Capítulo 3

Governança de Desenvolvedores em Ecossistemas de Software

Awdren de Lima Fontão, Igor Scaliante Wiese, Rodrigo Pereira dos Santos e Arilo Claudio Dias-Neto

Abstract

In a Software Ecosystem (SECO), software organizations have started to open their internal structure to external developers to reach goals, such as increasing the number of mobile applications (apps). In this way, the organization needs to develop and evolve the mechanisms of developer governance (i.e., concepts, values and practices) as a way to maintain ecosystem sustainability and diversity. The purpose of this paper is to present definitions and strategies relevant to SECO developer governance. This is a work with an introductory scope that presents the fundamentals of the following topics: SECO, Developer Relationships and Mining Software Repositories.

Resumo

Em um Ecossistema de Software (ECOS), as organizações de software passaram a abrir a sua estrutura para desenvolvedores externos visando atingir metas, como o aumento do número de aplicações móveis (apps). Desta forma, a organização precisa elaborar e evoluir os mecanismos (i.e., conceitos, valores e práticas) de governança de desenvolvedores como forma de manter a sustentabilidade e a diversidade do ecossistema. O objetivo deste trabalho é apresentar definições e estratégias relevantes para a governança de desenvolvedores em ECOS. Este é um trabalho com escopo introdutório que apresenta os fundamentos dos seguintes tópicos: ECOS, Relações com Desenvolvedores e Mineração de Repositórios.

3.1. Introdução

As organizações que mantêm plataformas de software, como Apple e Google, têm investido em uma infraestrutura para recrutar e engajar desenvolvedores externos como forma de contribuir para a expansão da plataforma. Este cenário que envolve o

relacionamento dos desenvolvedores com uma organização central (*keystone*) por meio de uma plataforma tecnológica central tem sido estudado na Engenharia de Software como Ecossistema de Software (ECOS) [Jansen e Brinkkemper 2009].

Um ecossistema envolve as dimensões técnica (o processo de desenvolvimento), social (a interação entre organização e desenvolvedores) e negócio (balanço entre os objetivos da organização e as expectativas do desenvolvedor). Neste contexto, o desenvolvedor é um ator essencial, pois é ele quem utiliza os recursos disponibilizados pela organização central para gerar contribuições para o ecossistema, que serão consumidas e avaliadas pelos usuários [Fontão *et al.* 2016]. O desempenho do desenvolvedor precisa ser acompanhado com o objetivo de identificar e prever áreas de melhoria. Este cenário na literatura técnica se refere ao conceito de saúde, que Manikas e Hansen (2013) definem como "*a habilidade do ecossistema em suportar perturbações e permanecer variável e produtivo durante o tempo*".

As organizações utilizam modelos de governança para atingir suas metas, melhorar os recursos disponíveis e, em última instância, aumentar seus lucros e reduzir eventuais riscos [Jansen e Cusumano 2012]. A governança consiste em um modelo que reúne um conjunto de premissas, conceitos, valores e práticas relativos à organização, ao relacionamento entre as partes envolvidas e a como os recursos são administrados e monitorados para que atinjam as metas estabelecidas [Alves *et al.* 2017].

O bem-estar econômico e social dos desenvolvedores assim como relacionamentos sinérgicos entre as expectativas do desenvolvedor e os objetivos da organização central devem ser garantidos por um grupo de relacionamentos com os desenvolvedores (DevRel, do inglês *Developer Relations*). Neste cenário, o ecossistema emerge como forma de apoiar a diversidade e sustentabilidade dos negócios da organização que mantém a plataforma.

Como a organização mantenedora do ECOS não mantém uma relação direta com um conjunto de desenvolvedores, estes usam repositórios como: portais de perguntas e respostas (StackOverflow) e repositórios de código (Github), entre outros, como uma forma de avançar dentro do ecossistema. Logo, os repositórios são fonte de informação relevante para adaptação de estratégias de governança de desenvolvedores.

Nesse contexto, o objetivo deste trabalho é apresentar definições e estratégias relevantes para a governança de desenvolvedores em ECOS. Este é um trabalho com escopo introdutório que envolve os seguintes tópicos: ECOS, Relações com Desenvolvedores e Mineração de Repositórios. A governança de desenvolvedores cobre três dimensões: social, técnica e negócio. Além disso, busca promover a sinergia nas relações que envolvem as expectativas dos desenvolvedores e os objetivos da organização que mantém um ecossistema.

Em uma edição em que o SBSI tem como tema "Sistemas de Informação: uma perspectiva social, sustentável e de negócios", discutindo uma prática gerencial que aplica as características das redes sociais e ferramentas sociais na administração de uma organização como forma de criar valor para os envolvidos, este trabalho vem então a contribuir com conceitos e aplicações em um cenário que envolve novos tipos e interações com/de sistemas de informação. No nosso caso, demonstramos as estratégias

relevantes para governar desenvolvedores e abordamos o uso de repositórios como fonte de informações técnicas, sociais e de negócios relativas aos desenvolvedores.

3.2. Ecossistemas de Software

Os relacionamentos que passaram a existir com a evolução do desenvolvimento de software, envolvendo componentes, infraestrutura e serviços de outras empresas, direcionaram o cenário do produto de software para um ecossistema Desta forma, fornecedores e consumidores de produtos de software começaram a criar tecnologia de forma colaborativa para gerar valor [Jansen e Bloemendal 2013].

Em [Jansen *et al.* 2009], os autores se referem a ECOS como um conjunto de negócios funcionando em unidade e interagindo com um mercado compartilhado de software e serviços, junto com suas relações apoiadas por uma plataforma tecnológica central e realizadas por meio de troca de informação, recursos e artefatos. Segundo Bosch (2009), ECOS consiste de uma plataforma de software, um conjunto de desenvolvedores internos e externos e uma comunidade de especialistas a serviço de uma comunidade de usuários que constroem soluções relevantes para satisfazer as suas necessidades.

3.2.1 Elementos de ECOS

O cenário de ecossistemas (Figura 3.1) fez novos modelos de negócios surgirem na Engenharia de Software, redefinindo os papéis e padrões de colaboração e inovação, criando complexas redes de organizações ou comunidades de contribuidores. Este cenário torna o ECOS um tema de pesquisa novo e importante na área de Engenharia de Software [Manikas 2016]. Para entender o tema "ECOS", é importante ter uma visão sobre os tipos de elementos que os compõem, seus papéis e atividades descritos a seguir [Jansen *et al.* 2013]:

- **Plataforma:** termo genérico que se refere a padrão de arquitetura, protocolo de comunicação ou qualquer conhecimento fundamental e compartilhado. Uma plataforma é a base na qual elementos técnicos de um ecossistema são construídos. Provê o suporte para a customização em larga escala;
- **Organização Central:** provê padrões e tecnologias que são o fundamento, ou parte, do ECOS. Cria e compartilha valor dentro do ECOS. Precisa ter visão geral do ECOS e saber qual o foco do contribuidor para assim identificar oportunidades. Influencia, define padrões e práticas, acelera a especialização e aumenta o valor em número de contribuidores. Tem como missão melhorar a saúde do ECOS pela disponibilização de um conjunto estável e previsível de artefatos comuns que outros elementos podem utilizar para construir, por meio do reúso, suas próprias contribuições dentro do ECOS. A organização central pode monitorar a saúde do ECOS a tomar medidas para promover a saúde do ECOS, se necessário. Para isso, a organização central deve ter visão geral do ECOS e estar consultando as suas medidas da saúde;
- **Desenvolvedor:** requer que os padrões ou tecnologias que sejam fornecidos pela organização central e que possam gerar valor de negócio estejam bem definidos e divulgados. Existem alguns perfis de contribuidores: os *evasivos*,

que participam em dois ECOS para minimizar riscos; os *discípulos*, que adotaram recentemente a plataforma e a divulgam para outros; e os *influenciadores*, que requisitam características da plataforma, organizam conferências e formam comunidades. Existe ainda o *agente intermediário* que serve como uma interface entre dois contribuidores. O desenvolvedor é responsável por novas ideias e pelo desenvolvimento de aplicações móveis no ECOS, respondendo aos requisitos dos usuários [Fontão *et al.* 2014]. Quando um desenvolvedor adota uma API para o desenvolvimento de aplicações, ele pode ser considerado um cliente [Kim *et al.* 2002]. Desenvolvedores podem ser classificados em individual ou organizacional [Hyrynsalmi *et al.* 2014];

