Diego Lins de Freitas

Uso de Padrões de Projetos no desenvolvimento de aplicativos Android

Faculdade Fucapi

Especialização em Engenharia de Software

Centro de Pós-Graduação e Extensão (CPGE)

Manaus-Am

2013, Julho

Resumo

A plataforma android foi adotada por vários fornecedores de smartphones e tablet promovendo uma disseminação do sistema operacional muito abrangente. Formou-se um grande mercado a ser explorado com fornecimento de aplicativos para as mais variadas finalidades. Para atender essa demanda é necessário desenvolver aplicativos de qualidade e o mais rápido possível mas por se tratar de pequenos aplicativos há pouca preocupação com a arquitetura é deixada de lado. Os padrões difundidos na comunidade são focados na utilização de componentes e desenvolvimento da interface. Neste artigo é apresentado alguns padrões de projetos aplicados ao desenvolvimento de aplicativos android dando um visão mais ampla de um aplicativo.

Palavras-chaves: Android. MVC. MVP.

Sumário

1 Introdução

A tendência do mercado de dispositivos handset é aumentar segundo IDC é esperado um crescimento de 32.7% na produção(IDC, 2013b). O sistema operacional android é o líder dominando 75% desse mercado com mais de 162 milhões de smartphones produzidos e embarcados com android(IDC, 2013a). Baseados nesses dados podemos concluir que existe uma damanda no desenvolvimento de novos aplicativos que agregem valor à esses aparelhos. Para produzir aplicativos de qualidade é necessário aplicar boas práticas de desenvolvimento de software. Levando em consideração que a linguagem de programação usado para o desenvolvimento na plataforma android é a Java, é natural que se aplique as práticas definidas pelos princípios de projeto orientado a objetos. Esses princípios guiam o desenvolvedor em como definir a estrutura interna do software afetando diretamente as características de qualidade como manutenabilidade, performance e outras(YANG; TEMPERO; MELTON, 2008).

Os padrões de projeto são aplicados na engenhsaria de software como forma de reproduzir soluções para problemas recorrentes melhorando a manutenabilidade e o reuso de componentes de software(GAMMA et al., 1995) então o uso de padrões de projetos é um meio de aplicar esses princípios. Segundo Pressman (2006), "A interface com o usuário pode ser considerada o elemento mais importante de um sistema ou produto baseado em computador", tendo em vista isso, será tratado nesse trabalho os padrões para desenvolvimento da camada de apresentação de em aplicativos android.

1.1 Motivação

A motivação para a execução desta pesquisa sugiu da participação do autor em projetos para desenvolvimento de aplicativos android em uma instituição de pesquisa e desenvolvimento localizada em manaus. O comprometimento com a qualidade promoveu o uso de ferramentas de código aberto para monitoração da qualidade dos projetos além do processo de testes adotado pela empresa. Com uma equipe composta por mais de 30 desenvolvedores produzindo aplicativos com diversas finalidades houve a necessidades de padronização da arquitetura.

1.2 Problematização

Dado que existem uma vasta diversidade de padrões de proejtos a serem aplicados no desenvolvimento de software, Qual padrão de projeto é aplicável para o desenvolvi-

mento da camada de apresentação de aplicativos android para melhorar a qualidade do produto final?

1.3 Hipótese

O padrão de projeto Model-View-Presenter é aplicável no desenvolvimento da camada de apresentação de aplicativos android, gerando um aumento da qualidade do produto final.

1.4 Objetivos

Este trabalho fará um estudo sobre a arquitetura de aplicativos android com o propósito de melhorar a qualidade desses softwares reduzindo riscos relacionados as mudanças que acontecem durente o ciclo de desenvolvimento e promova a reutilização de componentes e que permita o incremento de funcionalidades com baixo impacto. Este estudo fornecerá insumos para que os desenvolvedores possam ter uma visão geral da aplicabilidade dos padrões de projetos e analisar quais técnicas contribuem para o seu trabalho. Este trabalho tem como meta:

- Estudar os padrões de projeto que podem ser usados para a implementação da camada de apresentação de um aplicativo android.
- Avaliar os componentes do framework android identificando seus papéis e responsabilidades de acordo com os padões estudados levando em consideração características que podem dificultar a aplicação dos padrões.
- Identificar os impactos nas características de qualidade do aplicativo de acordo com o paradigma da Orientação a objetos.
- Elaborar um referência de implementação para a camada de apresentação de um aplicativo android utilizando padrões de projetos.

