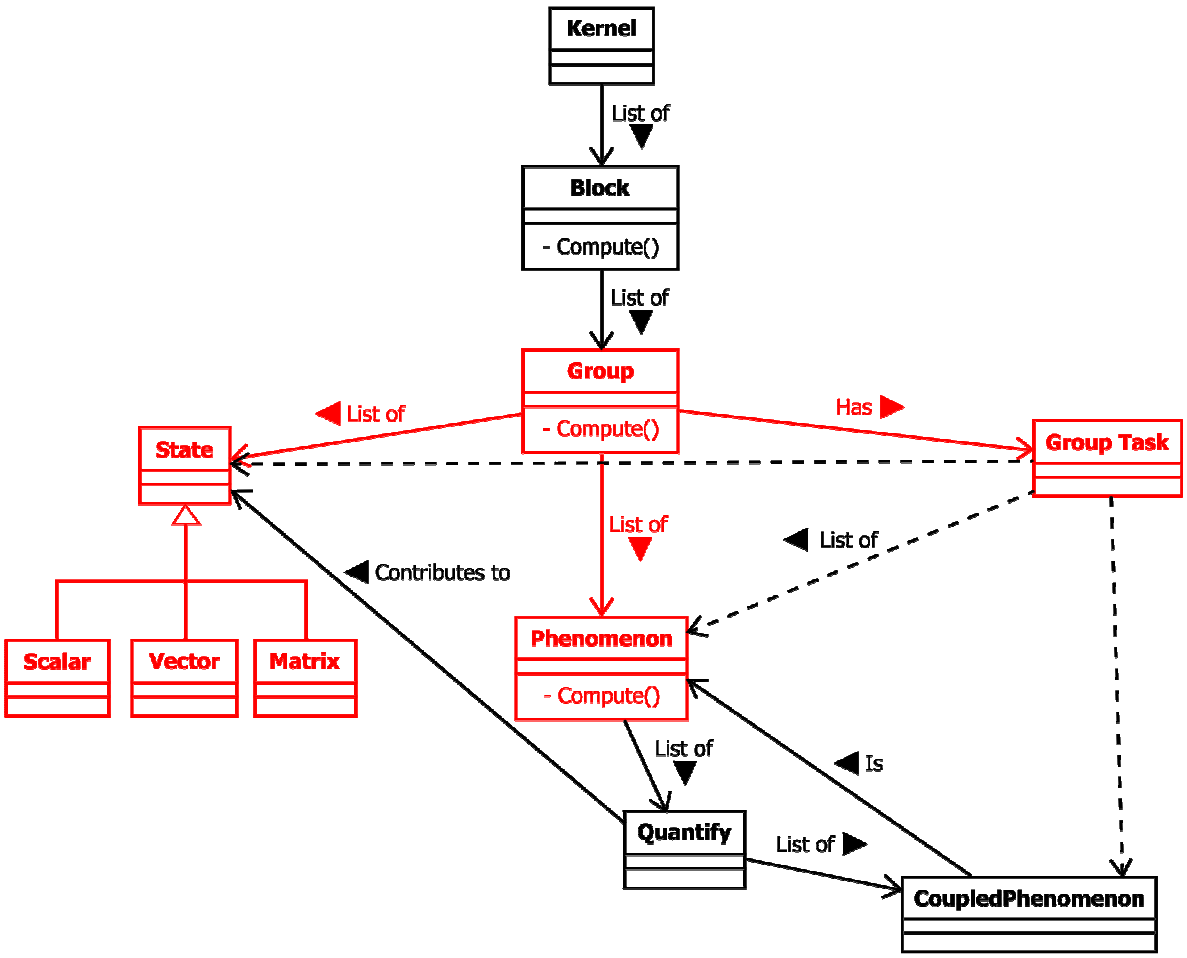


Figura 3-4 Representação com as relações de Grupo possui Fenômenos, *GroupTasks* e gerencia Estados.

Cada Fenômeno possui um conjunto de Quantias que podem ser calculadas por ele. Calcular uma Quantia significa contribuir para o preenchimento, total ou parcial, de um Estado (matrizes, vetores ou escalares). Os *GroupTasks* solicitam aos Fenômenos que calculem uma determinada Quantia pertencente ao Fenômeno. Estas relações são destacadas na representação mostrada na Figura 3-5.

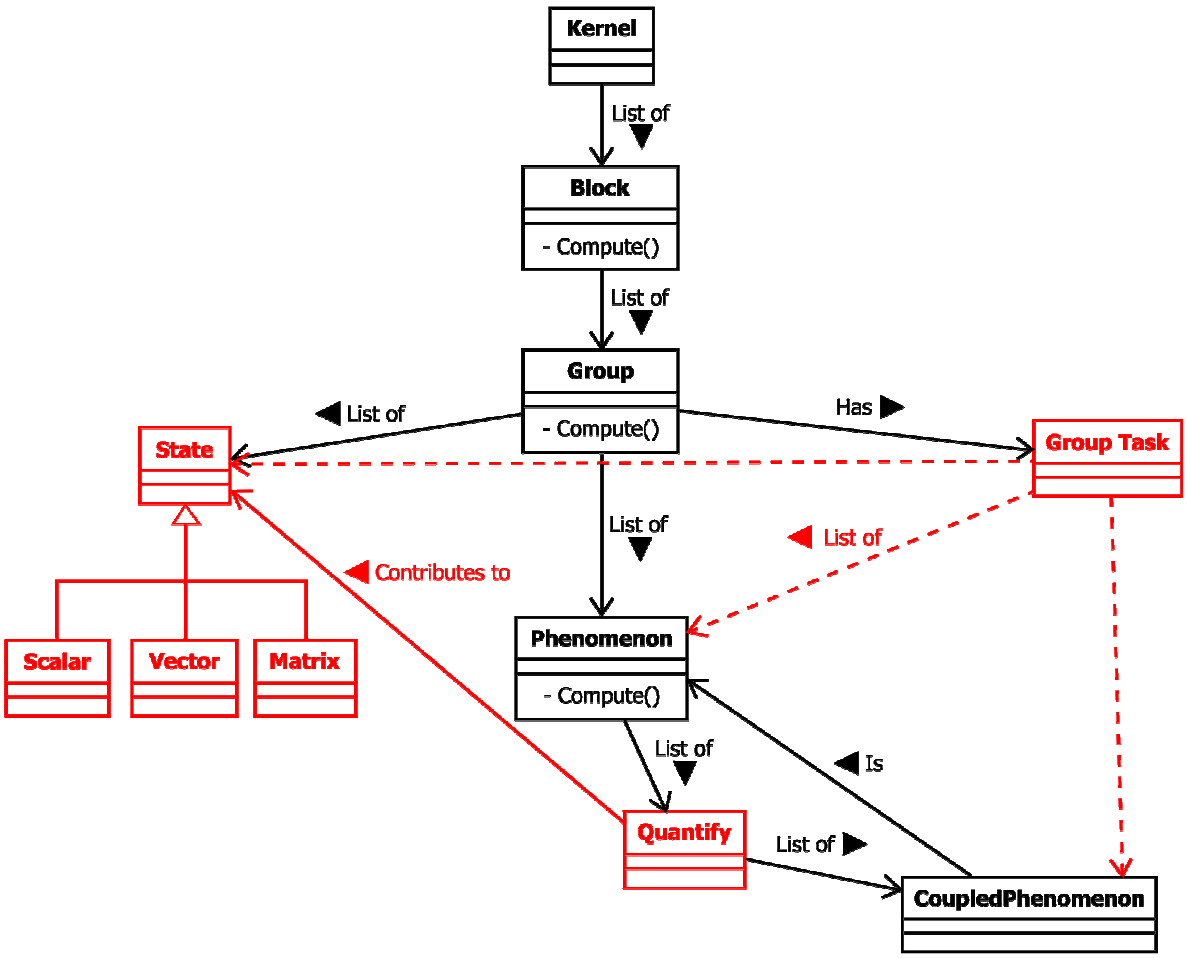


Figura 3-5 Representação com as relações de Quantia contribui para um Estado e *GroupTasks* solicita ao Fenômeno o cálculo de uma Quantia.

Para uma Quantia ser calculada, a mesma pode precisar de dados produzidos por outros Fenômenos, chamado de Fenômenos Acoplados. Quando há acoplamento entre Fenômenos, o *GroupTask*, ao solicitar o cálculo de uma Quantia a um Fenômeno, já o informa que outros Fenômenos são acoplados a ele e estes serão necessários para o cálculo. A Figura 3-6 destaca as relações referentes aos acoplamentos entre Fenômenos.

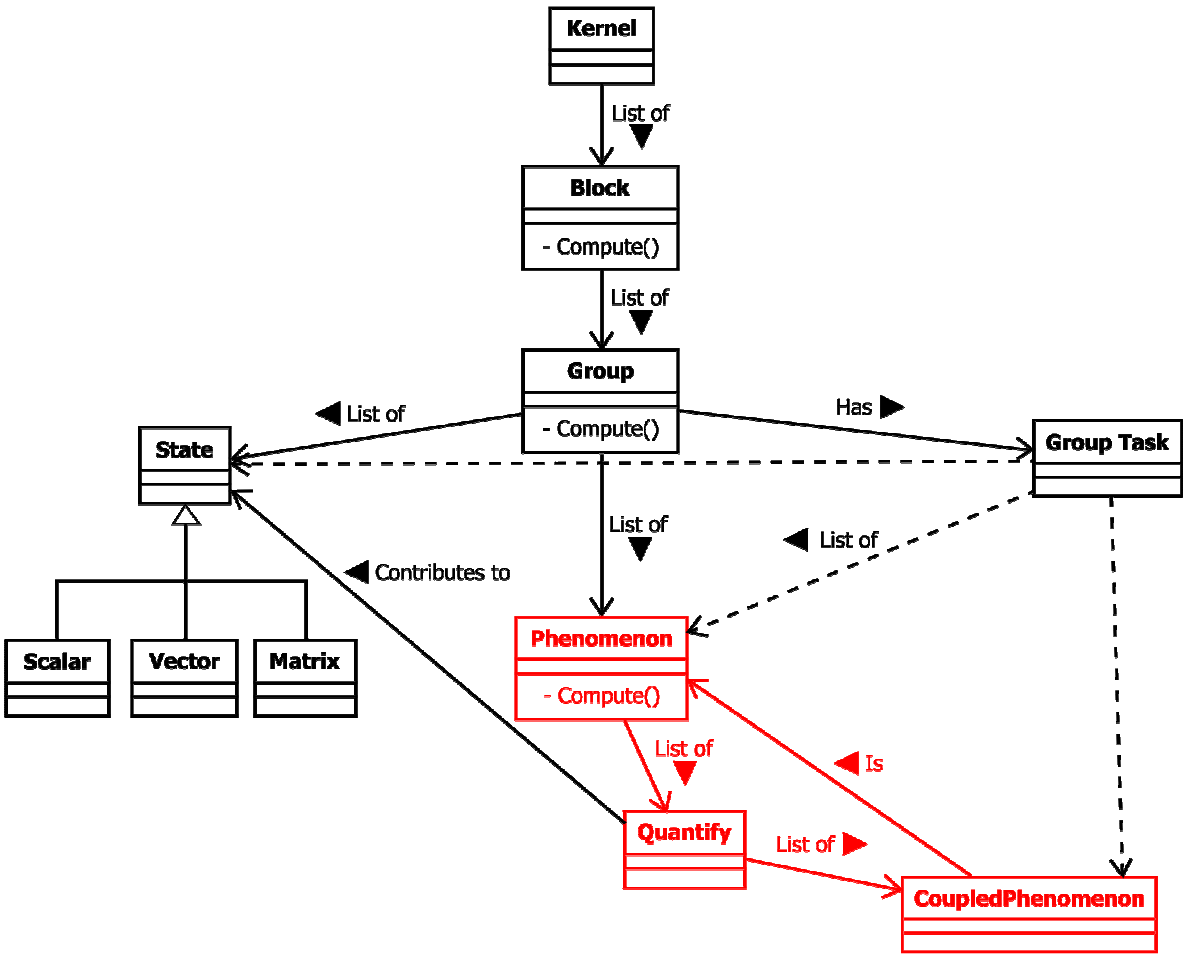


Figura 3-6 Representação com as relações de acoplamento.