- **Usuário:** é a pessoa, companhia ou entidade que pode comprar ou obter um produto parcial ou completo de um ECOS ou de um contribuidor;
- **Comunidade:** estrutura de colaboração e coordenação de atividades de um ECOS, composta por contribuidores internos e externos [Miranda *et al.* 2014]. Existem comunidades específicas, tais como: comunidades de usuários, comunidades de desenvolvedores e comunidades de especialistas, sendo que esta última realiza treinamento ou serviços de suporte [Taylor 2013];
- **Evangelista:** equipe de profissionais que mantém a relação com os desenvolvedores da organização que participa em treinamentos, palestras e competições de desenvolvimento com o objetivo de ajudar na expansão do ECOS e na formação de novos desenvolvedores [Fontão *et al.* 2014]. O evangelista faz parte da comunidade de especialistas e é um funcionário interno da organização do ECOS. É um especialista em um campo específico e tem conhecimento das atividades dentro do ECOS [Taylor 2013].

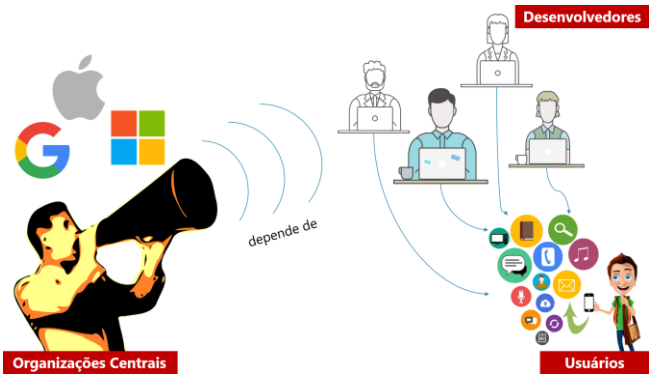

Figura 3.1. Um cenário em ECOS

3.2.2 Tipos e Dimensões de ECOS

Segundo Manikas (2016), pode-se classificar os ECOS por uma perspectiva de meios de criação de valor no ecossistema, como segue:

- **Proprietário:** a criação de valor no ecossistema é baseada em contribuições proprietárias. Normalmente, é protegido por processos de gerenciamento de propriedade intelectual e o valor se refere à compensação monetária. Industrial, plataforma como serviço e ecossistemas de comércio eletrônico são incluídos nesta categoria;

- **Código aberto:** as contribuições estão abertas aos participantes do ecossistema. O valor se refere a compensações não monetárias, e.g., conhecimento e experiência, ou pesquisa de satisfação. A Fundação Eclipse, o Gnome, a Fundação Apache e os ecossistemas do governo são incluídos nesta categoria;
- **Híbrido:** apoia contribuições proprietárias e de código aberto. Os seguintes ecossistemas foram adicionados a esta categoria: Android, iOS, Windows Phone e ecossistema de software móvel.

Além do contato com outras áreas, podem ser observadas oportunidades de pesquisa em dimensões dentro de ECOS. Santos e Werner (2012) apresentam uma visão "3+1" das dimensões de ECOS, como exibido na Figura 3.2:

- ***Dimensão Técnica:*** foca na plataforma de um ECOS. Isto engloba o mercado, a tecnologia, infraestrutura ou organização, definindo assim o seu ciclo de vida e as características da plataforma;
- ***Dimensão de Negócios:*** o fluxo do conhecimento dentro de um ECOS é o objetivo da análise a partir desta dimensão;
- ***Dimensão Social:*** todas as relações entre partes envolvidas e interessadas dentro de um ECOS, os *stakeholders*, fazem parte desta dimensão. Tem como objetivo ainda entender qual o motivo dos envolvidos integrar, estender e modificar o conhecimento em um ECOS e a interação entre os envolvidos;
- ***Dimensão de Gerenciamento e Engenharia:*** junção das três dimensões anteriores, por meio de três relacionamentos:
    - *Motivando o desenvolvimento e a evolução da plataforma*: entendimento dos relacionamentos e modelos do sistema, mas não de forma técnica;
    - *Contribuindo para o estabelecimento da plataforma:* envolvimento e atenção para a comunidade;
    - *Mapeamento de proposições valorosas e realizações:* conceito de valor da plataforma do ponto de vista de todos os *stakeholders* e qual o sentido deste valor.

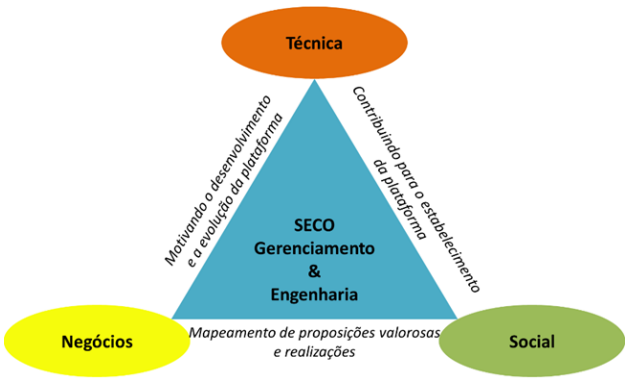

Figura 3.2. Visão "3+1" das dimensões de um ECOS
Fonte: [Santos e Werner, 2012]

3.3. Governança em ECOS

Governança é definida como a forma que uma organização é gerenciada, incluindo suas responsabilidades e processos de apoio à decisão [Dubinsky e Kruchten 2009]. A governança envolve a atribuição de responsabilidades e direitos de decisões, assim como indicadores e políticas que permitem a avaliação contínua. O gerenciamento e monitoramento bem-sucedidos ainda são grandes desafios para profissionais de ecossistemas [Dhungana *et al.* 2010]. Isto acontece porque a comunidade de pesquisa em ECOS ainda carece de teorias específicas de gerenciamento, ferramentas de suporte e experiência consolidada no tema [Manikas 2016].

Alves *et al.* (2017) definem governança de ecossistemas como ferramentas gerenciais para os atores do ecossistema que têm o objetivo de influenciar a saúde do ECOS. Saúde, neste contexto, se referee à garantia de que o ecossistema está funcionando como esperado. Indicadores específicos podem ser utilizados para prover uma visão geral do estado do ecossistema e ao mesmo tempo direcionam para ações e permitem a comparação com outros ecossistemas. Para medir o quanto um ecossistema é saudável, Iansiti e Richards (2006) definem as métricas:

- **Produtividade:** é a habilidade para criar energia. Pode ser medida por meio do fator total de produtividade, melhoria de produtividade por meio do tempo e entrega de inovações (i.e., habilidade do ecossistema de se adaptar e entregar aos seus membros novas tecnologias, processos e até mesmo ideias);
- **Robustez:** é a habilidade do ecossistema de sustentar perturbações e desligamentos. É medida por meio da persistência da estrutura do ecossistema (capacidade de manter os relacionamentos), previsibilidade (capacidade do núcleo de um ecossistema permanecer sólido mesmo acontecendo ruído entre os envolvidos), obsolescência limitada (capacidade de controlar a utilidade de componentes e tecnologia), e continuidade da experiência de uso e casos de uso (capacidade dos produtos evoluírem em resposta a mudança em tecnologias);
- **Criação de Nicho ou Inovação:** é a habilidade do ecossistema aumentar significativamente a diversidade de envolvidos ao longo do tempo. É medida por meio do crescimento/decrescimento na variedade de produtos da companhia e crescimento/decrescimento na variedade técnica (conhecimento dos desenvolvedores) e de produto (criação de valor).