1.5 Trabalhos Relacionados

1.6 Contribuições

Tendo em vista que padrões de projetos descrevem soluções genéricas, este trabalho contribui com uma interpretação do padrão de projeto mvp proporcionando uma referência prática dentro framework android.

1.7 Organização do Trabalho

Será descrito a metodologia, definicão do projeto, experimentos, conclusões.

2 Metodologia

Esta pesquisa se caracteríza como aplicada pois tem fins práticos ao realizar uma revisão na literatura existente sobre orientação a objetos e seus princípios bem como referências sobre padrões de projetos para proporcionar o embasamento que auxiliará na identificação dos componetes do android que podem assumir as responsabilidades definidas nos padrões e definir como implementa-los.

Para validar a hipótese apresentada neste trabalho a pesquisa terá uma abordagem quantitativa através da análise de dados estatistícos de métricas que expresssam atributos de qualidade em software orientado a objetos.

Essas medidas serão coletadas através de um procedimento experimental em laboratório utilizando um processo iterativo-incremental para executar refatorações no código afim de introduzir o padrão de projeto nos aplicativo e a cada iteração será feita a coleata das métricas. Para executar esse processo será identificado um conjunto de estórias que o aplicativo atende relaciondas as interações com o usuário. Desse conjunto será elencadas as estórias que serão refatoradas para cada iteração.

2.1 Objeto de Estudo

O projeto a ser refatorado será o aplicativo de Contatos do android[1] que é um dos aplicativos básicos pré-instalados com o sistema operacional. Este projeto é opensource mantido pelo The Android Open Source Projeto suportado pela Google com contribuições de desenvolvedores de do mundo todo.

Os critérios para a escolha do objeto de estudo são:

1. Tamanho/Complexidade - Um projeto muito complexo iria inviabilizar a pesquisa devido ao esforço para fazer a refatoração e as métricas coletadas apartir de projeto simples e triviais não forneceria dados suficientes para uma análise satisfatória. (o que foi usado para medir a complexidade. e qual o tamanho do objeto de estudo).

2. Open Soruce - Além de permitir o acesso ao código fonte sem limitações, tem a contribuição de vários desenvolvedores com experiência e formação em programação diversificada que se refletem no código fonte.

3. Origem - A escolha de um aplicativo mantido sobre o mesmo "guarda-chuva" que o sistema operacional android foi feita com o intuito de fazer a pesquisa uma código

[1] https://github.com/android/platform$_p ackages_a pps_c ontacts$

fonte que expresse as técnicas e práticas de programação difundidas nesse ecosistema.

2.2 Processo de Experimentos

A Figura 1 demonstra as atividades do processo de experimentação adotado:

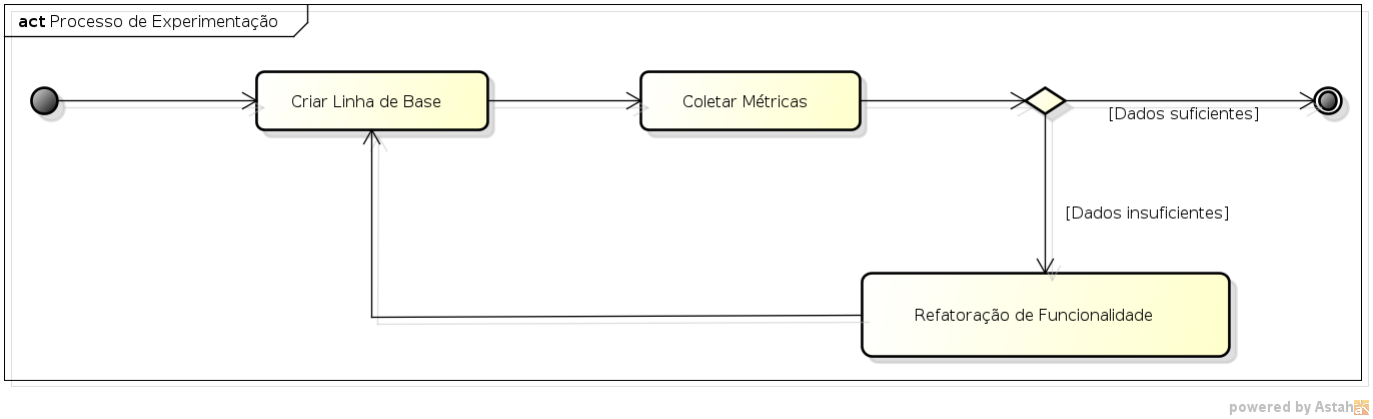

Figura 1 – Processo de Experimentação Fonte: Próprio Autor

Criação uma linha de base - Delimitar um marco no estado do código no repositório.

