10

Conclusão

O desenvolvimento de sistemas de grande porte é uma tarefa difícil que exige um grande esforço dos projetistas. Por esta razão, na maior parte das vezes, generalidade é uma funcionalidade que não é priorizada. A primeira conseqüência disto é o pequeno reuso observado em desenvolvimento de *software*. Pronomes, como aqueles descritos e propostos aqui, permitem a criação de códigos genéricos que favorecem o desacoplamento entre classes, uma vez que possibilitam que objetos enviem mensagens sem identificar seus destinos. Este desacoplamento, por sua vez, permite que o reuso possa ser melhor explorado.

Neste trabalho, o conceito de pronomes, proposto em [Car93] foi explorado utilizando-se linguagens abertas. Com estas, a principal preocupação dos trabalhos que anteriormente já haviam explorado este conceito [Car93,Coe92] minimizar o conjunto de pronomes introduzidos na linguagem para não onerá-la pôde ser relaxada. Isto porque funcionalidades providas através de metaprogramação podem ser configuráveis. Em especial, o conjunto de pronomes incorporado a uma linguagem pode ser configurado para um determinado tipo de aplicação, não obrigando esta a conviver com o custo adicional imposto por pronomes não utilizados. Esta facilidade possibilitou que o conjunto de pronomes explorado neste trabalho pudesse abranger todos os pronomes anteriormente propostos, CREATOR em [Car93] e PARENT, ALL e ANY em [Coe92], e ainda ser estendido com dois novos pronomes: SENDER e MAIN.

Nos experimentos computacionais realizados, a principal desvantagem do uso de pronomes o aumento do tempo de execução do sistema pela necessidade deste de manter as informações necessárias para identificar o objeto, ou conjunto de objetos, representado pelo pronome se mostrou pouco significativa. Concluiu-se que apesar de realmente aumentar ligeiramente o tempo de execução do sistema, a diferença é bem pequena. Para que esta diferença se fizesse perceptível, foi necessária a simulação de configurações

que, provavelmente, nunca seriam encontradas em sistemas reais. Mesmo assim, esta diferença nunca passou de 1 segundo, tempo provavelmente insignificante se comparado com o tempo total de execução de um sistema real que se utilize das configurações simuladas.

A principal vantagem do uso de pronomes dar acesso direto a objetos que só poderiam ser acessados através da adição de código extra também pôde ser verificada nas aplicações exploradas neste trabalho. Em especial, muitas referências cruzadas puderam ser eliminadas, diminuindo assim o acoplamento entre as classes do sistema. Por exemplo, pronomes permitem a construção de classes servidoras puras cuja ligação com o mundo exterior se faça apenas através do envio de mensagens através destes. Estas classes, pelo seu alto grau de desacoplamento, poderiam ser utilizadas em qualquer contexto.

Além do desacoplamento, com a utilização de pronomes conseguiu-se reduzir a distância entre o projeto do sistema e sua implementação, uma vez que associações criadas apenas para servir de referências cruzadas puderam ser eliminadas.

O fluxo de mensagens do sistema também foi reduzido já que, com pronomes representando objetos importantes, qualquer objeto, em qualquer nível da árvore de agregação, pode enviar mensagens a estes diretamente, sem a necessidade que estas passem por vários objetos antes de chegar a seu destino.

A definição de um pronome genérico cuja semântica poderia ser definida dinamicamente pelo programador chegou a ser pensada como extensão deste trabalho. No entanto, a facilidade encontrada na definição de pronomes em linguagens abertas fez com que esta idéia fosse abandonada. Por outro lado, esta mesma implementação deixou claras algumas semelhanças existentes entre as implementações dos diferentes pronomes. Esta observação trouxe de volta a idéia do pronome genérico, não focando mais, no entanto, a definição de um pronome com semântica configurável, mas provendo-se uma forma, o mais simples e automática possível, de se definir e implementar novos pronomes. Isso foi feito definindo-se um *framework* cuja instanciação provê a incorporação de novos pronomes na linguagem alvo. Este *framework* foi, ainda neste trabalho, instanciado tanto para a obtenção dos pronomes já anteriormente explorados quanto para a obtenção de dois novos pronomes: CLOCK e SIBLINGS. Nesta experiência, verificou-se um alto grau de automatização e simplicidade na utilização do *framework*.

Uma possível extensão deste trabalho seria a introdução do conceito de redirecionamento. Desta forma, uma classe poderia, de forma declarativa, redirecionar mensagens recebidas por instâncias suas para atributos ou para pronomes. Isso resolveria um problema observado em sistemas experimentais que utilizam pronomes, onde, um grande número de vezes, tratadores de mensagens são implementados apenas para repassá-las a outro objeto visível em seu contexto.