3.3.1 Sáude de ECOS

O conceito de saúde de ecossistemas começou a ser abordado por Iansiti e Levien (2004) como um modo de medir o desempenho de um ecossistema de negócios. A literatura técnica faz analogias com Ecossistemas Naturais e aproveita termos dos Ecossistemas de Negócios. Em ecossistemas naturais, um ecossistema saudável é definido como estável e sustentável, mantendo sua organização e autonomia em relação ao tempo e à sua capacidade de resistir a situações de estresse [Schaeffer *et al.* 1988].

Do ponto de vista do desenvolvedor nas comunidades que atuam com software de código aberto, o conceito de saúde de um projeto é definido como a capacidade de sobrevivência, ou seja, a capacidade do projeto sobreviver com o passar do tempo

[Manikas 2016]. Em [Mustofa *et al.* 2007], os autores identificam três métricas que afetam a saúde de um projeto de código aberto:

- **Ânimo da comunidade de desenvolvedores:** o projeto deve ser atrativo para novos desenvolvedores e deve manter os desenvolvedores atuais, melhorando a sua motivação (estimulação intelectual, melhoria de habilidades, acesso ao código fonte e necessidades dos usuários);
- **Ânimo da comunidade de usuários:** os usuários têm um importante papel na evolução do projeto ao reportar defeitos e ao requisitar novas funcionalidades. Uma comunidade grande e ativa de usuários indica que o software produzido é de qualidade;
- **Qualidade do produto:** um produto competitivo com outros produtos comerciais, tanto no uso quanto na qualidade, quando atrai usuários e desenvolvedores, aumenta a atividade no projeto e melhora a capacidade de sobrevivência.

Em uma analogia com a Ecologia, especificamente no estudo da ciência dos ecossistemas, Begon *et al.* (2007) apontam que um dos motivos pelos quais é necessário estudar ECOS é para entender os processos que dão suporte aos produtos que são construídos e consumidos e, essencialmente, aumentam a produção. Além de disseminar boas práticas e padrões arquiteturais para garantir a integração de produtos no ECOS, a organização central deve rever seus processos com o objetivo de melhorar a experiência dos contribuidores do ECOS e estabelecer controles adequados aos diferentes tipos de contribuidores. Wazlawick (2013) discorre sobre as vantagens de analisar o desenvolvimento de algum software a partir da perspectiva de processos:

- **Reduzir o tempo de treinamento:** é mais fácil encaixar novos indivíduos em uma equipe quando os processos são bem definidos e documentados;
- **Uniformizar a construção de produtos:** uma equipe com um processo bem definido tende a construir um produto mais bem definido se comparada a uma equipe sem processo;
- **Capitalizar experiências:** se um desenvolvedor faz algo de forma diferente a partir de sua criatividade, isto pode ser incorporado nos processos como uma melhoria.

3.3.2 Processos em ECOS

A partir da análise das fases e dos elementos que compõem um ECOS, Fontão (2016) identificou três processos (Figura 3.3): (1) *Orquestração* – que surge da interação entre a organização central e o evangelista, uma vez que este se baseia nas diretrizes do ECOS para executar atividades de suporte; (2) *Suporte* – que estrutura o fluxo de trabalho entre o evangelista e o desenvolvedor que é orientado para seguir as diretrizes do ecossistema na produção de aplicações móveis; e (3) *Desenvolvimento* – atividades para a construção da aplicação móvel que será adquirida e avaliada pelos usuários do ECOS.

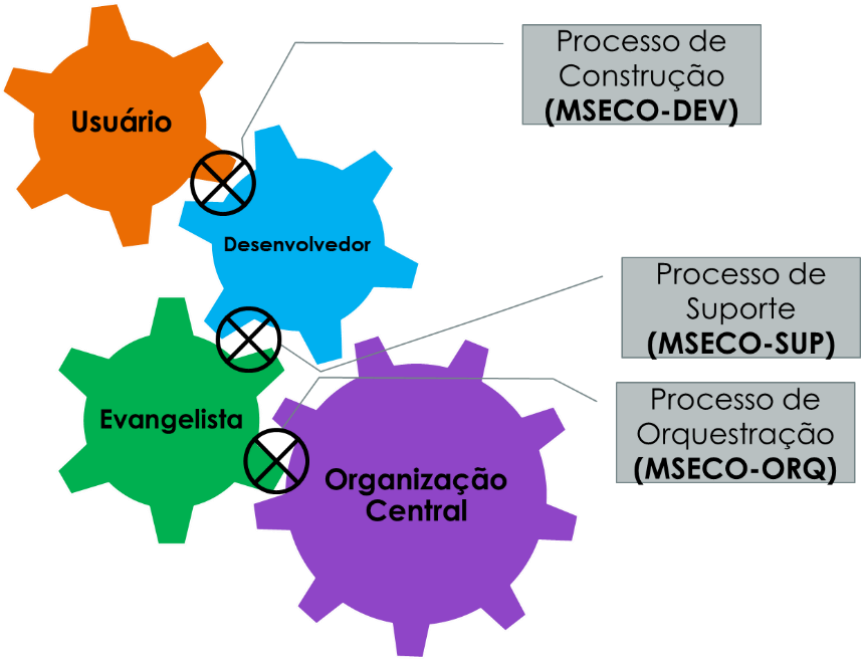

Figura 3.3. Relação entre os processos de Orquestração, Suporte e Construção

O objetivo do processo de orquestração é preparar, gerenciar e coordenar alguns elementos (e.g., organização central, desenvolvedor, evangelista e a contribuição, no caso, a aplicação móvel) [Jansen *et al.* 2009] e alguns de seus relacionamentos em ECOS. Além disso, visa fornecer diretrizes e guias necessários às contribuições para o ecossistema com o objetivo de manter os indicadores de saúde de ECOS: robustez, produtividade e criação de nicho.

O objetivo do processo de suporte é prover o relacionamento entre a organização central e os desenvolvedores. Para isso, o ECOS dispõe de um elemento responsável por esta ligação: o evangelista. Por fim, o objetivo do processo de construção é que o desenvolvedor idealize, planeje e construa uma aplicação móvel que será adicionada à loja de aplicações. Para isso, o desenvolvedor poderá utilizar artefatos gerados tanto no processo de orquestração como no processo de suporte. Dessa forma, ele pode contribuir com a produtividade e a criação de nicho do ECOS.

O evangelista precisa motivar e engajar outros desenvolvedores externos e apoiá-los na expansão da comunidade e, consequentemente, na disponibilização de uma maior quantidade de aplicações móveis para os usuários. A organização central precisa empreender esforços na disponibilização de uma estrutura de base para que o ECOS possa se expandir por meio de diretrizes para atrair novos desenvolvedores e criar aplicações móveis, além do fornecimento de ferramentas e suporte à comunidade. O desenvolvedor precisa criar e disponibilizar contribuições que atendam da melhor forma os nichos existentes de usuários e também obter visibilidade dentro do ECOS.

Na Figura 3.4, apresentamos um processo que a organização central utiliza para realizar a governança dos desenvolvedores dentro de um ECOS. A sua descrição é realizada a seguir:

O papel responsável por este processo é:

Papel:	**Proprietário da Plataforma (ou *Organização Central*)**
Descrição:	Responsável pelo provimento de padrões e práticas e pela identificação de oportunidades.

Os papéis participantes neste processo são:

Papel:	**Equipe de Desenvolvimento da Organização Central**
Descrição:	Responsável pelo desenvolvimento da plataforma que faz parte do ECOS.
Papel:	**Equipe de Design e UX (Experiência de Usuário) da Organização Central**
Descrição:	Responsável pela definição de padrões de interface da plataforma.
Papel:	**Equipe de Marketing de Produto**
Descrição:	Responsável pela divulgação da plataforma e dos produtos que são construídos a partir dela.
Papel:	**Equipe de Marketing de Desenvolvedores**
Descrição:	Responsável pela divulgação dos desenvolvedores e das contribuições deles dentro do ECOS.
Papel:	**Equipe de Estratégia de Tecnologia**
Descrição:	Responsável pela proposta, definição e análise das tecnologias pertencentes à organização central.
Papel:	**Equipe de Validação de Produto da Organização Central**
Descrição:	Responsável por definir requisitos de qualidade para aceitação de contribuições para o ECOS.
Papel:	**Equipe Jurídica da Organização Central**
Descrição:	Responsável pela parte legal que envolvem os elementos e a organização central no ECOS.
Papel:	**Equipe de Evangelismo**
Descrição:	Responsável pela ligação entre a organização central e o desenvolvedor. Faz parte de uma comunidade de especialistas da organização.