Esta representação da arquitetura, em diagrama de classes, é um primeiro passo para a correta definição de uma arquitetura em redes de Petri para o ambiente MphyScaS.

Capítulo 4

Especificação Formal

O rápido avanço tecnológico requer, cada vez mais, formas mais rápidas e confiáveis de desenvolver sistemas de *software* a fim de atender aos requisitos de usuários e sistema. É, portanto, essencial a esses desenvolvedores utilizar técnicas de especificação formal que ofereçam suporte a corretude, completude, consistência, concisão e clareza [18].

É ainda de suma importância que a técnica de especificação formal possua um conjunto de conceitos bem definidos e base matemática de modo a permitir a construção e uso de ferramentas computacionais para analisar e simular especificações, bem como verificar a conformidade de implementações com as respectivas especificações.

Um dos principais desafios deparados é o de assegurar a confiabilidade de sistemas projetados. Este objetivo torna-se ainda mais evidente quando o *software* é empregado em sistemas cuja falha operacional pode ocasionar perdas humanas, econômicas ou causar degradação na prestação de serviços.

Como os simuladores gerados pelo MPhyScaS analisam problemas reais, com diversos Fenômenos atuando sobre uma geometria contínua, esse processo de simulação pode ser bastante custoso, levando horas ou dias para ser realizado [1]. Por isto, é extremamente importante, para o usuário do simulador, verificar e validar se os dados inseridos estão corretos. O modelo proposto neste capítulo representa esses dados, permitindo que análises sejam feitas a fim de auxiliar o usuário nesta tarefa.

4.1 O modelo formal

O modelo formal para o problema apresentado deve garantir um nível de abstração dos dados de entrada do simulador que permitam que dúvidas possam ser respondidas por análises feitas sobre o modelo. Isto ajudará a validar os dados de entrada através de simulações, detectando inconsistências e ambigüidades, quando houver.

Ao terminar de inserir os dados de entrada, o usuário solicitará o início da simulação, mas antes que a simulação seja iniciada o simulador criará o modelo formal proposto e fará todas as análises definidas. Se nessas análises não for encontrado nenhum erro, o simulador permitirá que a simulação seja iniciada; caso contrário, o simulador informará ao usuário sobre os erros encontrados. O modelo criado é totalmente abstrato para o usuário, ou seja, ele não sabe de sua existência, muito menos como é criado e como são feitas as análises.

A técnica de redes de Petri será utilizada para a criação do modelo. O componente do *Kernel* será representado por um lugar da rede de Petri. Cada Bloco criado também será representado por um lugar da rede de Petri. Seguindo a metodologia, cada Grupo criado será representado por um lugar. A Figura 4-1 mostra um exemplo dessa representação.

As relações entre *Kernel* e Bloco, e Bloco e Grupo são representadas por uma transição que liga os componentes. Cada relação deve ser representada por uma transição, como podem ser identificadas na Figura 4-1. Um Bloco não deve estar relacionado com mais de um Grupo através da mesma transição. Isto não pode acontecer porque posteriormente será preciso analisar se existe possibilidade de chegar um *token* a um determinado Grupo, e para isto seria necessário fazer várias verificações. Esta análise exige uma marcação inicial que possui apenas um *token* no lugar que representa o *Kernel*.

Para ilustrar utilizando o exemplo da Figura 4-1, deseja-se analisar se o Grupo0 está relacionado a algum Bloco pertencente ao *Kernel*. Para isto, precisa-se saber se o Estado M(0, 0, 0, 1, 0, 0, 0, 0) da rede pode ser alcançado. Este Estado significa a quantidade de *tokens* dos lugares *Kernel*, Bloco0, Bloco1, Grupo0, Grupo1, Grupo2, Grupo3 e Grupo4, respectivamente. Se os Blocos fossem relacionados aos Grupos por uma única transição, seria necessário verificar todas as combinações de Estados, tendo que a quantidade de *tokens* do Grupo0 seria constante e igual a 1. Ou seja, seria necessário verificar se os Estados M(0, 0, 0, 1, 1, 1, 1, 1), M(0, 0, 0, 1, 0, 1, 0, 1) ou alguma das demais possíveis combinações, com o quarto número igual a 1 (número de *tokens* que deve haver no Grupo0), pode ser alcançado. Por este motivo, o modelo é melhor representado com uma transição para cada relação existente. Este mesmo motivo serve para as transições que ligam o *Kernel* aos Blocos.

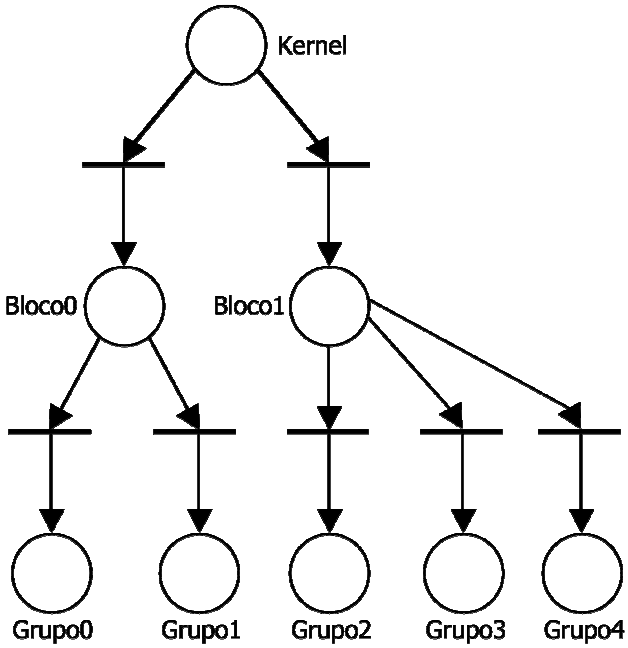


Figura 4-1 Exemplo de especificação para *Kernel*, Bloco e Grupo e suas relações.

Como visto no capítulo anterior, os Grupos possuem Fenômenos, *GroupTasks* e gerenciam Estados. Os Estados serão explicados posteriormente. Cada componente desses (*GroupTask* e Fenômeno) será representado por um lugar na rede de Petri. As relações entre Grupo e *GroupTask*, e Grupo e Fenômeno seguem a metodologia e são representados, cada uma, por uma

transição, devido ao mesmo motivo já explicado. A Figura 4-2 exemplifica uma rede de Petri, que é parte do modelo, que apresenta esses componentes e relações.

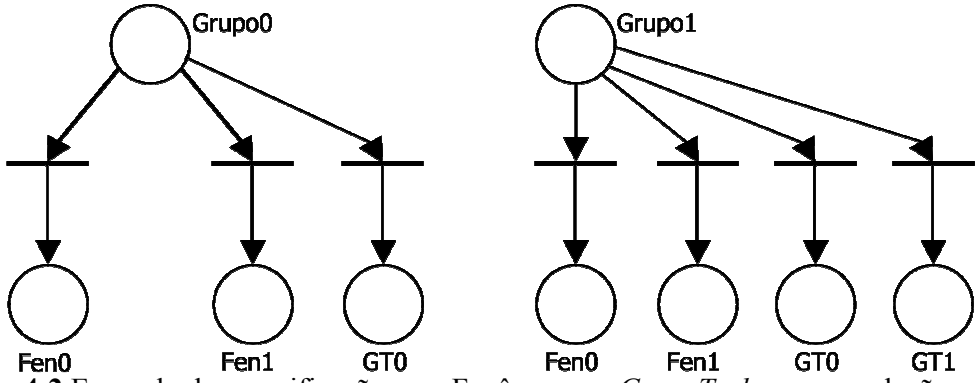


Figura 4-2 Exemplo de especificação para Fenômenos e *GroupTasks*, e suas relações com os Grupos.

Note que, na Figura 4-2, o Grupo0 possui o Fen0 e Fen1 e o Grupo1 também possui Fenômenos com os mesmos identificadores, mas isso não quer dizer que esses Fenômenos são os mesmos. Cada Grupo gerencia os identificadores de seus Fenômenos, o que significa que o Fen0 do Grupo0 não é o mesmo que o Fen0 do Grupo1. O mesmo acontece para os outros Fenômenos e para os *GroupTasks* de Grupos diferente.

Seguindo a metodologia para a criação do modelo, as Quantias e os Estados também são representados por um lugar na rede de Petri, como pode ser visto na Figura 4-3. As relações entre os Fenômenos e as Quantias, e as Quantias e os Estados são representadas por transições que ligam os componentes. Pelo motivo já explicado anteriormente, cada relação deve ser representada por uma transição. A Figura 4-3 ilustra um exemplo de rede de Petri, que é parte do modelo, com esses componentes e relações.

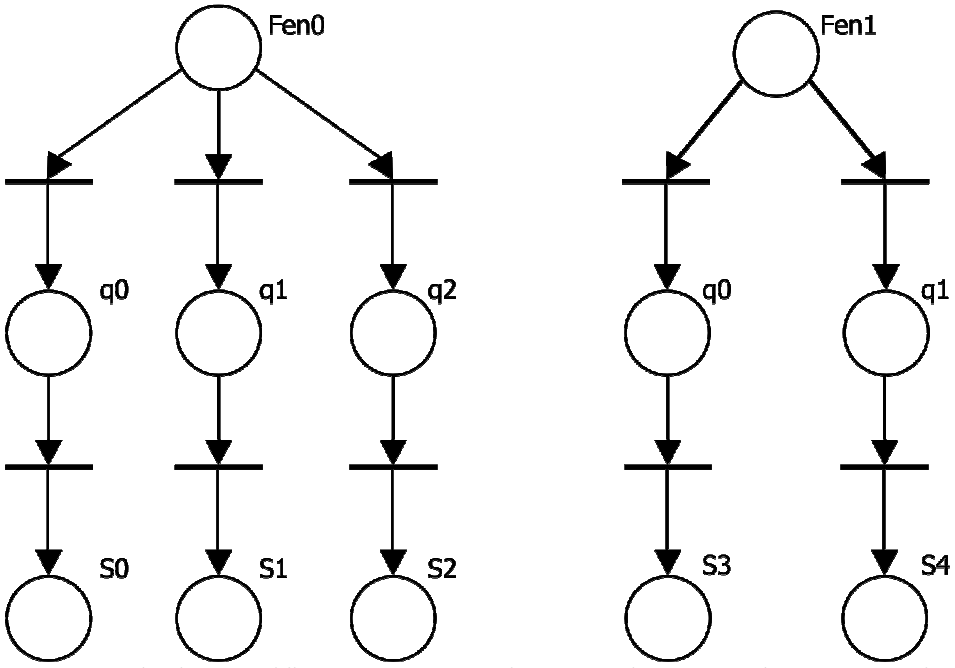


Figura 4-3 Exemplo de especificação para Quantias e Estados, suas relações entre si e com os Fenômenos.

É possível notar, na Figura 4-3, que Quantias de Fenômenos diferentes também podem possuir mesmo identificador, isto também implica em serem Quantias diferentes. Quanto aos Estados, isto não pode acontecer, mesmo sendo gerenciados por Grupos diferentes, pois as Quantias podem ser acopladas a Estados que precisam de identificação única para não serem confundidos.

Uma Quantia pode contribuir, total ou parcialmente, para mais de um Estado. Quando isso ocorre, o relacionamento entre a Quantia e os Estados aos quais ela contribui deve ser feito através de apenas uma transição. A Figura 4-4, que também é parte do modelo formal a ser criado, ilustra esse tipo de relacionamento.

Esta representação deve ser feita apenas por uma transição porque, obrigatoriamente, a Quantia contribui com os dois Estados em uma mesma ocasião. Como exemplo disso, suponha que o Estado da rede da Figura 4-4 fosse M(1, 0, 0, 0), sendo esse Estado a quantidade de *tokens* dos lugares q1, q2, S1 e S2, respectivamente. O Estado M(0, 0, 1, 0) da rede não deve ser alcançado, pois representaria a contribuição da Quantia q1 apenas para o Estado S1. Se a representação desse tipo de relação fosse como a que acontece entre Blocos e Grupos, com uma transição para cada relação, esse Estado poderia ser alcançado. Desta forma, o modelo não representaria os dados corretamente.