Refatoração Incremental Esta atividade tem como objetivo aplicar os padrões em uma funcionalidade do aplicativo.

Coleta de Métricas Este passo tem como objetivo fazer a coleta das métricas do código apartir de uma revisão em que se encontra no repositório para fazer a avaliação dos efeitos da refatoração na qualidade do código.

2.3 Ferramentas

Para elaborar uma documentação da implementação de referências usando será usada a notação UML v2 com o uso da ferramento Astah. O código será versionado no Github onde será feito o gerenciamento das linhas de base. As ferramentas a serem utilizadas para a refatoração, IDE Eclipse(Juno) com plugin ADT v21 para a refatoração. As métricas serão coletadas com a ferramenta PMD 5.

3 Referencial Teórico

3.1 Princípios e Padrões de Projetos

Desenvolver software orientado a objetos é um desafio. Criar uma representação computacional de uma faceta da realidade em que seus constituintes trabalhem de forma horminiosa para atingir as necessidades que o software se propõe a atender requer experiência, conhecimento do domínio do problema e um processo de análise e projeto. Apesear de existir várias abordagens para se conceber um sistema orientado a objetos(EVANS, 2004),(GOMAA, 2011) um sistema bem contruído apresetam características fundamentais. Coesão, acoplamento etc (DEMARCO, 1979),(PAGE-JONES, 1988).

Um padrão dentro do contexto de estudo deste presente trablho pode ser definido como uma técnica efetiva cuja a sua aplicabilidade é aceita e difundiida dentro de uma área de conhecimento com a intenção de atingir um objetivo(METSKER; WAKE, 2006).

Em densenvolvimento de software o catálogo mais difundido de padrões de projetos orientado a objetos é elaborado por Gamma et al. (1995) contendo um total de 23 padrões formalmente documentados que acumulam experiências bem sucedidas diversos sistemas. Esses padrões tem a seguinte classificação:

Criacionais Padrões que definem como criar novas instâncias de classes.

Estruturais Foca na estruturação das classes e objetos.

Comportamentais Definem como as classes e objetos interagem entre si e suas responsabilidades.

Com o intuito de documentar os padrões de projetos de forma objetiva para facilitar o aprendizado dos mesmos, a comparação e escolha do padrão a ser aplicado, Gamma et al. (1995) elabora um modelo que descreve as seguintes caracteristicas para cada padrão:

Nome e Classificação

Intenção

Também conhecido como

Motivação

Aplicabilidade

Estrutura

Participantes

Collaborations How the participants collaborate to carry out their responsibilities.

Consequences How does the pattern support its objectives? What are the trade-offs and results of using the pattern? What aspect of system structure does it let you vary inde- pendently?

Implementation What pitfalls, hints, or techniques should you be aware of when implementing the pattern? Are there language-specific issues?

Sample Code Code fragments that illustrate how you might implement the pattern in C++ or Smalltalk.

Known Uses Examples of the pattern found in real systems. We include at least two examples from different domains.

Related Patterns

Analisando a forma como um padrão de projeto é concebido definindo papéis de cada elemento participante e como eles interagem entre se podemos concluir que os padrões de projeto promovem maior coesão, melhor separação de interesses e baixo acoplamento no sistema.

Model View Controller

Onde Quando surgiu

Necessidade

Model

View

Controller

Relação com os padroes e principios.

Segundo Fowler (2002) “Of these the separation of presentation from model is one of the most fundamental heuristics of good software design”.

MVP

Onde Quando surgiu

Necessidades

Model

View

Presenter

Diferencas entre o mvc.

Android

Historia caracteristicas Arquitetura.

Componentes Activity service broadcastReceiver ContentProvider

Como DP resolvem problemas, seguir linha de raciocinio para aplicar padrões em android.