Os artefatos do processo de orquestração são:

Artefato:	**Especificação da Plataforma**
Descrição:	Descreve como a plataforma está organizada internamente e seu nível de abertura para os desenvolvedores externos. No caso de ECOS, deve descrever os dispositivos móveis envolvidos e suas características, linguagem de programação que pode ser utilizada, tecnologias suportadas, APIs, SDKs e o posicionamento no mercado e com os usuários.

Artefato:	**Guias de Design e Interface de Usuário**
Descrição:	Descreve quais são os padrões de interface de usuário dos dispositivos móveis e da interação entre a aplicação e o usuário (padrões de tela e elementos visuais, componentes, animações, gestos de interação, mensagens).

Artefato:	**Aplicações Móveis de Referência**
Descrição:	Um conjunto de aplicações móveis com os padrões de interface, componentes e APIs e SDKs, que podem servir como modelo para o desenvolvedor iniciar um novo projeto.

Artefato:	**Guias e Ferramentas de Marketing**
Descrição:	Descreve como otimizar o acesso às aplicações dentro da loja e lista ferramentas e orientações para divulgação logo após a publicação.

Artefato:	**Ferramenta de Desenvolvimento**
Descrição:	Permite a criação da aplicação utilizando as linguagens, APIs e SDKs disponíveis para a plataforma. Gera ainda o binário da aplicação, pacote que poderá ser disponibilizado na loja.

Artefato:	**Central do Desenvolvedor**
Descrição:	Permite o acesso a *links* relacionados a: plataforma, ferramentas, documentos, suporte, fóruns, *wikis* e controle de publicação de aplicações da loja.

Artefato:	**Loja de Aplicações Móveis**
Descrição:	É um ambiente democrático para acesso às aplicações desenvolvidas e que poderão ser adquiridas pelos usuários.

Artefato:	**Critérios da Loja**
Descrição:	Descrevem os critérios de aceitação para qualidade de aplicações que poderão compor a loja. Esses critérios se baseiam em requisitos funcionais.

Artefato:	**Diretrizes do ECOS**
Descrição:	Sintetizar e alinhar todos os outros artefatos gerados nesse processo para um conjunto coerente de diretrizes para atuar dentro de um ECOS.

Artefato:	**Pacote de Políticas de Incentivos**
Descrição:	Descreve um conjunto de políticas para executar as estratégias definidas pela organização central com o objetivo de motivar e engajar os desenvolvedores a partir das contribuições (aplicação ou conteúdo para compartilhar conhecimento) para o ECOS, por meio de metas estabelecidas pela própria organização central.

Nesse processo, as atividades têm como objetivo principal gerar a base para o funcionamento do ECOS. Por isso, o responsável por todas as atividades é a organização central. As atividades contam ainda com a participação de equipes que fazem parte da organização (e que podem contribuir na geração dos artefatos): de desenvolvimento de software, de design, de marketing de produto, de marketing de desenvolvedores, jurídica, de validação de produto, de evangelistas e de desenvolvimento do sistema operacional da plataforma do ecossistema.

Os documentos gerados ao final deste processo formam a base do ecossistema, que compõem, ao final do processo, o artefato *Diretrizes do ECOS*, que servirá para outros processos. Esses documentos são: *Especificação da Plataforma*, *Guias de Design e Interface*, *Guias e Ferramentas de Marketing*, *Ferramentas de Desenvolvimento*, *Central do Desenvolvedor*, *Loja de Aplicações* e *Critérios da Loja.* Na Figura 3.4, o Processo de Orquestração é apresentado.

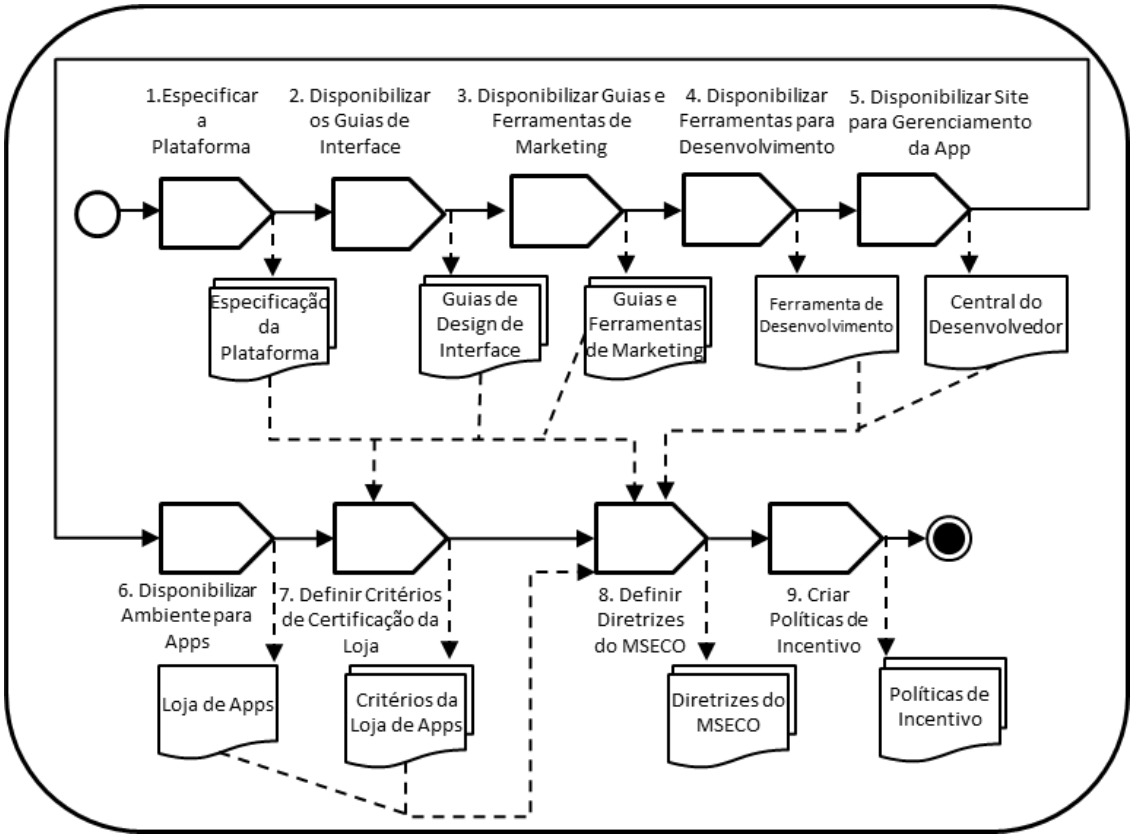

Figura 3.4. Processo de orquestração de desenvolvedores

As atividades que estão relacionadas à definição de estratégias têm como objetivo fornecer o suporte para funcionamento do ecossistema, inclusive as ameaças ou oportunidades que possam afetar de alguma forma o ecossistema. Essa atividade é de responsabilidade da organização central, podendo contar com a participação de evangelistas e de outros setores da organização. Abaixo, são apresentadas as atividades que compõem o processo de governança:

Atividade:	**1 Especificar a Plataforma**
Descrição:	Especificar detalhes das plataformas apoiadas pelo ecossistema é uma atividade de base para o bom funcionamento do ECOS. Seu objetivo é descrever dispositivos suportados e suas características, tecnologias, linguagens de desenvolvimento, APIs, SDKs, posicionamento no mercado e com os usuários.
Critérios de Entrada:	Não se aplica.
Critérios de Saída:	Especificação da plataforma consolidada e aprovada.