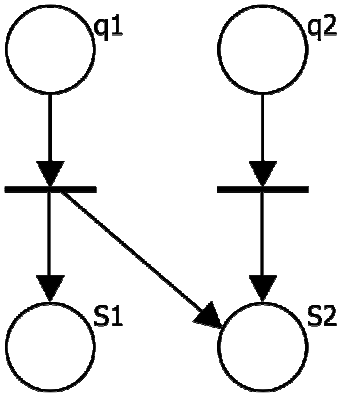


Figura 4-4 Exemplo de especificação quando uma Quantia contribui para mais de um Estado.

Quando ocorre um acoplamento entre Fenômenos, isto significa dizer que uma Quantia necessitará de um Estado, produzido por outra Quantia, para contribuir com outro Estado. Este tipo de relacionamento deve ser representado apenas por um arco que ligue o Estado necessitado pela Quantia à transição entre a Quantia e o Estado ao qual ela contribui. Isto é exemplificado na Figura 4-5, que mostra o acoplamento entre o Fen0 e o Fen1, pois uma Quantia do Fen0 necessita de um Estado produzido por uma Quantia do Fen1 para ser calculada.

A representação desse tipo de relação é feita apenas por um arco pois, ainda utilizando o exemplo da Figura 4-5, para o Estado S0 ser produzido, além de ter sido solicitado o cálculo da qauntia q0, o Estado S1 já deve ter sido produzido. A ligação que o arco faz, do Estado S1 à transição entre q0 e S0, é suficiente para representar essa dependência. Outras representações poderiam ser feitas, mas esta foi considerada por esse trabalho como a mais simples delas.

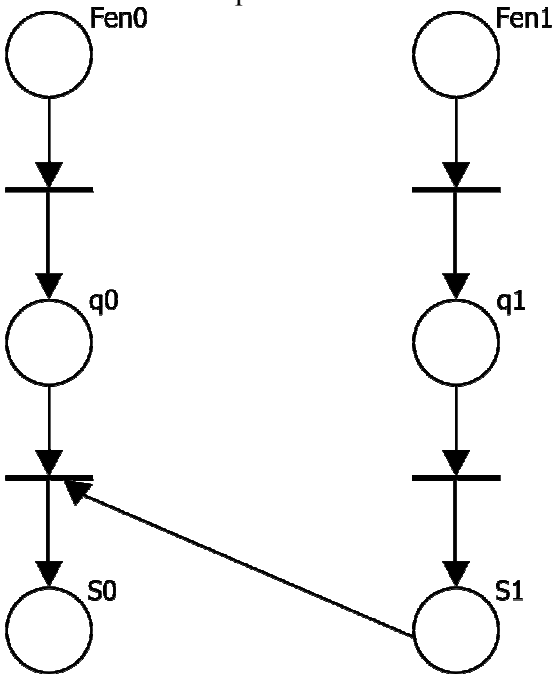


Figura 4-5 Exemplo de especificação quando acontece acoplamento.

Voltando ao relacionamento entre Grupo e Estado, tem-se que este tipo de relação deve ser representada por uma transição, para cada relação, entre estes dois componentes. A Figura 4-6 mostra este tipo de relacionamento em um exemplo.

Como visto no capítulo anterior, um *GroupTask* solicita a um Fenômeno o cálculo de uma determinada Quantia. Por questões de representação, pois não é possível diferenciar arcos em redes de Petri, não serão criadas relações diretas entre o *GroupTask* e o Fenômeno. Será criado, então, uma transição que ligará o *GroupTask* diretamente a Quantia cujo cálculo foi solicitado, como pode ser visto na Figura 4-7.

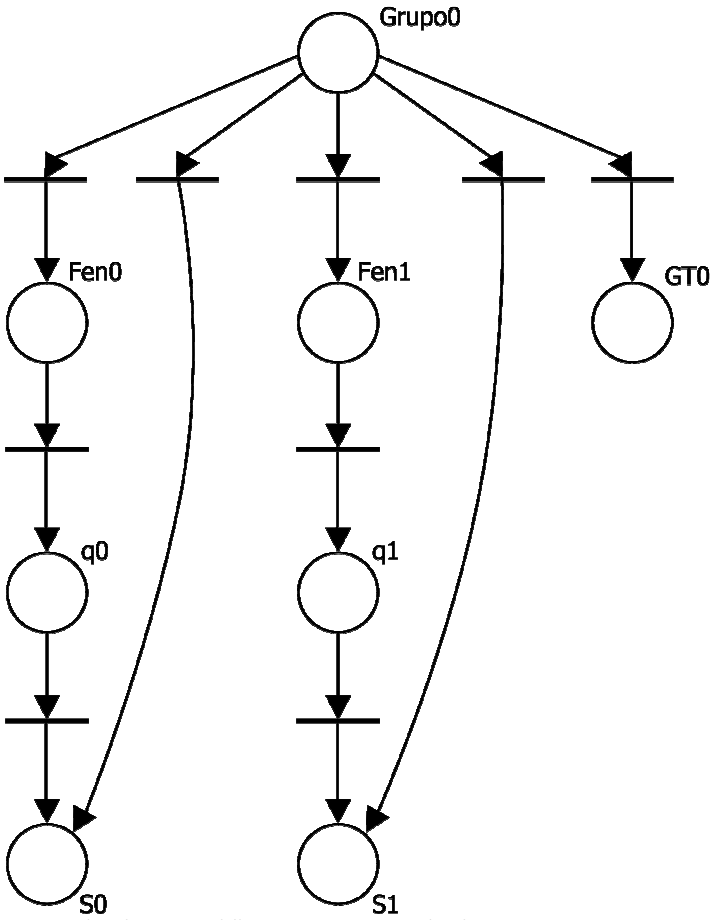


Figura 4-6 Exemplo de especificação para o relacionamento entre Grupo e Estado.

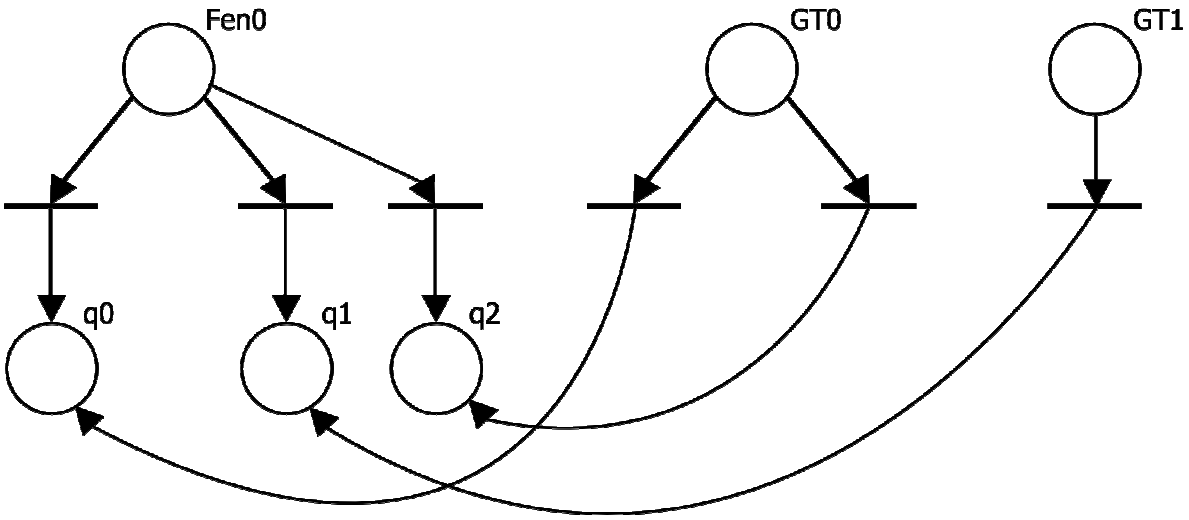


Figura 4-7 Exemplo de especificação para solicitação do cálculo de uma Quantia pelo *GroupTask*.

Para ilustrar como ficaria um modelo completo da especificação formal dos dados inseridos no simulador pelo usuário, a Figura 4-8 mostra um exemplo de todas as representações vistas neste capítulo.

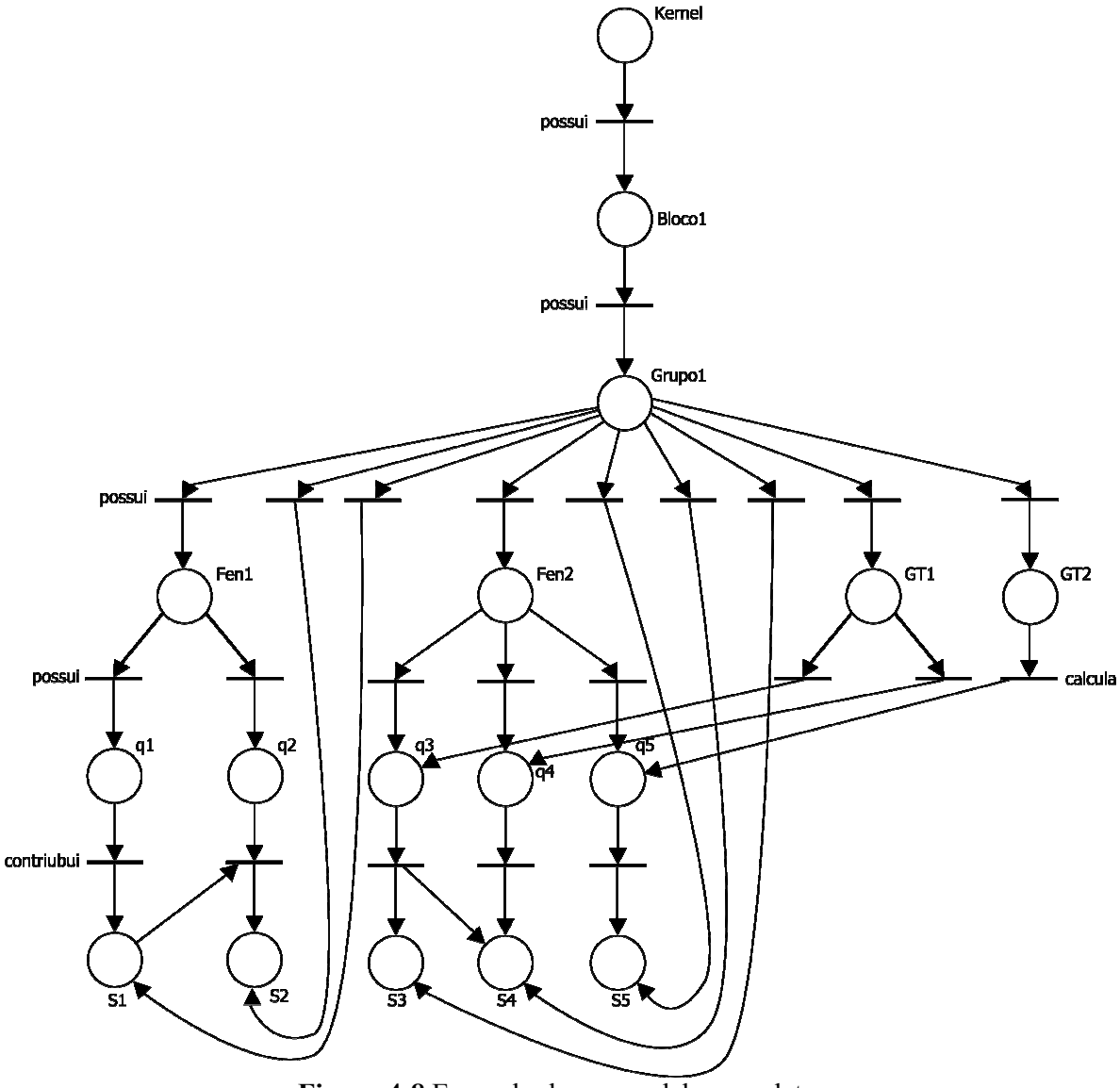


Figura 4-8 Exemplo de um modelo completo.

