CrowdCode: Uma Plataforma para o Desenvolvimento através de Grupos de Pessoas

Thomas D. LaToza[1], Eric Chiquillo[2], W. Ben Towne[3], Christian M. Adriano[1], André van der Hoek[1]

[1]Departamento de Informática
Universidade da California, Irvine
{tlatoza, adrianoc, andre}@ics.uci.edu

[2]Zynga
echiquil@gmail.com

[3]Insituto para a Pesquisa de Software
Universidade de Carnegie Mellon
wbt@cs.cmu.edu

Resumo

O crowdsourcing de micro tarefas organiza um trabalho que é complexo em vários fluxos de trabalho, decompondo tarefas grandes em tarefas pequenas e independentes que são executadas por trabalhadores que se presumem transitórios e possivelmente pouco fiáveis.

Aplicado ao desenvolvimento de software, este conceito pode ao mesmo tempo aumentar o paralelismo no trabalho de desenvolvimento e aumentar a participação no desenvolvimento open source, reduzindo as barreiras para as contribuições, possibilitando novos modelos económicos e permitindo que se desenvolvam programas substancialmente mais rápido. Contudo, presume-se que com o crowdsourcing de micro tarefas, o requerente possa previamente especificar o fluxo de trabalho. No desenvolvimento de software este pressuposto não é sustentável, pois a estrutura, o tipo e o conteúdo das tarefas são muito mais dinámicas. Como poderá então tal trabalho ser decomposto e coordenado em micro tarefas?

A nossa perspectiva é coordenar o trabalho através de um gráfico de processos, gerando micro tarefas em resposta a eventos que possam ocorrer nesses mesmos processos, em vez de utilizar um fluxo de trabalho concreto. Cada micro tarefa indica ao trabalhador para realizar uma tarefa pequena e bem definida num único processo (ex: uma sub-rotina ou um teste), permitindo ao trabalhador prosseguir em muitos processos em paralelo. À medida que os trabalhadores vão completando as microtarefas, são gerados eventos no processo, que poderão despoletar mais microtarefas a serem geradas. Quando um processo muda, são enviados eventos para os processos que estão dependentes desses mesmos eventos, permitindo as micro tarefas serem dinámicas e não possuirem hierarquias. Por exemplo, quando uma sub-rotina muda a sua assinatura (ex: adicionar um parâmetro), os processos que lhe estão dependentes (valores de retorno e testes) são notificados, gerando micro tarefas para lidar com essas mudanças. Como os processos poderão ter muitas dependências, poderão ter pendentes várias notificações de mudança. Para coordenar este trabalho, cada processo possui uma fila de micro tarefas que permite que cada alteração seja executada sequencialmente e assegura que essas mesmas mudanças não entrem em conflito. Finamente, este modelo suporta fluxos de trabalhos iterativos. Por exemplo, os programadores ao editar uma sub-rotina poderão escrever pseudo código, gerando tarefas iterativamente até que todo o pseudo código tenha sido subtituído.

Primeiramente o trabalho inicia com um conjunto de cenários descrevendo o comportamento desejado das aplicações (providenciado pelo requerente), gerando micro tarefas tendo em vista editar uma sub-rotina para dar início à implementação de cada cenário. À medida que os trabalhadores editam as sub-rotinas, poderão escrever pseudo valores, descrevendo uma chamada de uma sub-rotina que desejem ver. Após uma micro tarefa que verifica se alguma sub-rotina providencia o comportamento desejado, um passo recursivo pode ocorrer, criando uma nova sub-rotina e micro tarefas adicionais. À medida que as sub-rotinas vão sendo criadas, é gerada uma micro tarefa para enumerar e depois implementar casos de teste para a sub-rotina, introduzindo restrições entre os processos criados independentemente que possam ser conferidas para assegurar a qualidade. Quando as sub-rotinas ficarem completas, são corridos os testes; se os testes falharem, são geradas micro tarefas para eliminar os defeitos das sub-rotinas (ou editar os testes) para permitir que a sub-rotina passe nos testes. Para permitir que os trabalhadores eliminem os defeitos de uma sub-rotina isoladamente, mesmo que esta chame outras sub-rotinas, os trabalhadores podem ver e editar os valores de retorno das sub-rotinas recursivamente, criando novos testes para essa sub-rotina.

Implementámos esta nossa abordagem no CrowdCode, um protótipo de um Ambiente de Desenvolvimento Integrado na nuvem para o desenvolvimento em grupos de pessoas. CrowdCode permite o desenvolvimento de bibliotecas em Javascript (ex: código Javascript que pegue num objeto como entrada e produza um objeto como saída), que poderão ser incorporadas numa aplicação web. Estamos neste momento a avaliar a nossa abordagem utilizando o crowdsourcing com um reduzido número de pessoas para a construção de um pequeno programa.