Responsável:	*Organização Central*.
Participantes:	Equipe de Desenvolvimento da Organização Central e Equipe de Estratégia de Tecnologia.
Artefatos Requeridos:	Não se aplica.
Artefatos Produzidos:	Especificação da plataforma (dispositivos suportados e todas suas características, tecnologias, linguagens de desenvolvimento, APIs, SDKs, posicionamento no mercado e com os usuários).
Recomendação:	A especificação deve ter versões tanto para acesso interno da organização (com informações mais restritas), como para acesso dos desenvolvedores externos, de forma a não expor informações confidenciais.

Atividade:	**2 Disponibilizar Guias de Design e Interface de Usuário da Plataforma**
Descrição:	A interface de usuário de uma plataforma é sua identidade. É importante divulgar guias que ajudem os desenvolvedores a criar e desenvolver soluções que respeitem os padrões de design e interface da plataforma do ECOS. Essa atividade tem como objetivo disponibilizar a base necessária para manter a boa interação com o usuário e o padrão da plataforma do ECOS.
Critérios de Entrada:	Ter-se a Especificação da Plataforma pronta ou atualizada.
Critérios de Saída:	Documentos com especificação de interface e interação da plataforma criados e aprovados.
Responsável:	*Organização Central*.
Participantes:	Equipe de Design e Equipe de UX (Experiência do Usuário) da Organização Central.
Artefatos Requeridos:	Especificação da Plataforma.
Artefatos Produzidos:	Guias de Design e Interface de Usuário, e Aplicações Móveis de Referência.
Recomendação:	Deve-se levar em consideração as limitações e potencialidades da plataforma na execução desta atividade.

Atividade:	**3 Disponibilizar Guias e Ferramentas de Marketing**
Descrição:	Toda aplicação precisa de um trabalho para ganhar exposição e visibilidade. É necessário que seja criado um conjunto de guias para rapidamente entregar a aplicação ao mercado e dar o reconhecimento que ela necessita, assim como ao seu desenvolvedor. É necessário que sejam disponibilizadas ferramentas para ajudar no reconhecimento da aplicação e desenvolvedor e também na entrega da aplicação ao mercado.
Critérios de Entrada:	Para essa atividade iniciar, a especificação da plataforma deve ter sido concluída, assim como os guias de design de interface da plataforma devem estar prontos.
Critérios de Saída:	Ferramentas e documentos que ajudem na otimização da aplicação quanto a marketing devem estar concluídos e aprovados.
Responsável:	*Organização Central*.
Participantes:	Equipe de Marketing de Produto, Equipe de Marketing de Desenvolvedores.
Artefatos Requeridos:	Especificação da Plataforma e Guias de Design e Interface de Usuário.
Artefatos Produzidos:	Guias e Ferramentas de Marketing.

Recomendação:	É importante que os guias de marketing levem consideração: •Marketing dentro da Loja de Aplicações •Marketing fora da Loja de Aplicações •Marketing dentro do dispositivo

Atividade:	**4 Disponibilizar Ferramenta para Desenvolvimento**
Descrição:	Essa atividade tem como objetivo disponibilizar as ferramentas principais que são necessárias para a construção de aplicações para a plataforma.
Critérios de Entrada:	Para essa atividade iniciar é necessário que a plataforma tenha sido especificada, os guias de design e interface de usuário estejam prontos e aprovados.
Critérios de Saída:	Ferramenta deve permitir a geração de um pacote binário da aplicação, criação de projetos, disponibilização de modelos de projetos de aplicações e depuração de aplicações.
Responsável:	*Organização Central*.
Participantes:	Equipe de Desenvolvimento da Organização Central e Equipe de Experiência do Usuário.
Artefatos Requeridos:	Aplicações Móveis de Referência, Especificação da Plataforma, Guias de Design e Interface de Usuário.
Artefatos Produzidos:	Ferramenta de Desenvolvimento.
Recomendação:	Uma ferramenta de desenvolvimento deve permitir: •O acesso a guias da plataforma; •Desenvolvimento de uma aplicação completa; •Desenvolvimento de testes, no mínimo em nível de unidade; •Depuração do código da aplicação; •Geração de pacote tanto em modo de depuração como em modo de entrega; •Interagir com emuladores de dispositivos da plataforma.

Atividade:	**5 Disponibilizar Site para Gerenciamento da Aplicação Móvel**
Descrição:	Com esta atividade, pretende-se disponibilizar um site que permita o envio e o gerenciamento das aplicações criadas pelo desenvolvedor.
Critérios de Entrada:	Para essa atividade iniciar, é necessário que a plataforma tenha sido especificada, os guias de design de interface estejam prontos e aprovados e que, além disso, exista uma ferramenta que possa gerar um pacote com o conteúdo da aplicação.
Critérios de Saída:	Site que permita a submissão de uma nova aplicação, edição, remoção e atualização. Deve-se ainda permitir o acesso a relatórios de *downloads* (total e por cada aplicação) e de revisões de usuários (estrelas e comentários).
Responsável:	*Organização Central*.
Participantes:	Equipe de Desenvolvimento da Organização Central e Equipe de Marketing de Produto.
Artefatos Requeridos:	Especificação da Plataforma.
Artefatos Produzidos:	Central do Desenvolvedor.
Recomendação:	Deve-se levar em consideração que o desenvolvedor, nesse momento, é um publicador de conteúdo (a aplicação), logo, o site deve conter todas as informações e/ou *links* que possam juda-lo durante o processo de submissão

	e gerenciamento de aplicações.

Atividade:	**6 Disponibilizar Ambiente para Aplicações Móveis**
Descrição:	Atividade que tem como objetivo a disponibilização de um ambiente que represente: para o desenvolvedor, um local de distribuição de aplicações; para o usuário, um local de aquisição de aplicações; e para o ecossistema, um local democrático de acesso a aplicações pelos elementos do ECOS.
Critérios de Entrada:	Deve existir um portal de gerenciamento da aplicação, que foi desenvolvido sobre a especificação da plataforma do ECOS e dos guias de design de interface.
Critérios de Saída:	Este ambiente deve estar disponível como um *site* e também estar embarcado nos dispositivos da plataforma do ECOS. Deve permitir que usuários possam adquirir e revisar aplicações.
Responsável:	*Organização Central*.
Participantes:	Equipe de Desenvolvimento da Organização Central.
Artefatos Requeridos:	Não se aplica.
Artefatos Produzidos:	Loja de aplicações.
Recomendação:	O ambiente deve ser democrático dos seguintes pontos de vista: •*Desenvolvedor/publicador:* a partir da participação de tipos variados de desenvolvedores (individuais ou companhias), oferecer opções para disponibilização de uma aplicação gratuitamente, de forma paga ou com pagamento dentro da aplicação; •*Usuário:* ambiente embarcado no celular como uma loja que permita a busca, indicações de aplicações, aquisição de aplicação e revisão da aplicação. O usuário pode adquirir aplicações gratuitas ou pagas e aplicações em testes.

Atividade:	**7 Definir Critérios de Certificação de Qualidade para Aceitação da Loja**
Descrição:	Os critérios de certificação de qualidade de uma loja de aplicações têm como objetivo garantir que requisitos em conformidade com a plataforma e com a interface de usuário sejam atendidos. Essa atividade tem como objetivo consolidar um guia com os critérios para a loja do ECOS que devem ser atendidos pela aplicação construída por um desenvolvedor.
Critérios de Entrada:	Especificação da plataforma e guias de design de interface devem estar prontos e aprovados.
Critérios de Saída:	Os critérios devem considerar especificação da plataforma, guias de marketing, legislação referente a conteúdo local do país e guias de interface da plataforma.
Responsável:	*Organização Central*.
Participantes:	Equipe de Validação de Produto da Organização Central e Equipe de Desenvolvimento da Organização Central.
Artefatos Requeridos:	Especificação da Plataforma, Guias e Ferramentas de Marketing e Guias de Design e Interface de Usuário.
Artefatos Produzidos:	Critérios da Loja.