4.2 Análises sobre o Modelo

Antes de iniciar as análises sobre o modelo, será necessário criar três lugares na rede de Petri que servirão apenas para auxiliar tais análises. Um lugar, representado por aux0 na Figura 4-9, será ligado a todas as transições entre Grupo e *GroupTask*; um segundo lugar, representado por aux1 na Figura 4-9, será ligado a todas as transições entre Grupo e Fenômeno; e o último lugar, representado por aux2 na Figura 4-9, será ligado a todas as transições entre Grupo e Estado.

Estes lugares estão ligados apenas aos componentes que pertencem ao Grupo, pois servirão para restringir análises que sejam direcionadas a um tipo de componentes. A análise da seção 4.2.2 é um exemplo de análise direcionada aos *GroupTasks*, pois deseja-se saber se há uma seqüência de disparos, passando por um *GroupTask*, que leve a uma Quantia.

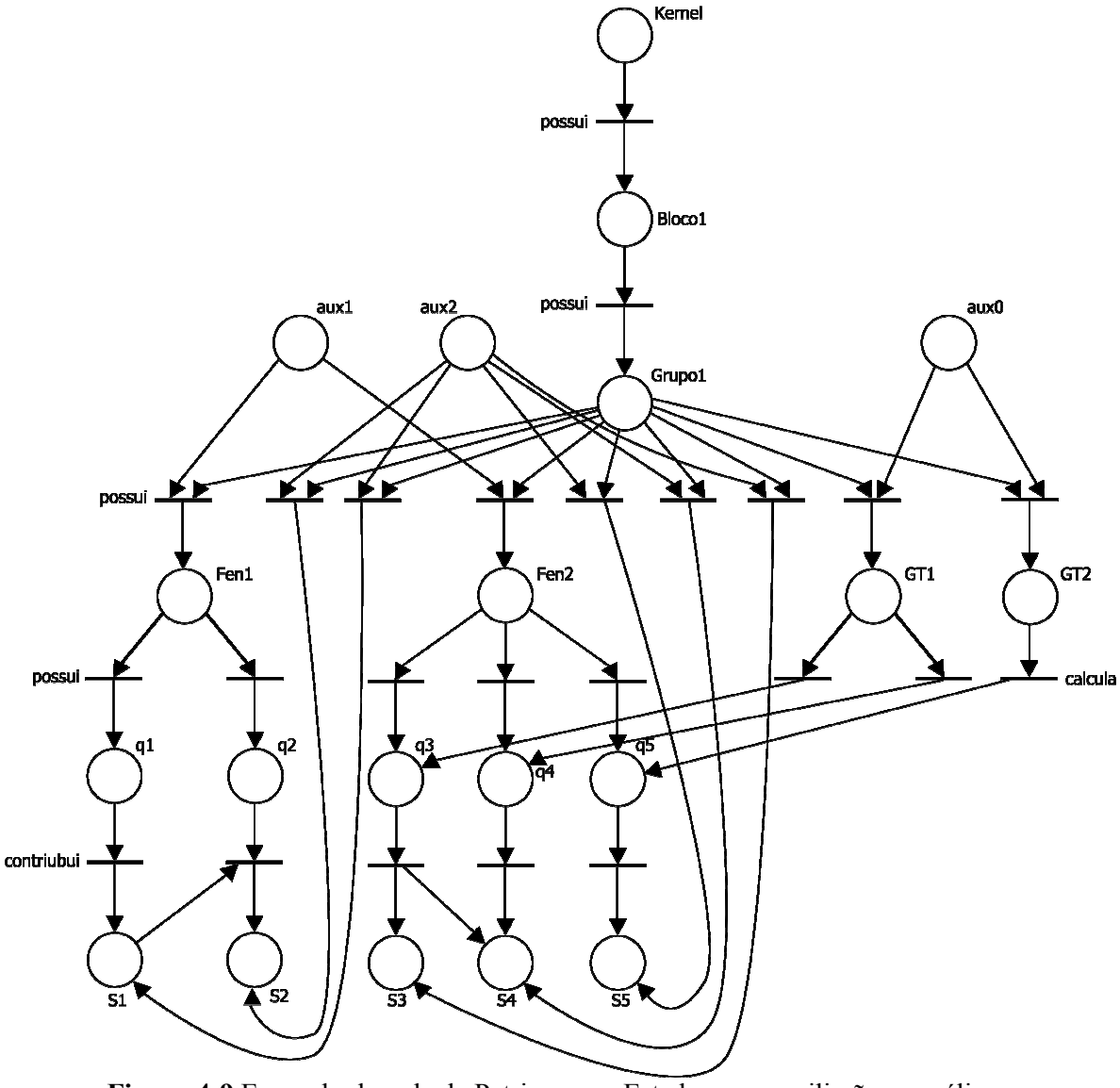


Figura 4-9 Exemplo de rede de Petri com os Estados que auxiliarão as análises.

4.2.1 Dependência entre Quantias

Esta análise tem o objetivo de verificar se uma Quantia q0 necessita de um Estado S1 para ser calculada, mas a Quantia q1 que produz esse Estado necessita do Estado S0 produzido pela por q0. Estes relacionamentos estão representados na Figura 4-10.

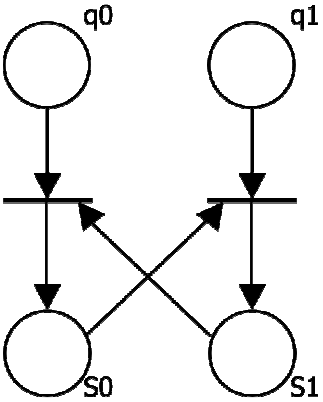


Figura 4-10 Exemplo de dependência entre Quantias.

Este tipo de relacionamento pode travar a simulação. Isto ocorre caso o simulador esteja sendo executado em uma arquitetura paralela, porque o processo de cálculo da Quantia q0 estará sendo executado por um processador e o cálculo da Quantia q1 por outro processador. Para este caso, um processador esperará o outro terminar de processar para utilizar seus resultados e vice-versa, o que acarretará em uma pausa infinita (*deadlock*).

Se o simulador estiver sendo executado em uma arquitetura seqüencial, é importante que isso seja informado ao usuário para que ele possa, caso o tenha feito inadvertidamente, modificar antes de começar a simular. Se tiver feito com um propósito, uma Quantia escolhida pelo usuário será calculada primeiro com base em um Estado com valores iniciais, atribuídos explicitamente pelo usuário.

Para realizar esta análise de forma automatizada, utilizando a ferramenta INA (*Integrated Net Analyzer* [19]), basta executar a análise que verifica a existência de deadlock. Caso exista, esta dependência entre Quantias aconteceu; caso contrário, não há dependências deste tipo.

4.2.2 Quantia ativada ou fenômeno criado não é utilizado

Esta análise tem o objetivo de informar ao usuário se ele criou algum Fenômeno que nunca será utilizado na simulação, ou ainda, se ele ativou uma Quantia de um Fenômeno e seu cálculo não será solicitado. Esta análise é importante porque pode indicar uma falha de configuração do simulador.

Para identificar isso na rede de Petri montada, é necessário verificar se há alguma Quantia cujo cálculo não é solicitado por algum *GroupTask*. Ao encontrar Quantias com essa característica, deve-se verificar se elas são necessárias para calcular o Estado de algum Fenômeno, ou seja, se elas totalizam conjunto de Quantias de um Fenômeno. Se isto ocorrer, significa dizer que este Fenômeno foi criado e não é utilizado durante a simulação.

Com a utilização do INA é possível automatizar esta análise, verificando se existe uma seqüência de disparos que gere uma marcação em uma Quantia, a partir de uma marcação inicial. Esta marcação inicial é definida pela presença de um *token* no lugar que representa o *Kernel* e um *token* no lugar auxiliar ligado às transições entre Grupo e *GroupTask*, como está representado na Figura 4-11.

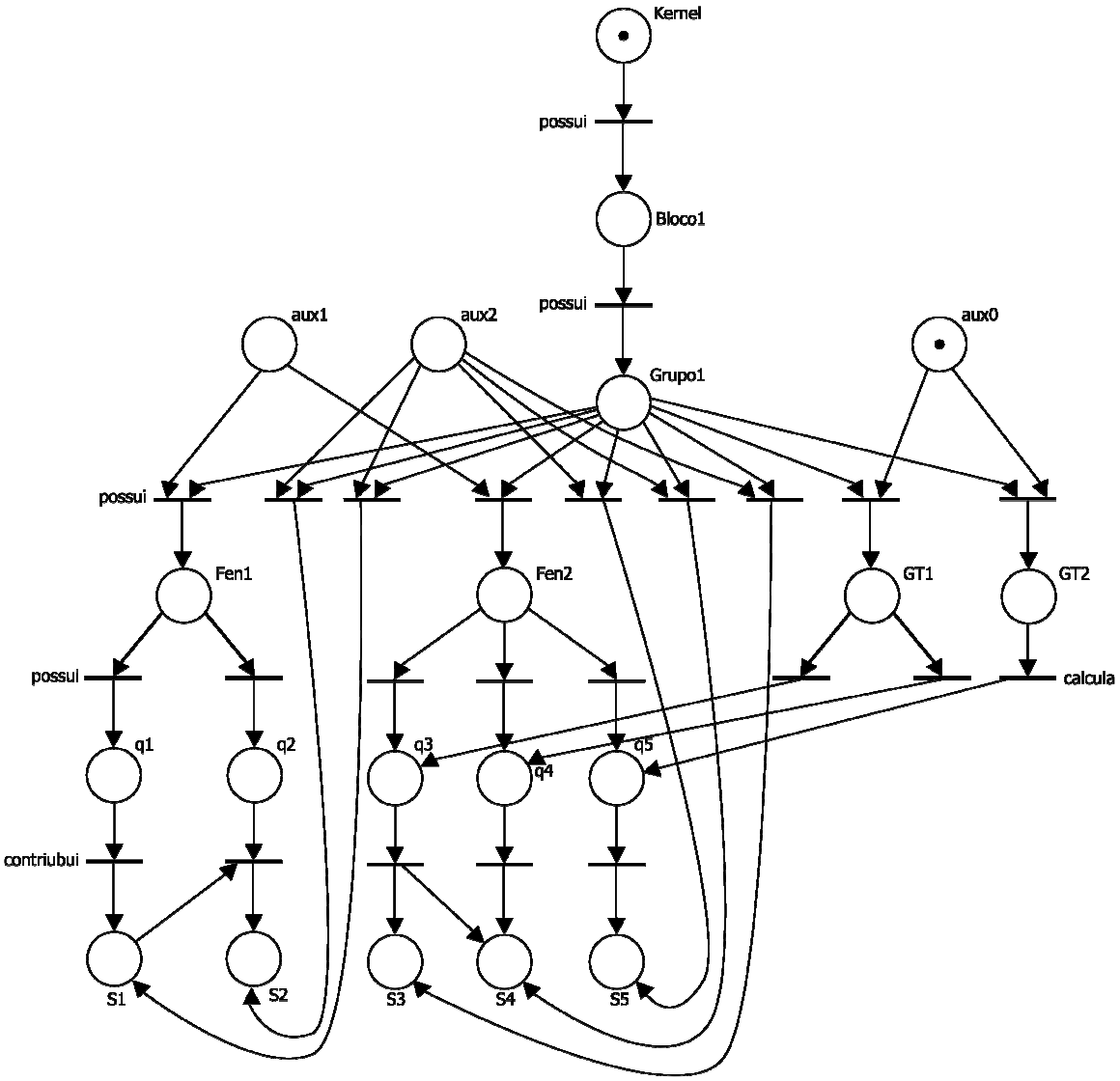


Figura 4-11 Estado inicial para a análise que verifica se existe Quantia ativa ou Fenômeno criado que não é utilizado.