Analise do componente Activity, UI Thread, hierárquia de herança e algumas funcionalidades que dependem da activity e como isso interfere na aplicação do padrão(Por experiência confirmo que é negativa). mostrar exemplo de uma view em lista para smartphone e outra para tablet com grid usando o mesmo modelo

Experimentos

activity - controller+async task - listeners, activity - controller+async task - observer pattern, activity - controller+async task - localbroadcast activity - controller+async task - Messages Activity+Fragment

3.2 Refatoração do Objeto de estudo

Com base na literatura revisada esta seção tem como objetivo fazer uma análise dos componentes do framework android e projetar uma camada de apresentação utilizando o padrão MVP. Será realizado uma análise tendo como objetivo definir usada um referência de implemntação a ser aplicada na refatoração(Referências sobre Refatoração de Código) Quem vai ser o model?

OnTouchListener pode exercer papel de controller pois pode ser usado para interpretar os gestos do usuário e direcioar para o model.

Na definição clássica do padrão MVC é possível identificar a implementação do padrão Observer usado para a comunicação entre o model e a view.

Observer=Lapsed listener problem=memory leak e bugs porque uma activity pode ser ou estar sendo referenciada por um listener que se não for desregistrado vai causar problema

Destacar problema com o observer pattern e pra solucionar isso usar publish–subscribe messaging pattern

O Broadcast Receiver é uma implementação do padrão pubsub onde diversos subscribers se registram para receber mensagens(intents) de seu interesse.

Padrões Criacionais. O objeto Application é um singleton, a activity e fragmentos são criados pelo android e fica dificil aplicar padrões criacionais é possível identificar o padrão templete method nos ciclos de vida desses componentes.

Para concluir a seção sobre MVC e MVP destacar que A principal características desses padrões de projetos é que ele promover maior coesão nos citar Tom deMarco

baseado em outras pesquisa será aplicado padões de projetos para melhorar a qualidade pegar referências, relacionar Coesão = Qualidade.

4 Resultados e Conclusões

4.1 Trabalhos Futuros

Existem outros padrões de projetos para o desenvolvimento da camada de apresentação de um software que não foram analisados nesse trabalho: MVVM, MVP-VM, MVPC.

Este trabalho não fez uma avaliação dos impactos na performance do aplicativo devido ao uso desses padrões. A inclusão de mais objetos interagindo, redirecionando mensagens pode depreciar a performance.

4.2 Conclusões

Possível aumento do código.

Referências

DEMARCO, T. *Structured Analysis and System Specification*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1979. ISBN 0138543801. Citado na página 11.

EVANS, E. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. [S.l.]: Addison-Wesley, 2004. Citado na página 11.

FOWLER, M. *Patterns of Enterprise Application Architecture*. Boston, MA, USA: Addison-Wesley, 2002. ISBN 0321127420. Citado na página 12.

GAMMA, E. et al. *Design Patterns: Elements of Reusable Object-Oriented Software*. [S.l.]: Addison Wesley, Boston, 1995. (Professional Computing Series). Citado 2 vezes nas páginas 5 e 11.

GOMAA, H. *Software modeling and design : UML, use cases, patterns, and software architectures*. [S.l.]: Addison-Wesley, 2011. Citado na página 11.

IDC. *Android and iOS Combine for 92.3System Shipments in the First Quarter While Windows Phone Leapfrogs BlackBerry*. 2013. IDC - Worldwide Quarterly Mobile Phone Forecast. Disponível em: <http://www.idc.com/getdoc-.jsp?containerId=prUS24143513>. Acesso em: 9.7.2013. Citado na página 5.

IDC. *Smartphones Expected to Grow 32.7% in 2013 Fueled By Declining Prices and Strong Emerging Market Demand*. 2013. IDC - Worldwide Quarterly Mobile Phone Forecast. Disponível em: <http://www.idc.com/getdoc.jsp?containerId=prUS24143513>. Acesso em: 9.7.2013. Citado na página 5.

METSKER, S. J.; WAKE, W. C. *Design Patterns in Java*. 2. ed. Upper Saddle River, NJ: Addison-Wesley, 2006. ISBN 978-0-321-33302-5. Citado na página 11.

PAGE-JONES, M. *The practical guide to structured systems design*. 2. ed. ed. London: Prentice-Hall International, 1988. (Yourdon Press Computing Series). ISBN 0136907776. Citado na página 11.

PRESSMAN, R. S. *Engenharia de Software*. 6. ed. [S.l.]: Mcgraw-hill Interamericana, 2006. Citado na página 5.

YANG, H. Y.; TEMPERO, E.; MELTON, H. An empirical study into use of dependency injection in java. In: *Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on*. [S.l.: s.n.], 2008. p. 239–247. ISSN 1530-0803. Citado na página 5.