Recomendação:	Os critérios devem levar em consideração os dispositivos que compõem a plataforma, a especificação tanto de hardware quanto de software, os guias de design de interface e legislação própria de cada país para questões de acesso a conteúdo digital.

Atividade:	**8 Definir Diretrizes do ECOS**
Descrição:	O objetivo desta atividade é condensar todos os artefatos gerados nas atividades anteriores de forma a criar um documento e/ou local para acesso a todos os artefatos. A Equipe de Marketing de Desenvolvedores reunirá em um documento, que pode ser um *website*, todos os artefatos gerados nas fases anteriores provendo rápido e fácil acesso aos artefatos.
Critérios de Entrada:	Todos os artefatos devem estar consolidados e atualizados.
Critérios de Saída:	Documento que possa ser acessado tanto por desenvolvedores internos quanto externos do ECOS.
Responsável:	*Organização Central*.
Participantes:	Equipe de Marketing de Desenvolvedores e Equipe de Estratégia de Tecnologia.
Artefatos Requeridos:	Especificação da Plataforma, Guias de Design e Interface de Usuário, Guias e Ferramentas de Marketing, Ferramenta de Desenvolvimento, Central do desenvolvedor.
Artefatos Produzidos:	Documento com diretrizes do ECOS contendo as ligações entre os artefatos gerados em atividades anteriores.
Recomendação:	As diretrizes devem fazer uma ligação coerente entre todos os artefatos gerados nas atividades anteriores, deve ainda estar em um local de fácil acesso para qualquer contribuidor que desejar visualizar.

Atividade:	**9 Criar Políticas de Incentivo**
Descrição:	Definir, planejar e executar estratégias que envolvam a motivação e reconhecimento dos desenvolvedores e de suas contribuições ajudam a manter um ecossistema produtivo e a sua arquitetura estável. Essa atividade define políticas para incentivar a participação e engajamento de desenvolvedores dentro de ECOS.
Critérios de Entrada:	Deve-se utilizar os guias de marketing e os direcionamentos do ECOS presentes no documento de diretrizes.
Critérios de Saída:	Políticas detalhadas (público, metas e formas de reconhecimento) e validadas.
Responsável:	*Organização Central*.
Participantes:	Equipe de Evangelistas, Equipe Jurídica da Organização Central e Equipe de Marketing de Desenvolvedores.
Artefatos Requeridos:	Diretrizes do ECOS.
Artefatos Produzidos:	Pacote de Políticas de Incentivo.
Recomendação:	As políticas devem: •Reconhecer a contribuição; •Reconhecer o desenvolvedor;

	•Facilitar o engajamento do desenvolvedor; •Definir regras e envolvidos, inclusive, os responsáveis pelo suporte ao desenvolvedor, como os evangelistas; •Validadas pelo jurídico da organização central a partir da definição de um regulamento.

Neste cenário, há três principais categorias de mecanismos de governança: 1) Criação de valor – gerar e distribuir valor; 2) Coordenação – manter a consistência e integração de atividades, relacionamentos e estruturas do ecossistema; e 3) Controle e abertura organizacional – capturar a atenção entre modelos abertos e fechados.

Baars e Jansen (2012) afirmam que a governança pode ajudar uma empresa a atingir seus objetivos, fazer melhor uso dos recursos disponíveis e direcionar a um aumento na renda e na redução de riscos. A governança de ecossistema requer um equilíbrio de controle pela organização central e de autonomia entre os desenvolvedores que são externos à organização [Tiwana *et al.* 2010]. Entretanto, uma vez que este é um campo relativamente novo, muitas organizações podem não saber como gerenciar efetivamente seus ecossistemas, ou como iniciar um ecossistema. Não existe uma formalização adequada para a governança do ecossistema e há muitos desafios a serem superados pelas organizações [Jansen e Finkelstein 2009], por exemplo, a atração e o envolvimento dos desenvolvedores. Existe também a necessidade de: um vocabulário comum na governança dos ECOS, orientação prática e o entendimento da governança de desenvolvedores.

Jansen *et al.* (2012) ainda indicam que é necessário buscar a clareza da governança dentro do ecossistema. Para isso, os autores apontam partes da governança que ajudam a estabelecer esta clareza, que são:

- **Clareza do ecossistema:** há questões essenciais para o sucesso de um ecossistema, se um ecossistema não possuir uma estrutura que seja clara para os elementos que o compõe não é possível ter uma estratégia de governança clara;
- **Clareza da estratégia de governança:** as organizações podem estabelecer regras, procedimentos, protocolos e processos formalizados, o que pode ajudar no controle do ecossistema. Isto ajuda a estabelecer uma estratégia de governança clara;
- **Responsabilidade:** os elementos que compõem o ecossistema são responsáveis por ele. Se não houver execução correta das responsabilidades de cada elemento, uma estratégia de governança não poderá ser garantida;
- **Medição:** acontece com o objetivo de determinar o benefício para o ecossistema a partir da utilização de uma estratégia de governança, utilizando-se de indicadores para analisar o estado atual do ecossistema e prospectar o estado futuro;
- **Compartilhamento de conhecimento:** é um aspecto importante do núcleo de negócios de um ecossistema como parte de uma estratégia de governança.

3.4. Governança de Desenvolvedores em ECOS

A governança de desenvolvedores, no contexto de ECOS, pode ser definida como um conjunto de mecanismos (e.g., abordagens, estratégias, práticas, algoritmos, ferramentas de análise) para apoiar relações sinérgicas (“ganha-ganha”) entre uma comunidade próspera de desenvolvedores e uma organização central com o objetivo de garantir e monitorar o bem-estar social e econômico dos desenvolvedores. Neste cenário, as comunidades de desenvolvedores emergem para apoiar a sustentabilidade do ecossistema, ou seja, a capacidade de um ecossistema aumentar ou manter sua comunidade de desenvolvedores durante o tempo e sobreviver a mudanças.

A perspectiva da análise das expectativas e experiências do desenvolvedor durante a entrada, engajamento e abandono do ecossistema pode ser estudada por meio de métodos e indicadores que capturem a Experiência do Desenvolvedor (DX, do inglês *Developer eXperience*). A DX ajuda na análise do impacto no desenvolvimento de aplicações (popularmente conhecidas como *apps*) a partir das expectativas, emoções e percepções do desenvolvedor relacionadas a todos os tipos de artefatos que ele pode encontrar como parte do seu envolvimento no ecossistema [Fagerholm 2015] visando um ecossistema que mantenha a sustentabilidade e diversidade [Dhungana *et al.* 2010]. Neste cenário, a governança consiste na busca pelo equilíbrio entre os objetivos da organização central e as expectativas do desenvolvedor (Figura 3.5).

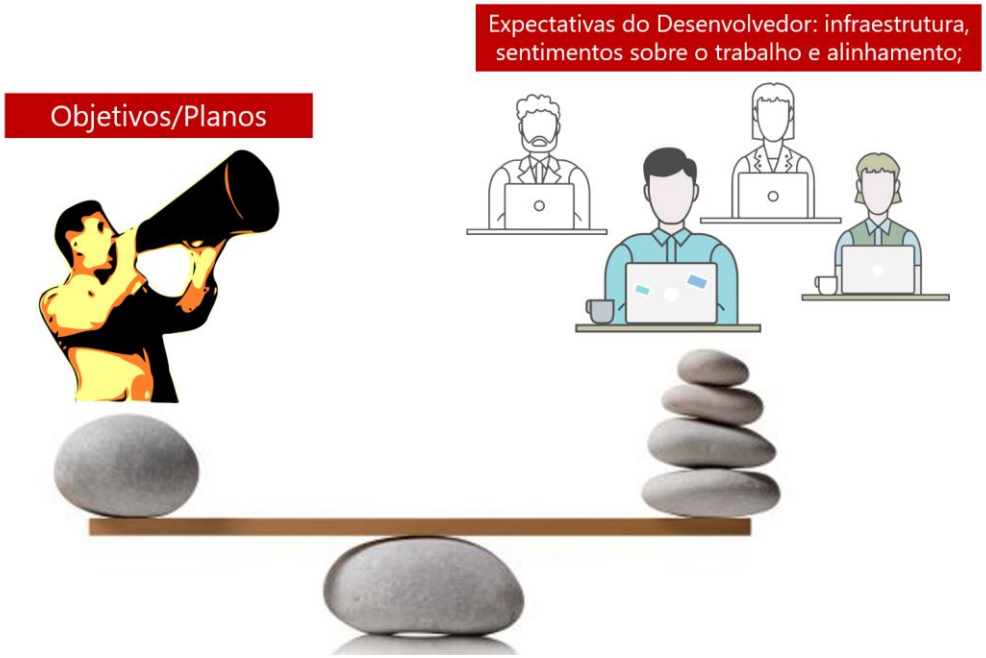

Figura 3.5. Equilíbrio entre objetivos da organização central e a experiência do desenvolvedor