Isto implicará em, quando um *token* chegar a um Grupo, este seja direcionado apenas aos *GroupTasks*, mostrando caminhos possíveis apenas se passar por um deles. A Figura 4-12 mostra o estado da rede neste momento e destaca as transições habilitadas para disparo. Desta forma, é possível verificar que os caminhos encontrados devem passar por ao menos um *GroupTask*. Esta análise deve ser realizada para todas as Quantias.

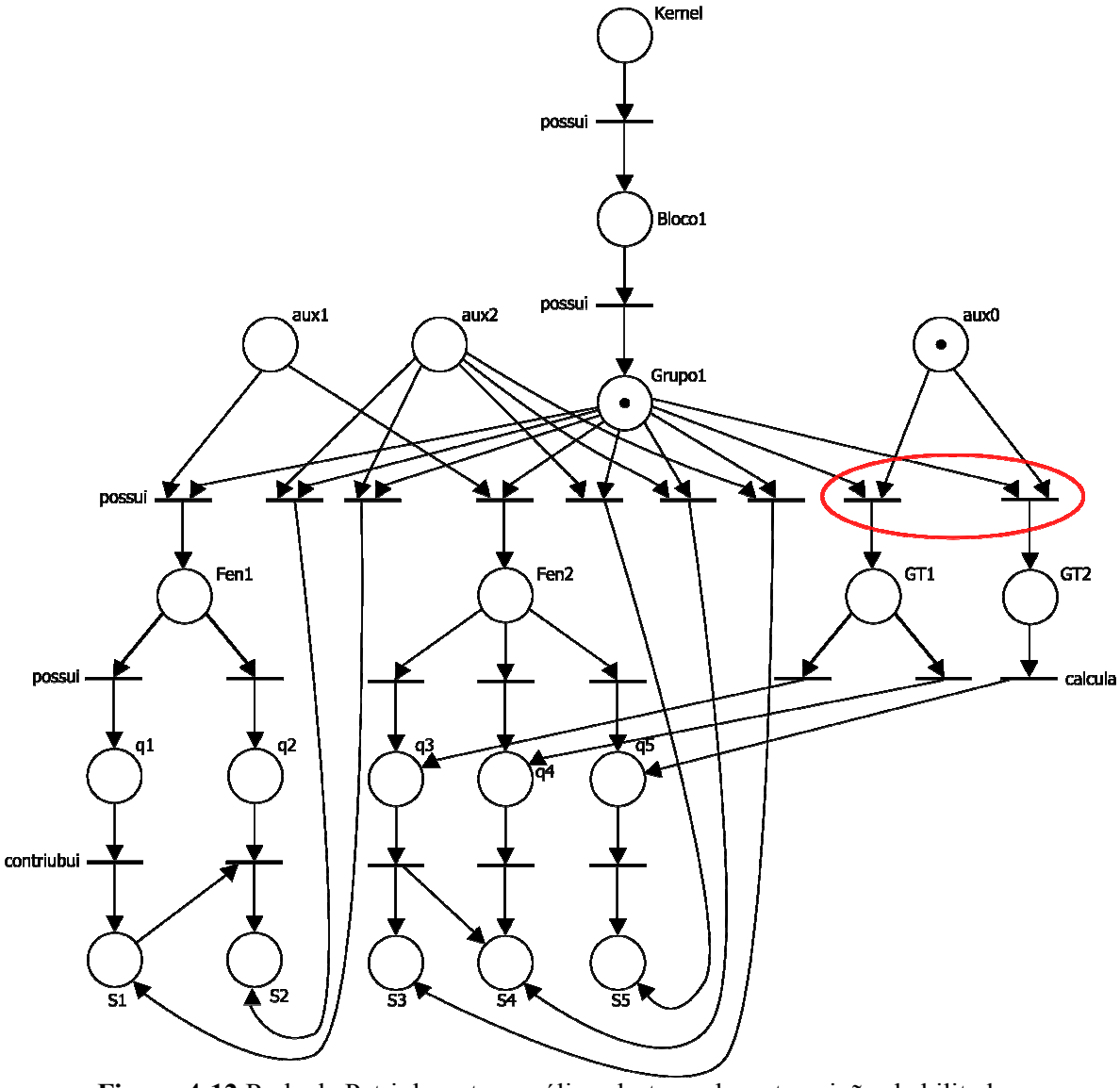


Figura 4-12 Rede de Petri durante a análise, destacando as transições habilitadas.

4.2.3 Bloco ou Grupo criado não é utilizado

Semelhante à análise anterior, esta análise tem o objetivo de informar ao usuário se ele criou um Bloco ou Grupo que nunca será utilizado na simulação. Esta situação é identificada quando há Grupos que não estão associados a algum Bloco e se há Blocos que não estão associados ao *Kernel*. Caso seja encontrado um Bloco que não está associado ao *Kernel*, os Grupos que ele possui também não serão considerados na simulação.

Para a automatização desta análise com a ferramenta INA, utiliza-se a mesma estratégia da análise anterior de verificar se existe uma seqüência de disparos que leve a aparecer um *token* a cada um dos Blocos e a cada um dos Grupos, a partir de uma marcação inicial. Esta marcação inicial seria apenas um *token* no lugar que representa o *Kernel*, pois não há necessidade de utilizar os lugares auxiliares. A Figura 4-13 mostra o estado inicial da rede para esta análise, não havendo necessidade de utilizar os estados auxiliares, pois a análise considera apenas a parte destacada da rede.

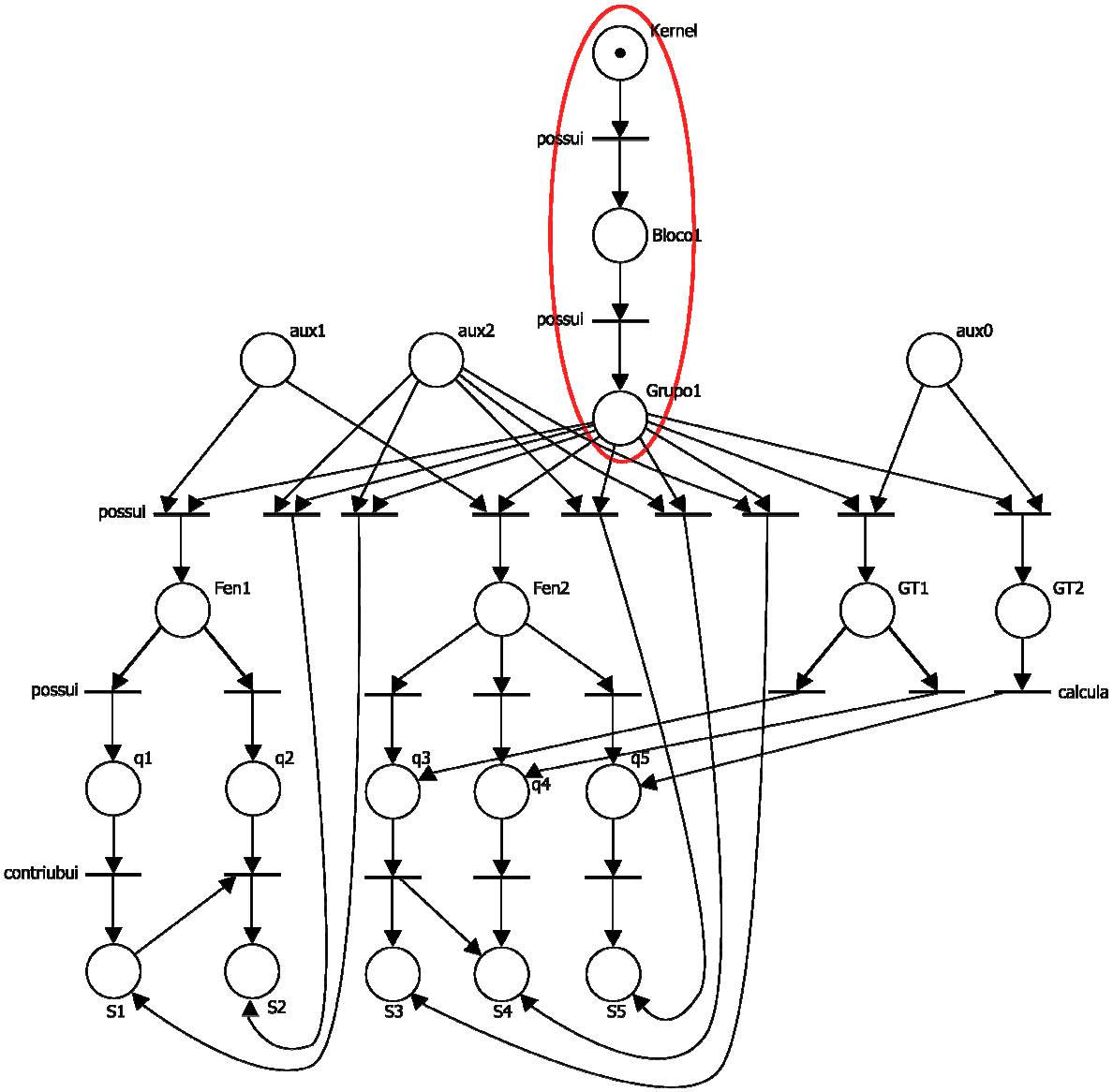


Figura 4-13 Estado inicial da rede de Petri para a análise que verifica se há algum grupo ou bloco criado que não é utilizado.

4.2.4 Estado de um Grupo não é utilizado

Esta análise tem o objetivo de verificar se um Estado de um Grupo foi criado e não será utilizado pela simulação. O usuário deve ser informado quando isto acontecer, porque o Estado pode ser criado para ser utilizado como auxiliar. Neste caso, o Estado armazena os valores de outro Estado em um determinado instante de tempo a fim de ser utilizado posteriormente.

Para realizar esta análise na ferramenta INA, é verificado se existe uma seqüência de disparos que leve a aparecer um *token* a cada um dos Estados, a partir de uma marcação inicial. Esta marcação deve ser definida por um *token* no lugar que representa o *Kernel* e um *token* no lugar auxiliar ligado às transições entre Grupo e Fenômeno, porque deseja-se verificar se o Estado não é produzido por alguma Quantia. Isto implicará em, quando um *token* chegar a um Grupo, este seja direcionado apenas aos Fenômenos, mostrando caminhos possíveis apenas se

passar por um deles. A Figura 4-14 mostra o estado inicial da rede de Petri para esta análise, destacando as transições que ficarão habilitadas quando o *token* chegar ao Grupo1.

