1 Introdução

Uma das áreas da Ciência da Computação que tem aumentado de interesse nos últimos anos é a do desenvolvimento de sistemas orientados a objectos, de que são exemplo os sistemas de informação, os sistemas operativos e as linguagens de programação, entre outros.

Um sistema orientado a objectos pode ser encarado como uma comunidade de objectos concorrentes, que interactuam entre si ([Sernadas e Sernadas 87], [Sernadas et al 89b], [Sernadas et al 89a], [Ehrich et al 91], [Ehrich e Sernadas 91]). Um objecto, por sua vez, pode ser encarado como uma entidade que possui um estado interno e que pode interactuar com outros objectos por troca de mensagens. Este estado interno caracteriza, em geral, os valores dos atributos, e apenas pode ser modificado pelo próprio objecto ([Sernadas e Ehrich 91]).

Neste contexto, e a juntar a algumas linguagens de especificação orientadas a objectos, de que são exemplo a linguagem OBLOG ([Sernadas et al 92a], [Sernadas et al 92b], [Sernadas et al 92c], [Sernadas et al 91], [ESD93], [Costa et al 89]), TROLL ([Jungclaus et al 91]), surgiu a linguagem GNOME ([Sernadas e Ramos 94], [Ramos e Sernadas 94]). A linguagem GNOME é uma versão textual e revista da linguagem OBLOG. Dispõe de novas primitivas de comunicação (comunicação síncrona por *chamada de serviços* e assíncrona por *emissão de eventos*), de capsulação (*regiões*) e de instanciação (*bibliotecas*), não dispondo, no entanto, de transacções nem de construções específicas para modelação de dados.

Um dos objectivos principais desta dissertação é definir uma semântica formal para um fragmento da linguagem GNOME, considerando apenas uma das primitivas de comunicação (chamada de serviços), e sem considerar as primitivas de capsulação e instanciação.

Segundo [Manna e Pnueli 91b] a lógica temporal é um dos formalismos vulgarmente aceites para especificação e verificação e sistemas concorrentes. Neste contexto, a lógica apresentada em [Sernadas et al 92e] e refinada em [Sernadas e Sernadas 94] para especificação e verificação de sistemas concorrentes, orientados a objectos, surge como um bom candidato para definir uma axiomatização para o fragmento da linguagem considerado. Pretende-se que esta axiomatização defina uma semântica formal para a linguagem.

Para definir esta axiomatização é, no entanto, necessário adaptar alguns dos conceitos apresentados em [Sernadas e Sernadas 94]. Em particular, é necessário introduzir, ao nível das assinaturas de classe, uma distinção entre acções internas e acções externas (ou serviços). Esta modificação conduz por sua vez a algumas alterações ao nível da semântica, tornando-se necessário, em particular, introduzir um mecanismo que de alguma forma represente o *ambiente* . A esta nova lógica chama-se *Lógica de Certificação GNOME* (GCL).

A lógica temporal, para além da especificação e verificação de sistemas concorrentes, pode também ser utilizada para especificação de propriedades sobre estes sistemas, de que são exemplo as *propriedades de segurança*, *propriedades de animação* e *propriedades de resposta*, entre outras. Assim, apresenta-se também um sistema dedutivo para raciocinar sobre especificações GCL, que inclui um conjunto de regras para raciocinar sobre algumas destas propriedades.

No segundo capítulo apresenta-se um fragmento da linguagem GNOME. A definição desta linguagem é feita através da definição de uma gramática e de alguns exemplos, seguindo sempre o que é apresentado em [Sernadas e Ramos 94] e [Ramos e Sernadas 94].

No terceiro capítulo define-se a Lógica de Certificação GNOME, de acordo com o que atrás se referiu. Ainda neste capítulo, apresenta-se uma "semântica axiomática" para o fragmento da linguagem GNOME.

No quarto capítulo introduz-se um sistema dedutivo para especificações GCL, com especial ênfase na prova de propriedades de segurança e propriedades de resposta, seguindo uma metodologia semelhante à de [Manna e Pnueli 91b], [Manna e Pnueli 93a] e [Manna e Pnueli 93b].