Segundo Falgerholm e Münch (2012), a DX consiste nas experiências relativas a todos os tipos de artefatos e atividades que um desenvolvedor possa encontrar como parte de sua participação no desenvolvimento de software. Pode ser analisada a partir de três fontes:

1. **Infraestrutura de Desenvolvimento:** relacionada à forma como os desenvolvedores percebem a infraestrutura de desenvolvimento. Alguns tópicos

precisam ser estudados, tais como: ferramentas de gerenciamento, linguagens de programação, bibliotecas, plataformas, métodos, técnicas, habilidades e procedimentos;

2. **Percepções sobre o Trabalho:** analisa como os desenvolvedores se sentem sobre o seu trabalho, ou seja, sobre o que produzem. Nesta fonte, a análise se volta para o respeito, reconhecimento, interação social, time e afeto;
3. **Senso de Contribuição:** como os desenvolvedores veem o valor de suas contribuições. Para entender esta fonte de DX, analisa-se os seguintes fatores: planos, objetivos, intenções, alinhamento e engajamento.

Quando passamos a analisar DX a partir do desenvolvedor e o alinhamento com a organização central de um ECOS e suas respectivas funções, temos os pontos:

- Em relação à fonte *infraestrutura de desenvolvimento*, a organização central precisa investir esforços para prover aos desenvolvedores um *framework* com o objetivo de expandir as fronteiras do ECOS. Isso pode ser alcançado com diretrizes para atrair novos desenvolvedores e criar *apps*, provendo ferramentas e suporte ao desenvolvedor;
- Do ponto de vista da fonte *percepções sobre o trabalho*, o desenvolvedor precisa criar e entregar *apps* que alcancem os melhores nichos de usuários e que ganhem visibilidade com base em sua qualidade. Uma *app* poderá ser adquirida por usuários e contribuir para as metas da organização central. Neste contexto, número de *apps*, médias de avaliação das *apps* por usuários e número de desenvolvedores que publicam *apps* alimentam as métricas de ECOS.
- Sobre a fonte *senso de contribuição*, a organização central tem metas que podem ser atingidas e potencializadas por desenvolvedores externos: (i) o número de *apps*, (ii) a quantidade de renda gerada pela venda de *apps*, e (iii) o número de *apps* publicadas na loja. O alinhamento dos planos da organização central com os planos dos desenvolvedores pode ser utilizado para atender a demanda por *apps* da sociedade uma vez que a organização central, somente com sua estrutura interna, pode dificilmente responder a esta demanda [Fontão *et al.* 2016].

Em uma analogia com a ecologia, os indivíduos (desenvolvedores externos) possuem características que são únicas. Uma comunidade de desenvolvedores externos é formada por indivíduos com características diferentes [Begon *et al.* 2007]. Explicamos o comportamento de uma comunidade a partir do comportamento dos indivíduos que a compõem. Uma forma de analisar o comportamento e interação dos desenvolvedores em ECOS é minerando os dados disponíveis em repositórios de software. Isto permite "escutar" a voz do desenvolvedor.

3.5. Mineração de Repositórios

Não há comunicação direta entre grandes organizações que mantêm plataformas móveis (e.g., Apple, Google e Microsoft) e desenvolvedores terceirizados para resolver questões técnicas que surgem no design e desenvolvimento de contribuições de desenvolvedores em ECOS. Neste cenário, as organizações podem não saber como definir e evoluir

estratégias para governar seus desenvolvedores no sentido de atingir suas metas organizacionais. Essas organizações usam uma infraestrutura para dar suporte a desenvolvedores, por exemplo, um repositório de perguntas e respostas (Q&A, do inglês *Questions and Answers*). As interações entre os desenvolvedores nesses portais alimentam um repositório de Q&A que pode servir como um mecanismo para entender e definir estratégias para dar suporte aos desenvolvedores.

O desenvolvedor é um ator essencial para sustentar as contribuições do ECOS, como aplicativos e documentação técnica [Fontão *et al.* 2016]. Tais contribuições são normalmente armazenadas em um repositório oficial ECOS interno, como repositórios Android e Apple Developers. Além disso, existem links entre ECOS e repositórios externos, como código (e.g., GitHub) [Casalnuovo *et al.* 2015] e repositórios de Q&A. Como exemplo de um repositório de perguntas e respostas externo, o Stack Overflow (SO) tem um conjunto de perguntas/respostas técnicas que surgem do uso de APIs, SDKs e ferramentas de desenvolvimento [Lin e Serebrenik 2016].

Esses repositórios externos ajudam a manter a interação entre os desenvolvedores em uma plataforma comum, resultando em um conjunto de contribuições e influenciando direta ou indiretamente o ecossistema como um todo. [Santos e Werner 2012]. Portanto, um repositório de perguntas e respostas como o SO tem comunicações arquivadas entre os desenvolvedores de ecossistema e pode ser usado para investigar alguns aspectos do ECOS, por exemplo, engajamento do desenvolvedor e fragmentos de código.

A partir de uma dimensão técnica, uma grande quantidade de dados está disponível em repositórios de software. Esses dados são estáveis e não influenciados por pesquisadores [Shull *et al.* 2008]. Neste cenário, um método usado para conduzir estudos experimentais no campo de ecossistemas é a mineração de repositórios de software [Farias *et al.* 2016]. Este método pode ajudar na definição e evolução de estratégias para governar desenvolvedores em ECOS.

Repositórios de software podem ser fontes valiosas de informação uma vez que eles contêm (ou podem permitir a extração) de informação sobre aspectos sociais, técnicos e de negócios de um projeto, como acontece nas fontes da comunicação entre desenvolvedores [Genc-Nayebi e Abran 2016]. A área de Mineração de Repositórios de Software (MRS, do inglês *Mining Software Repositories*) foca em descobrir informação útil sobre o software por meio da extração e análise de dados de diferentes repositórios de software [Hassan 2008]. Abordagens de MRS têm sido usadas para diferentes objetivos, por exemplo, análises de contribuições e comportamento do desenvolvedor. Neste cenário, podemos dar como exemplos de tipos de repositórios software: Q&A, Código e Projeto, e Desafios.

Repositórios de Q&A são plataformas colaborativas web que têm como objetivo apoiar o compartilhamento de conhecimento ao permitir que usuários postem e respondam questões. Esses repositórios não são apenas uma plataforma para especialistas compartilharem o conhecimento e serem identificados como membros do ecossistema, mas também para ajudarem desenvolvedores iniciantes na resolução de seus problemas eficientemente [Bhat 2014]. O SO é um exemplo deste tipo de repositório [Shah *et al.* 2014].

SO é um portal de Q&A direcionado para a comunidade de desenvolvedores que postam questões e respostas (aproximadamente 14 milhões de questões e 19 milhões de respostas) relacionadas à programação de computadores. Tanto as questões quanto as respostas podem receber votos dos usuários (contra/a favor de). Tais votos se tornam pontos de reputação que possibilitam aos desenvolvedores conseguirem alguns privilégios como a liberação de restrições na criação de uma publicação e edição de perguntas e respostas de outros usuários. Outro mecanismo de privilégio envolve a atribuição de distintivos, ou seja, reconhecimento de desenvolvedores ao usarem o SO. Os desenvolvedores podem obter distintivos de várias atividades, por exemplo, um desenvolvedor pode receber um distintivo se ele fez uma pergunta que atingiu mais de mil visitas.

