Geração de Código Android para Aplicações CRUD Visando Múltiplas Plataformas de Nuvem

Kellerson Kurtz[1], **Lisane B. Brisolara**[1]

[1]Centro de Desenvolvimento Tecnológico – Universidade Federal de Pelotas (UFPEL)

{kkurtz,lisane}@inf.ufpel.edu.br

Resumo. *Este artigo propõe uma abordagem para geração automática de código a partir de modelos UML, voltada a aplicações Android que utilizam plataformas de nuvem por meio das operações CRUD. A abordagem abstrai detalhes da infraestutura de nuvem visando suportar múltiplas plataformas. O emprego da abordagem é demonstrado através de um estudo de caso, onde código é gerado para as plataformas Bluemix e Google App Engine.*

1. Introdução

O mercado de computação móvel tem crescido nos últimos anos. No entanto, o desenvolvimento de aplicações voltadas a dispositivos móveis traz uma série de desafios, associados, principalmente, a diversidade de plataformas de desenvolvimento empregadas e as limitações de capacidade de armazenamento, processamento e de consumo energético.

A computação móvel na nuvem vem sendo empregada para contornar estas limitações de *hardware*, através do armazenamento e ou processamento de dados externamente [Dinh et al. 2013]. Contudo, o desenvolvimento de aplicações que utilizam nuvem também envolve desafios adicionais, visto que cada plataforma de nuvem exige uma interface de comunicação distinta, dificultando o desenvolvimento e a portabilidade dos aplicativos. As operações CRUD - criar, ler, atualizar e remover dados - são as essenciais no emprego da nuvem para armazenamento de dados. É por meio destas operações que um grande número de aplicações móveis comunicam-se com a nuvem.

Abordagens de desenvolvimento orientadas a modelos vêm sendo propostas para lidar com os desafios do desenvolvimento de aplicativos móveis, dentre elas destacam-se a *GenCode* [Parada et al. 2015] e a *MobiCloud* [Ranabahu et al. 2011]. Dentre as abordagens encontradas na literatura, apenas a *MobiCloud* foca em aplicações que empregam o paradigma de computação na nuvem e o suporte as operações CRUD. Embora, um esforço interessante por gerar também o *back-end* das aplicações para as plataformas *Google App Engine* e *Amazon EC2*, *MobiCloud* propõe uma linguagem textual de domínio específico (DSL). Diferentemente da *MobiCloud*, a ferramenta *GenCode* [Parada et al. 2015] utiliza diagramas UML para gerar código nativo para aplicações *Android* e *Windows Phone*. O emprego de notações gráficas e padronizadas pode ser destacado como um ponto positivo desta abordagem. No entanto, a GenCode, não suporta a geração de código para aplicações envolvendo a nuvem.

Este trabalho propõe uma abordagem para geração de código Android a partir de modelos UML voltada a aplicações CRUD envolvendo a nuvem. A abordagem é independente de plataforma de nuvem, visando suportar a geração de código para múltiplas

plataformas. Para tal, uma abordagem padronizada para modelagem de aplicações CRUD é proposta, a qual foca na interação aplicação-nuvem. Para avaliar a abordagem proposta, uma extensão da ferramenta *GenCode* foi desenvolvida, a qual gera código de interação aplicação-nuvem para as plataformas *Bluemix* [1] e *Google App Engine* [2] . Através de um estudo de caso, o emprego da abordagem é demonstrado e discutido.

2. Visão Geral da Abordagem

A Figura 1 ilustra o fluxo proposto para a geração de código de aplicativos *Android* que envolvem operações CRUD na nuvem. Neste fluxo, a partir de diagramas UML construídos seguindo algumas regras de modelagem, a ferramenta *GenCode Cloud* gera código compatível com diferentes plataformas de nuvem. Atualmente, a ferramenta oferece suporte para as plataformas *IBM Bluemix* e *Google App Engine*.

Visando tornar a modelagem independente da nuvem empregada, nossa abordagem determina que o projetista deve especificar quais tipos de objetos serão hospedados em nuvem, quais operações CRUD (criar, ler, atualizar e remover dados) serão realizadas sobre cada tipo de objeto na nuvem e, por fim, quais classes serão responsáveis por realizar estas operações. Todas estas informações ficam representadas no modelo de classes.