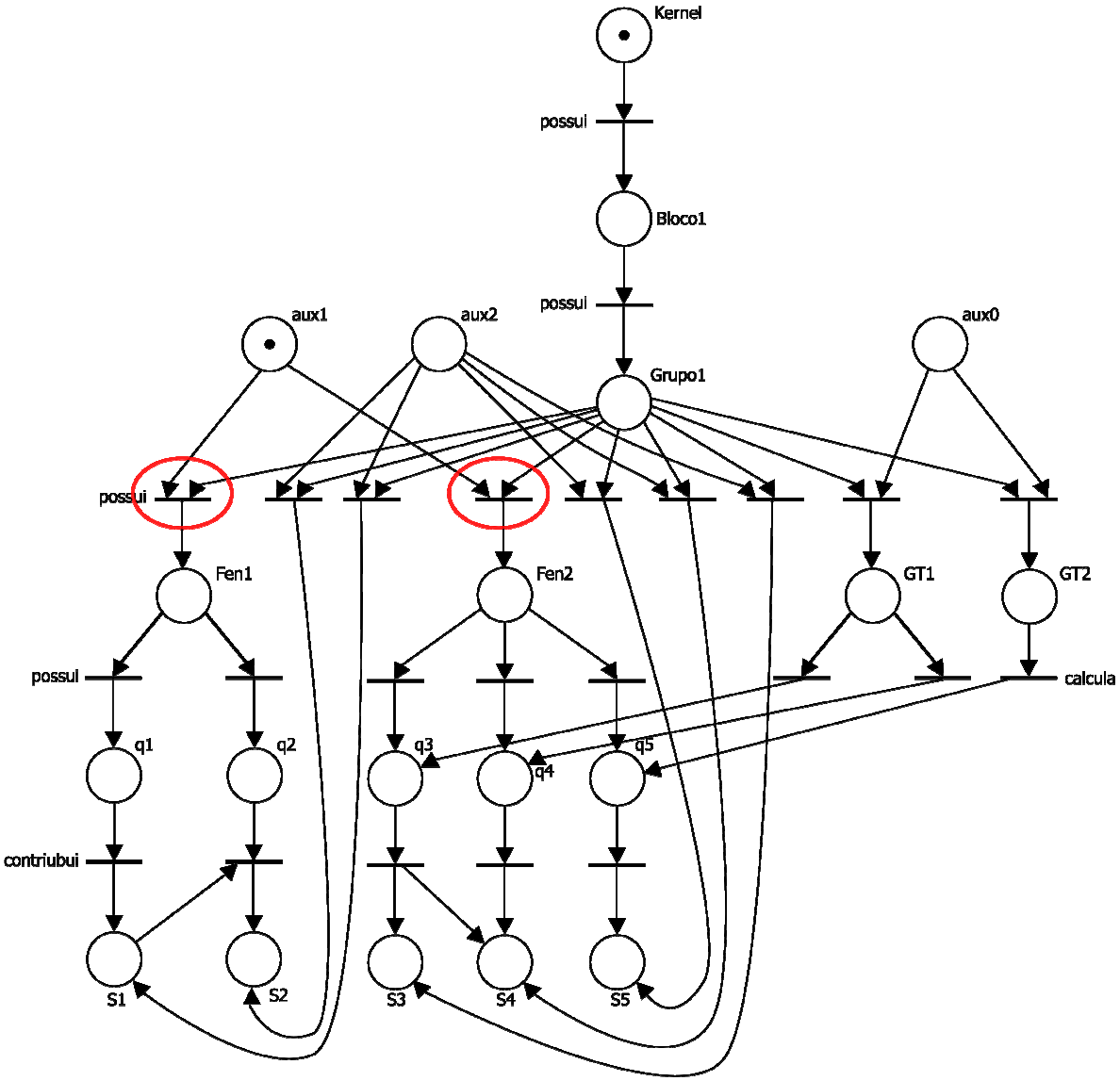


Figura 4-14 Estado inicial da rede de Petri para a análise que verifica se algum Estado criado não é utilizado, destacando as transições que poderão disparar quando o *token* chegar ao Grupo1.

Capítulo 5

Estudo de Caso

No começo da produção de petróleo, os tubos de produção utilizados eram fabricados, em sua totalidade, com aço carbono. O petróleo deixa uma camada de resíduos na parede dos tubos, que sofrem corrosão [20]. No início da extração os poços de petróleo não apresentam corrosão, mas, com o passar do tempo, estes poços tendem a produzir água em quantidade e gradualmente retira os resíduos de óleo da parede dos tubos, e assim os poços com grande produção de água tendem a ser altamente corrosivos.

Para evitar a corrosão nos tubos de petróleo, as indústrias passaram a utilizar tubos de material do tipo Liga Resistente a Corrosão (CRA – *Corrosion Resistant Alloys*). O fator negativo deste material é que possui um coeficiente de difusão baixo, possibilitando a ocorrência de danos por hidrogênio, como trincas durante o seu processo de fabricação ou durante seu uso.

O fenômeno de difusão consiste no movimento de partículas de uma área de maior concentração para uma área de menos concentração, buscando o equilíbrio. No momento em que a concentração de hidrogênio aumenta na composição do aço e este hidrogênio se aglomera em um ponto, ocorre trincas no material.

A ocorrência de uma trinca durante o processo de fabricação é algo custoso, mas é possível ser descoberto antes de colocar o tubo defeituoso para o uso. Já a ocorrência de um trincamento durante o uso pode ocasionar o rompimento do tubo antes da devida manutenção, já que estes tubos geralmente são subterrâneos, e pode vir a causar enormes prejuízos financeiros e ao meio ambiente.

Para tentar prever o momento de realizar a manutenção em tubos de petróleo foi gerado um simulador nos moldes do MPhyScaS, para analisar o desgaste de tubos a ataques devido a presença de hidrogênio. O simulador gerado resolve problemas de difusão acoplado com elasticidade.

5.1 A Simulação

Para simular o problema de corrosão nos tubos de petróleo é necessário inicialmente definir a geometria que sofrerá a ação dos Fenômenos. A geometria do problema pode ser vista na Figura 5-1, onde R0 = 1.0 e R1 = 2.0. Também é preciso que sejam informadas as condições de contorno para o problema. Desta forma, poderá haver Quantias definidas no domínio, ou seja, na geometria, ou no contorno.

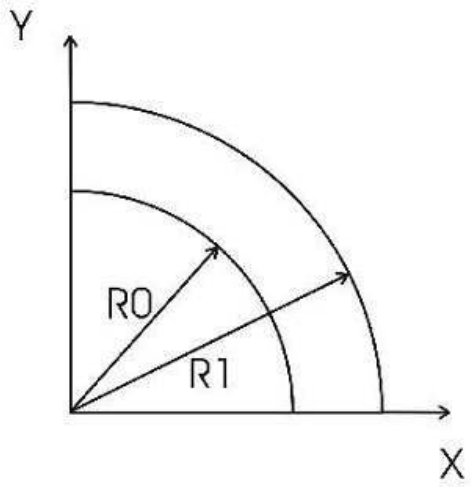


Figura 5-1 Geometria do problema.

Com o objetivo tornar a simulação o mais real possível, foram necessários cinco Fenômenos atuando sobre a geometria e/ou no contorno. Cada Fenômeno é responsável por resolver uma parte do sistema e, por isso, atuam de maneiras diferentes. Os Fenômenos utilizados são:

1. Deslocamento: possui Quantias que atuam no domínio e no contorno. É responsável por calcular a rigidez e a força elástica sofrida pelo material. Também gerencia as condições de contorno.
2. Deformação: possui apenas uma Quantia que atua no domínio. Calcula a deformação sofrida pelo material.
3. Elasticidade: possui apenas uma Quantia que atua no domínio. Esta Quantia monta a matriz de elasticidade.
4. Tensão: possui apenas Quantias que atuam no domínio. É responsável por calcular as tensões nos elementos, definir o traço das tensões nodais e calcular essas tensões.
5. Difusão: possui Quantias que atuam no domínio e no contorno. É responsável por calcular o termo de fonte do soluto, que, para o problema estudado, é o hidrogênio, além de gerenciar as condições do contorno.

Foram feitas duas simulações para este problema. A primeira delas variou a concentração do soluto (hidrogênio) no tempo. A segunda simulação variou esta concentração no tempo e no espaço. Os resultados para as duas simulações foram diferentes, como podem ser vistos nas Figura 5-2 e Figura 5-3, respectivamente.

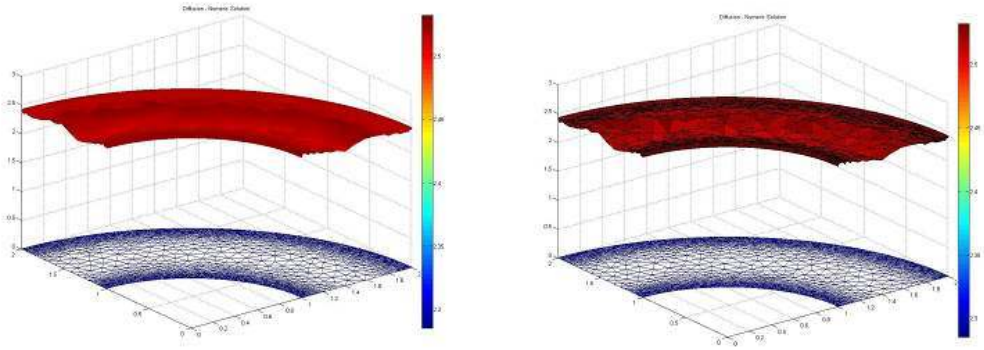


Figura 5-2 Resultado da simulação quando a concentração do soluto varia no tempo.

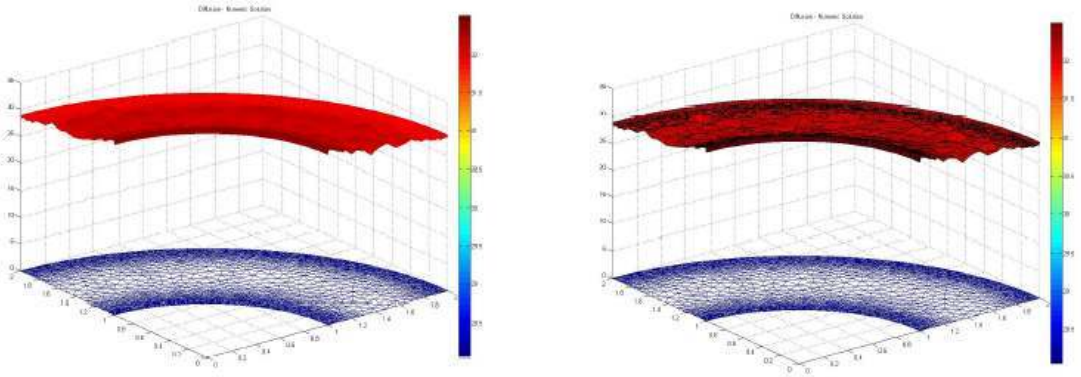


Figura 5-3 Resultado da simulação quando a concentração do soluto varia no tempo e no espaço.

É possível verificar, através da comparação da Figura 5-2 e da Figura 5-3, que o material dos tubos de petróleo sofreu maior deformação na segunda simulação. A escala de cores mostra isso, pois é uma escala aproximada da região da geometria. Quanto mais profunda for essa deformação, maior é a variação de cores, seguindo a variação da escala.

5.2 O Modelo Criado

Para que fosse possível fazer verificações sobre os dados de entrada inseridos no simulador para estas simulações, foi necessário criar o modelo formal seguindo a metodologia vista nesta monografia. Os componentes e suas relações foram fornecidos por um especialista na área da simulação.

Como descrito na seção anterior, participaram da simulação cinco Fenômenos. Estes podem ser encontrados na Figura 5-4. Cada Fenômeno pertence a um Grupo diferente, tendo um total de cinco Grupos. Todos os Grupos pertencem ao mesmo Bloco, havendo apenas um relacionado ao *Kernel*. Essas relações também podem ser identificadas na Figura 5-4.

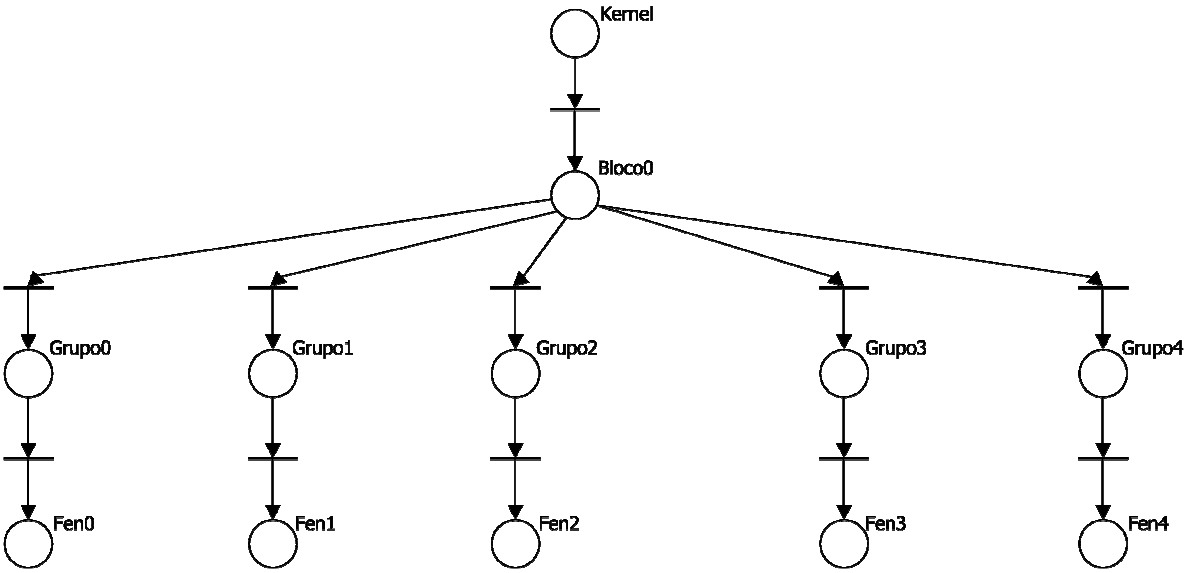


Figura 5-4 Parte do modelo que possui as relações entre *Kernel*, Bloco, Grupo e Fenômeno.

Cada Fenômenos possui uma quantidade de Quantias diferentes, cada uma delas tem uma funcionalidade específica. Cada Quantia contribui, total ou parcialmente, para algum Estado. A representação das Quantias e seus relacionamentos com os Fenômenos, bem como a representação dos Estados e seus relacionamentos com as Quantias estão representados nas Figura 5-5, Figura 5-6 e Figura 5-7.

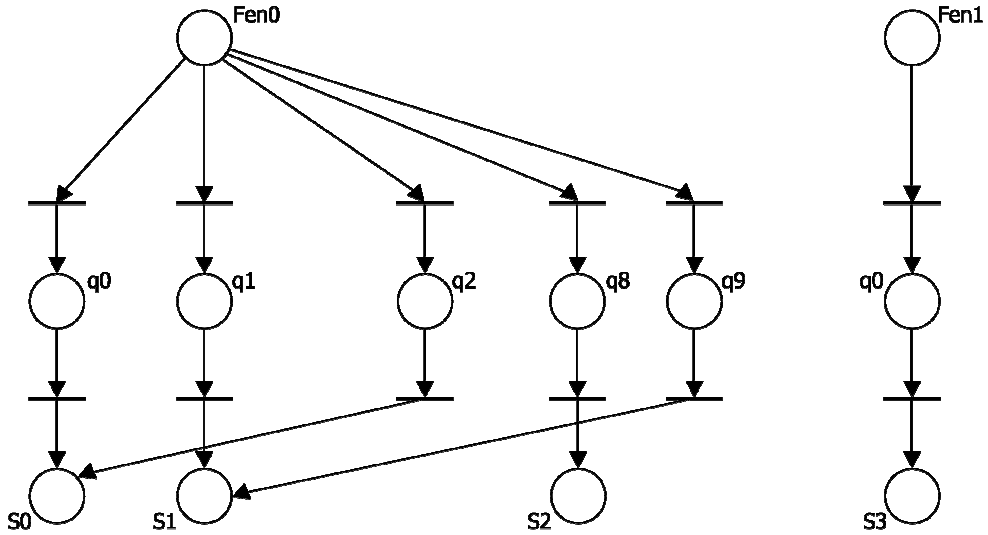


Figura 5-5 Parte I do modelo que possui as relações entre Fenômeno, Quantia e Estado.

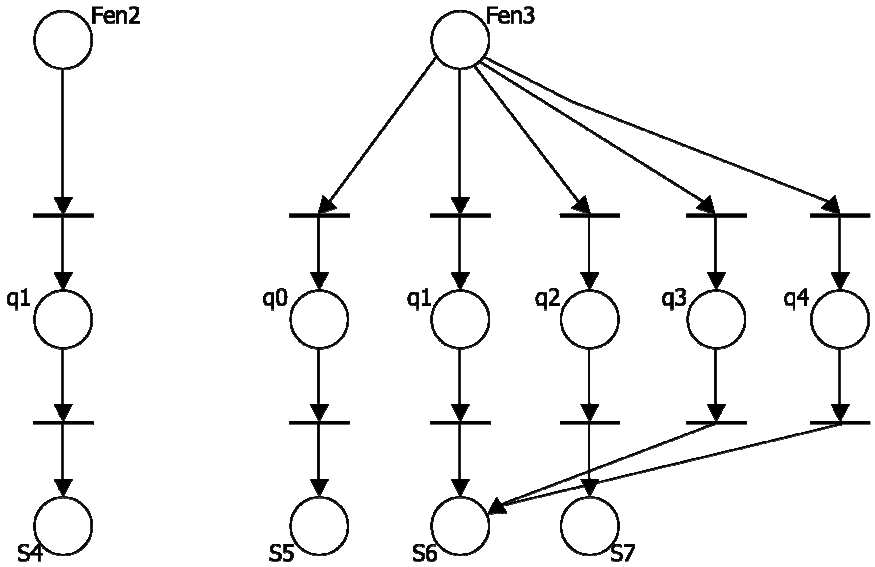


Figura 5-6 Parte II do modelo que possui as relações entre Fenômeno, Quantia e Estado.

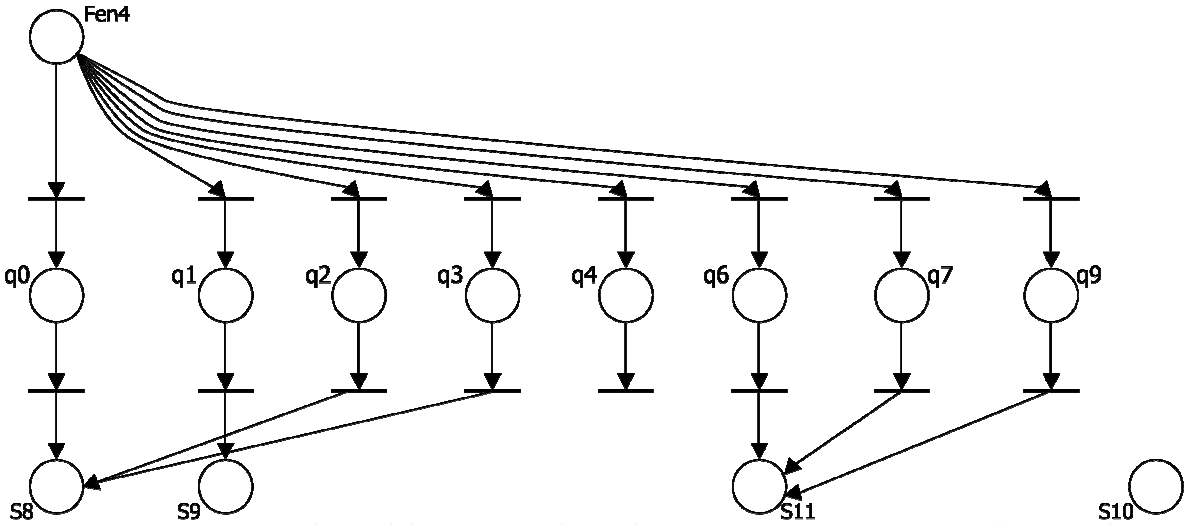


Figura 5-7 Parte III do modelo que possui as relações entre Fenômeno, Quantia e Estado.

A representação dos *GroupTasks* definidos pelo especialista pode ser encontrada nas Figura 5-8, Figura 5-9 e Figura 5-10. Nela também é possível visualizar os relacionamentos entre *GroupTasks* e Quantias, que representam a relação de solicitação, por parte do *GroupTask*, a um Fenômeno o cálculo de uma determinada Quantia.

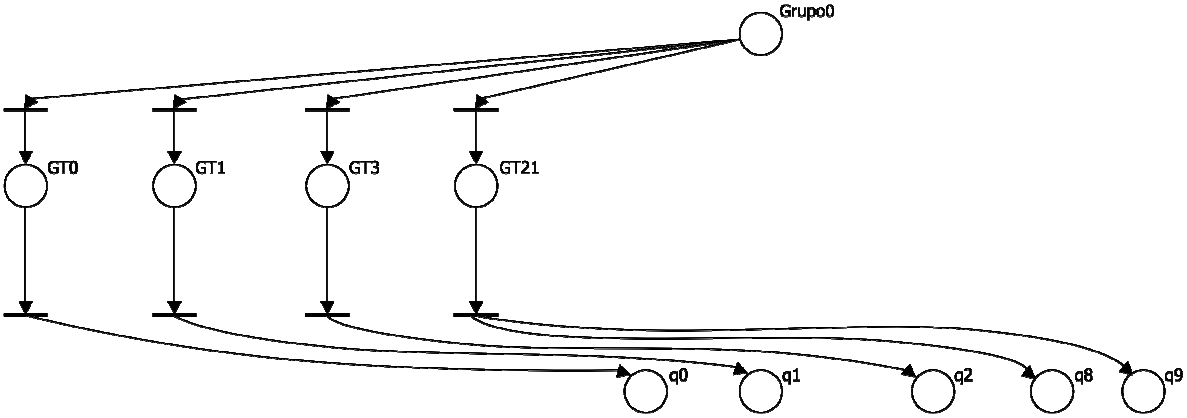


Figura 5-8 Parte I do modelo que possui as relações entre Grupo, *GroupTask* e Quantia.

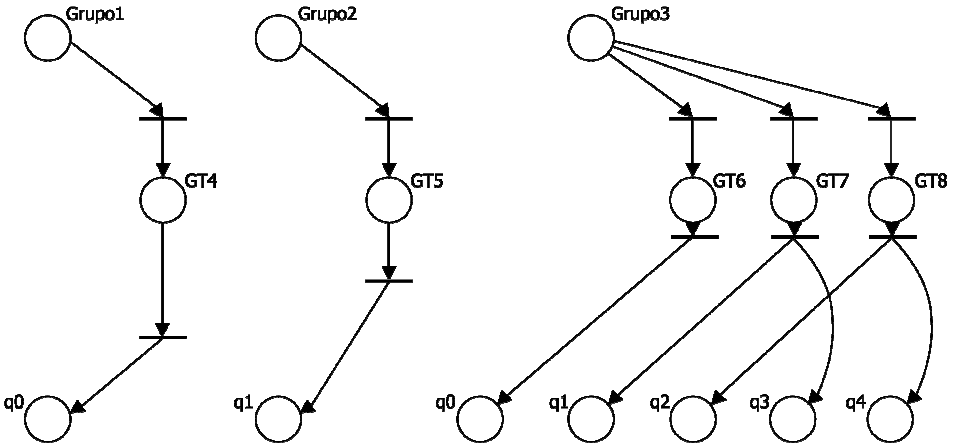


Figura 5-9 Parte II do modelo que possui as relações entre Grupo, *GroupTask* e Quantia.

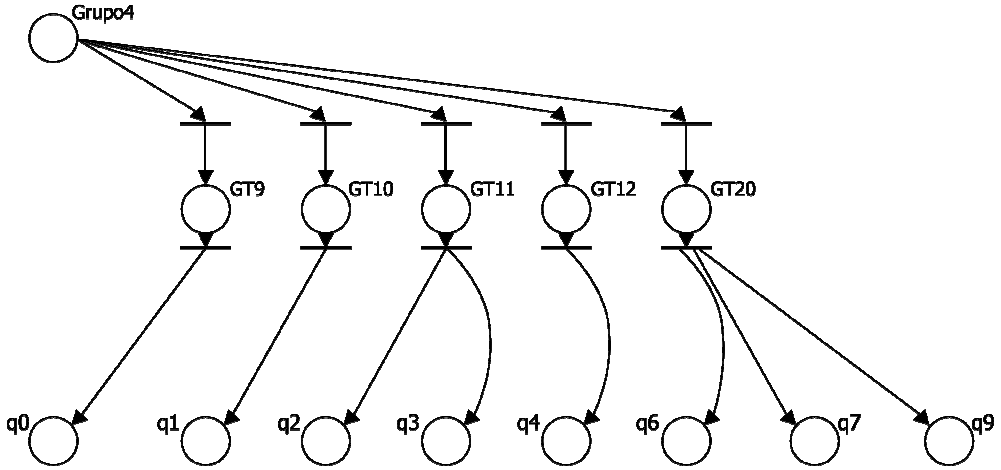


Figura 5-10 Parte III do modelo que possui as relações entre Grupo, *GroupTask* e Quantia.

Os modelos criados para as duas simulações citadas são iguais. Isto aconteceu porque a variação das simulações se encontra na mudança de valores dos atributos dos Fenômenos. Algumas constantes da simulação também tiveram seus valores alterados para satisfazer condições específicas de cada uma delas.