A partir deste modelo independente de plataforma, a *GenCode Cloud* detecta, por exemplo, os métodos responsáveis pela comunicação com a plataforma de nuvem e os implementa para a plataforma de nuvem selecionada, abstraindo aos programadores todos os detalhes da programação da comunicação entre aplicação e a infraestrutura de nuvem. A ferramenta gera trechos de código dependentes de plataforma de nuvem baseando-se em informações extraídas do modelo de classes e em templates construídos com base em estudos de caso.

A *GenCode Cloud* é uma extensão da ferramenta *GenCode* [Parada et al. 2015], a qual emprega diagramas de sequência para a geração de código comportamental. Assim, para uma geração mais completa de código comportamental, a *GenCode Cloud* depende do detalhamento de outras partes do comportamento através de diagramas de sequência. Na Seção 3, a abordagem proposta será explicada através de um estudo de caso.

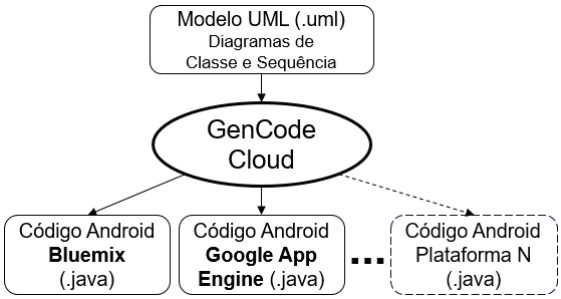

Figura 1. Fluxo para geração de código para aplicações CRUD na nuvem

3. Estudo de Caso: Aplicação Lista de Compras

Para detalhar o conjunto de regras que modelam os elementos para comunicação com a nuvem, vamos demonstrar a aplicação deste padrão na modelagem de um aplicativo

[1] http://www.ibm.com/cloud-computing/bluemix/br-pt/
[2] https://cloud.google.com/appengine/

chamado “Lista de Compras”, no qual o usuário adiciona itens a serem comprados em uma lista e esta é hospedada em uma infraestrutura de nuvem.

A Figura 2 apresenta o diagrama de classes simplificado da aplicação alvo. A partir da especialização da superclasse “*CLOUD*”, o projetista indica os tipos de objetos que serão hospedados na nuvem. Desta forma, a classe *Item* representa objetos que serão armazenados na nuvem pela aplicação. Classes derivadas de “*CLOUD*” também precisam ter o atributo “*key*” do tipo *String*, este será o identificador dos objetos usado nas buscas.