Devido à complexidade no entendimento da rede de Petri completa, foram apresentados apenas alguns componentes e relações pertencentes aos dados inseridos no simulador pelo especialista. A rede de Petri completa pode ser encontrada no Apêndice A.

5.3 As Análises

Quanto à análise de dependência entre Quantias, foi encontrado que a Quantia q4 do Fenômeno Fen4 necessita do Estado S9 para contribuir com o Estado S8. A Quantia q1 do Fenômeno Fen4, por sua vez, necessita do Estado S8 para produzir o Estado S9. Isto ocasionaria uma dependência mútua das Quantias.

No caso das simulações realizadas, isso não ocasiona um erro e foi feito propositalmente. Isto aconteceu porque foi desejado que a Quantia q1 do Fenômeno Fen4 produzisse o Estado S9 com o valor do Estado S8. Este último é produzido pela contribuição de mais de uma Quantia, incluindo a Quantia q4.

Por serem simulações complexas, o especialista não utilizou alguns elementos definidos como dados de entrada para estas simulações. Com o auxilio do modelo esses elementos foram identificados facilmente, permitindo que o especialista alterasse os dados de entrada para que os elementos pudessem ser utilizados. A rede de Petri mostrada no Apêndice A representa os dados de entrada após as modificações feitas pelo especialista.

Verificou-se ainda que não havia Quantia contribuindo para a produção do Estado S10, que é necessário para calcular o valor da Quantia q1. Consultando o especialista, identificou-se que este é um Estado que, em seu algoritmo de simulação, é empregado para guardar o valor de outro Estado. Desta maneira, o Estado S10 estaria sendo utilizado pelo simulador para representar um instante anterior de outro Estado do sistema.

Seria difícil, para o especialista, identificar esses casos sem a ajuda do modelo. Este proporcionou ao especialista verificar que haviam erros cometidos por ele mesmo, embora o próprio especialista não os tenha detectado.

Capítulo 6

Conclusões e Trabalhos Futuros

Com o rápido avanço da tecnologia, um importante desafio no desenvolvimento de sistemas é assegurar a confiabilidade de sistemas projetados. Isto se torna mais evidente quando estes sistemas são custosos e podem acarretar grandes riscos, inclusive riscos à vida humana, podendo levar à morte.

Para resolver este problema, é necessário que esses sistemas sejam especificados formalmente por uma técnica que dê suporte a analisar questões como sua corretude, completude, consistência, concisão e clareza [18].

O desenvolvimento deste trabalho resultou, principalmente, na definição de uma especificação formal em redes de Petri que representa, de forma abstrata, os dados de entrada para uma simulação. O modelo foi criado especificamente para os dados inseridos em simuladores criados pelo MPhyScaS.

Esta especificação é importante, pois permite ao usuário verificar e validar sua simulação antes de iniciá-la, justificando o prosseguimento para a etapa de simulação e garantindo que erros não aconteçam nesta fase.

6.1 Contribuições

As principais contribuições apresentadas nesta monografia foram:

1. A definição de um modelo que busca representar, de forma fiel, a maioria dos componentes encontrados nos dados que podem ser inseridos como entrada para uma simulação, suas interações e dependências.
2. A criação de uma especificação formal que represente, de forma abstrata, os dados de entrada inseridos pelo usuário. O modelo e sua metodologia, para serem criados, foram baseados na arquitetura de software. Estes foram definidos de forma a facilitar as análises que serão realizadas.
3. Foram definidas algumas análises que podem ser feitas sobre o modelo a fim de dar informações ao usuário sobre possíveis erros nos dados inseridos por ele. Desta forma, o usuário pode verificar e validar esses dados.

6.2 Trabalhos Futuros

O modelo formal definido nesta monografia ainda não considera todos os componentes encontrados nos dados que podem ser inseridos como entrada dos simuladores criados pelo MPhyScaS. É sugerido que, posteriormente, o modelo seja expandido de forma a tratar todos eles. Como exemplo desses componentes, podem ser citados a geometria, a malha geométrica e operações realizadas no algoritmo do *Kernel* e nos algoritmos de resolução dos Blocos. Este último componente evitaria que Estados auxiliares, como o encontrado no estudo de caso, pudessem ser confundidos com Estados não utilizados.

A proposição de outras análises que envolvessem os componentes já definidos e componentes inseridos após este trabalho devem ser realizadas. Isto permitirá ao usuário fazer a verificação e validação dos dados quanto a outros tipos de erros.

A automatização da criação de modelos seguindo a metodologia do Capítulo 3 pode ser desenvolvida a fim de diminuir o tempo gasto para esta atividade.

Por fim, outras partes ou sistemas do MPhyScaS podem ser especificados formalmente com o objetivo de verificar a corretude e confiabilidade dos mesmos. Para isto, a arquitetura de software definida deve ser expandida, ou mesmo, outra arquitetura de software pode ser definida para esta parte ou sistema.

É de suma importância que, para um projeto complexo como o MPhyScaS, todo o sistema seja verificado e validado junto a todas as partes interessadas. Isto permitirá uma melhor comunicação entre as partes, permitirá que decisões sejam tomadas antes de começar a implementação do sistema e aumenta o reuso de software.

Bibliografia

[1] WILKINS, Mark L. *Computer Simulation of Dynamic Phenomena.* First Edition. London: Springer, 1999.

[2] BASS, Len; CLEMENTS, Paul; KAZMAN, Rick. *Software Architecture in Practice.* Second Edition. Addison Wesley, 1998.

[3] *O que é Simulação.* Disponível em: <http://www.paragon.com.br/conteudo/default.asp?ID=28>. Acesso em: 16 de junho de 2007.

[4] SANTOS, F. C. G., BARBOSA, J. M. A., BRITO Jr., E. R. de, SILVA, J. M. B. *Transfer and sharing of data between coupled multi-physics phenomena during simulations: the Phenomenon Computational Pattern.* In: World Conference on Computer Science – International Conference on Modeling, Simulation & Visualization Methods, 2006, Las Vegas, pages 49-55.

[5] SANTOS, F. C. G., BARBOSA, J. M. A., BRITO Jr., E. R. de, SILVA, J. M. B. *Dealing with Coupled Phenomena in the Finite Element Method.* In: XXVII CILAMCE – Iberian Latin American Congress of Computational Methods in Engineering, 2006, Bélem.

[6] BOSCH, Jan. *Software Reuse: Methods, Techniques, and Tools.* In: 8th International Conference - ICSR, 2004, Madrid, Spain.

[7] HEINEMAN, George T., et al. *Component-Based Software Engineering.* In: 8th International Symposium – CBSE, 2005, St. Louis, USA.

[8] LOGAN, Daryl L. *A First Course in the Finite Element Method.* Third Edition. Califórnia: Brooks/Cole, 2002.

[9] CRUZ, Maria L. P. M. *Conceptualisation of an Environment for the Development os Simulators based on the Finite Element Method.* 2004. 181 f. Tese de Doutorado em Ciência da Computação, Universidade Federal de Pernambuco, Pernambuco.

[10] HEUSER, C. A. *Modelagem Conceitual de Sistemas: Redes de Petri.* In: V EBAI, 1990.

[11] MACIEL, Paulo R. M.; LINS, Rafael D.; CUNHA, Paulo R. F. *Introdução às Redes de Petri e Aplicações.* X Escola de Computação, Campinas – SP, 1996.

[12] PENHA, Dulcinéia O. de; FREITAS, Henrique C. de; MARTINS, Carlos A. P. da S. *Modelagem de Sistemas Computacionais usando Redes de Petri: Aplicação em Projeto, Análise e Avaliação.* In: Escola Regional de Informática Rio de Janeiro – Espírito Santo. 2004.

[13] DESEL, Jörg; ESPARZA, Javier. *Free Choice Petri Nets.* Pbk Version. Cambridge: Cambridge University Press, 2005.

[14] WANG, Jiacun. *Timed Petri Nets: Theory and Application.* 1 edition. London: Springer, 2002.

[15] JENSEN, Kurt. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use.* Second Edition. London: Springer, 1996.

[16] HAAS, Peter J. *Stochastic Petri Nets: Modelling, Stability, Simulation.* London: Springer, 2002.

[17] MARSAN, M. A., et al. *Modelling with Generalized Stochastic Petri Nets.* New York: Wiley, 1995.

[18] FILHO, Antonio M. da S. *O Papel da Especificação Formal no Desenvolvimento de Sistemas de Software.* In: Revista Espaço Acadêmico, 2004.

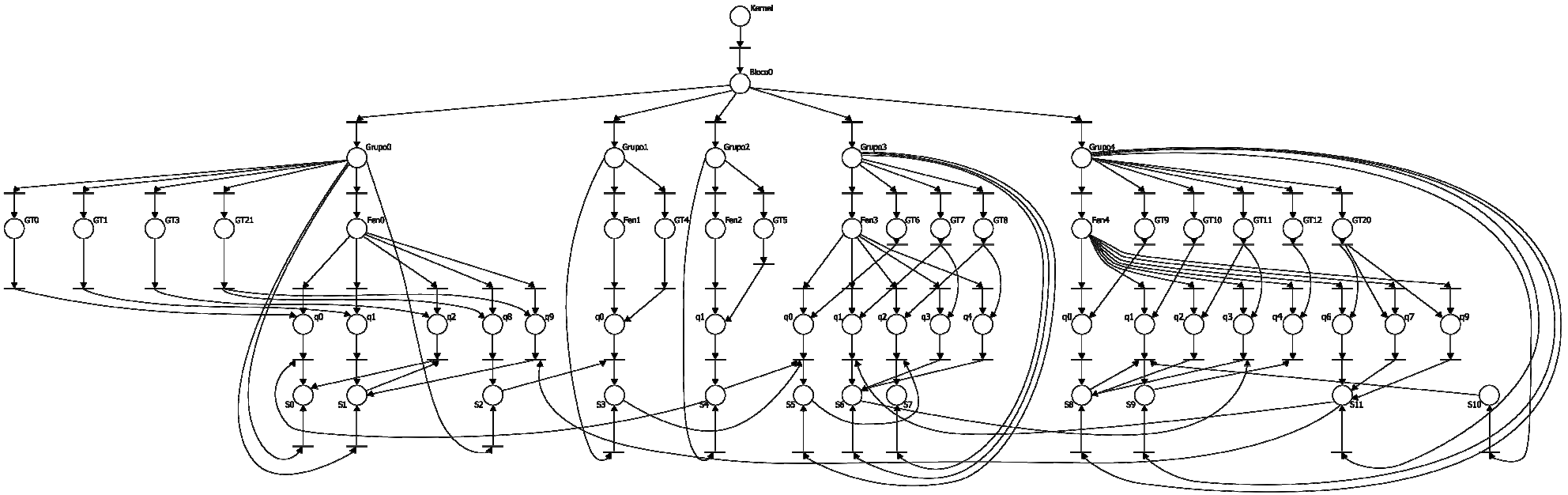
[19] *INA – Integrated Net Analyzer*. Disponível em: < http://www2.informatik.hu-berlin.de/~starke/ina.html>. Acesso em: 23 de maio de 2007.

[20] MENEZES, Carlos. *Caracterização de Dano por Hidrogênio em Aços API 5CT L80 13Cr por Meio de Ondas Ultra-Sônicas.* 2006. 130 f. Dissertação de Mestrado em Engenharia Metalúrgica e de Materiais, Universidade Federal do Rio de Janeiro, Rio de Janeiro.

Apêndice A

Modelo do Estudo de Caso

Representa o modelo em redes de Petri do Estudo de Caso, Capítulo 5, seção 5.2


Kernel
Bloco0
Grupo0
Grupo1
Grupo2
Grupo3
Grupo4
GT0
GT1
GT3
GT21
Fen0
Fen1
GT4
Fen2
GT5
Fen3
GT6
GT7
GT8
Fen4
GT9
GT10
GT11
GT12
GT20
q0
q1
q2
q8
q9
q3
q4
q6
q7
S0
S1
S2
S3
S4
S5
S6
S7
S8
S9
S11
S10