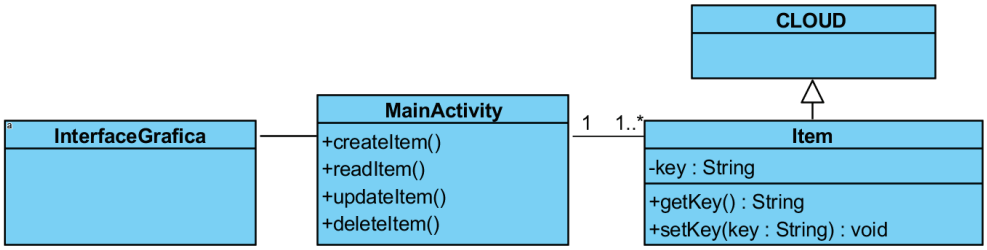

Figura 2. Diagrama de classes simplificado da aplicação Lista de Compras

A classe responsável pela interação de cada tipo de objeto com a nuvem deve conter os métodos que representam as operações CRUD empregadas na aplicação. Estes métodos devem seguir a nomenclatura “<operação><tipo_objeto>”, na qual “<operação>”deve ser substituído por “*create*”, “*read*”, “*update*” ou “*delete*”; e “<tipo_objeto>” pela classe do objeto manipulado. Como indicado no diagrama da Figura 2, a “*MainActivity*” realizará as operações CRUD sobre os objetos do tipo *Item*.

```
1       public void createItem(){
2           String key = "SET KEY";
3           if (!key.equals("")) {
4               Item item = new Item();
5               item.setKey(key);
6               createItemCloud(item);
7               itemList.add(item);
8           }
9       }
10
11      private void createItemCloud(Item item){
12              item.save().continueWith(new Continuation<IBMDataObject, Void>() {
13                  @Override
14                  public Void then(Task<IBMDataObject> task) throws Exception {
15                      if (task.isCancelled()){
16                          Log.e(CLASS_NAME,"Exception:Task"+task.toString()+" was cancelled.");
17                      }
18                      else if (task.isFaulted()) {
19                          Log.e(CLASS_NAME, "Exception : " + task.getError().getMessage());
20                      }
21                      else {
22                          listItems();
23                      }
24                      return null;
25                  }
26              });
27      }
```

Figura 3. Código gerado para a operação criar item para *Bluemix*

A partir, unicamente, do diagrama da Figura 2, a ferramenta *GenCode Cloud* gera automaticamente 220 linhas de código quando selecionada a plataforma *Bluemix*,

e 170 linhas quando selecionada a *Google App Engine*. Estas linhas de código incluem, além da estrutura da aplicação (classes com atributos e métodos), a implementação dos métodos CRUD, bem como comandos de configuração e tratamentos de exceção para a comunicação com a plataforma selecionada. Quando, juntamente ao diagrama de classe, são empregados diagramas de sequência para descrever mais detalhes do comportamento da aplicação, a ferramenta é capaz de gerar um número maior de linhas automaticamente.

A Figura 3 ilustra o trecho de código gerado para a operação de criação de um item para armazenamento na nuvem. Neste código, no método *createItem*, que é independente de plataforma de nuvem, o desenvolvedor deve inicializar atributos para o novo item. Este método invoca o método *createItemCloud*, que representa o procedimento requerido pela plataforma *Bluemix* para realizar esta operação. Se a plataforma escolhida for a *Google App Engine*, uma implementação diferente do método *createItemCloud* será gerada pela *GenCode Cloud*, de forma a atender o formato de interação aplicativo-nuvem empregado pela plataforma.

4. Conclusão

Este artigo propôs um fluxo automatizado para geração de código a partir de modelos UML, com foco em aplicações *Android* envolvendo operações CRUD na nuvem. Para gerar código referente à comunicação com a nuvem, bem como a implementação das operações CRUD, uma abordagem de padronização da modelagem foi também proposta. Esta padronização baseia-se em um número pequeno de regras a serem empregadas no diagrama de classes e que tornam o modelo independente de plataforma.

Para suportar o fluxo proposto, a ferramenta *GenCode Cloud* foi implementada como uma extensão da *GenCode*. Esta ferramenta recebe como entrada o modelo construído conforme o padrão e gera trechos de código, estrutural e comportamental, referentes a interação com a plataforma de nuvem. Desta forma, os projetistas implementam mais facilmente aplicações CRUD envolvendo a nuvem sem que sejam requeridos conhecimentos específicos sobre a programação com uma dada plataforma de nuvem. Além disso, a partir de um mesmo modelo UML, código para diferentes infraestruturas de nuvem podem ser gerados, facilitando a portabilidade. Atualmente, nosso estudo de caso considerou as plataformas *Google App Engine* e *Bluemix*. Em trabalhos futuros, pretende-se suportar outras importantes plataformas, como *Microsoft Azure* e *Amazon Web Services*.

Referências

Dinh, H. T., Lee, C., and an P. Wang, D. N. (2013). A survey of mobile cloud computing: architecture, applications and approaches. *Communications and Mobile Computing, vol. 13*, pages 1587–1611.

Parada, A., Marques, M., and Brisolara, L. (2015). Automating mobile application development: Uml-based code generation for android and windows phone. *Revista de Informática Teórica e Aplicada, vol. 22*, pages 31–50.

Ranabahu, A. H., Maximilien, E. M., Sheth, A. P., and Thirunarayan, K. (2011). A domain specific language for enterprise grade cloud-mobile hybrid applications. In *Proc. of the Compilation of the Co-located Workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11*, SPLASH '11 Workshops, pages 77–84, New York, NY, USA. ACM.