Zagalsky *et al.* (2016) identificaram que os desenvolvedores usam o SO por vários motivos: (a) a capacidade de obter reconhecimento de pares; (b) interface rica e amigável; (c) as respostas são diretas ao ponto; (d) as perguntas geralmente são respondidas mais rapidamente do que outros fóruns; e (e) é fácil procurar por perguntas e respostas anteriores. Neste trabalho, como os estudos anteriores baseados na mineração do SO, usamos uma abordagem (Figura 3.6) não supervisionada para extrair tópicos de perguntas de SO. Nossa metodologia é composta por quatro etapas descritas a seguir.

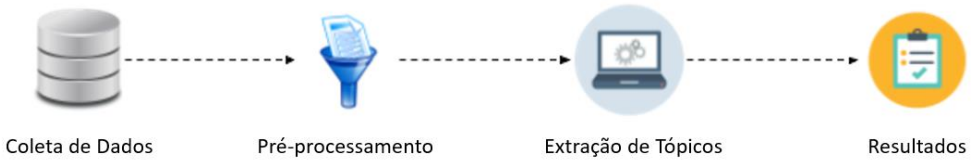

Figura 3.6. Etapas para Mineração e Tratamento dos Dados

Coleta de Dados: para este trabalho, utilizamos a base de dados fornecida pelo SO que está disponível publicamente em formato XML licenciado sobre a licença CC BY-as, que é uma licença pública que permite a distribuição gratuita de uma obra protegida por direitos autorais. Para os propósitos deste trabalho, nós utilizamos o arquivo posts.xml que contêm: o texto das postagens, quantidade de visualizações e respostas, *tags*, contagem de favoritos e data de criação, por exemplo.

Pré-processamento: pré-processamos o conteúdo textual (*Body*) dos *posts* extraídos em três etapas. Primeiro, descartamos qualquer fragmento de código presente nos *posts* (ou seja, entre *tags* HTML <code>), porque a sintaxe do código fonte (por exemplo, instruções “if” e *loops* “for”) introduz ruído na fase de análise. Em seguida, removemos todas as *tags* HTML (por exemplo, <p> e <a href="...">), pois esse não é o foco de nossa análise. Terceiro, removemos as palavras comuns em inglês, como "a", “the” e “is”, que não ajudam a criar tópicos significativos. Usamos o Spark como uma estrutura que suporta a análise de *big data*. O Spark[1] é um *framework* que ajuda no processamento de grande volume de dados com uma variedade de conjuntos de dados diversos (e.g., texto, grafos etc.) e de diferentes origens (*batch* ou *streaming* de dados

[1] http://spark.apache.org/

em tempo real) e foi desenvolvido em 2009 pelo AMPLab da Universidade da Califórnia com foco em velocidade e análises sofisticadas.

O procedimento de mineração de dados foi automatizado desde a construção do conjunto de dados até a análise do tópico. O pré-processamento é o passo responsável pela eliminação de termos não representativos (e.g., palavras irrelevantes, URLs, *emoticons* e *hashtags*) na coleta e o processo de extração de recursos. Neste contexto, usamos a ferramenta NTLTK para eliminar termos não representativos e, para fazer o processo de extração de características, usamos a abordagem *bag-of-words* com o TF-IDF (do inglês, *Term Frequency-Inverse Document Frequency*), onde eliminamos termos com frequência menor que cinco.

A abordagem *bag-of-words* consiste na transformação dos dados que não estão estruturados em um formato estruturado, especificamente uma tabela atributo-valor. O TF-IDF [Forman 2008] consiste em uma medida estatística para indicar a importância de uma palavra dentro de um documento em relação ao corpo do documento. O valor de TF-IDF aumenta à medida que cresce o número de ocorrências de uma palavra dentro do documento.

Extração de Tópicos: utilizamos *Latent Dirichlet allocation* (LDA) [Krestel *et al.* 2009], um modelo de tópico estatístico usado para recuperar automaticamente tópicos em vários domínios de um corpus de documentos de texto. Escolhemos o LDA porque ele é capaz de modelar tópicos em grandes volumes; no nosso caso, o corpo de questões dos desenvolvedores relacionadas aos ecossistemas.

Para a extração das informações do SO (Figura 3.7), podemos usar dados como: o título, corpo, pontuação da questão, número de visualizações, quantidade de respostas e *tags* utilizadas para categorizar a questão.

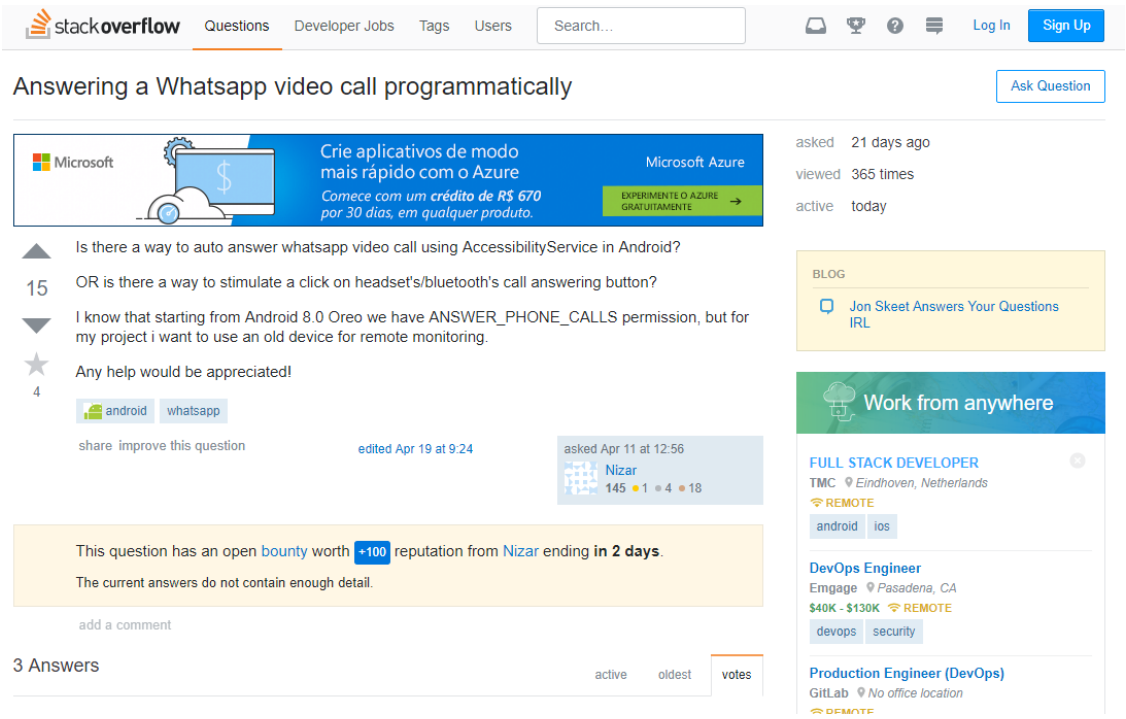

Figura 3.7. Estrutura de uma pergunta no Stack Overflow

Além disso, com o objetivo de identificar o número apropriado de tópicos (e não uma escolha aleatória), aplicamos uma técnica de particionamento chamada *Silhouette*. Esta técnica retorna um número apropriado (silhueta) de agrupamentos em relação às características dos documentos que estão sendo analisados [Rousseeuw 1987]. Esse número, fornecido pela silhueta, foi aplicado como entrada para o LDA.

Cada cluster é representado por uma silhueta chamada, que é baseada na comparação de sua rigidez e separação. Essa silhueta mostra quais objetos estão bem dentro de seu cluster e quais estão meramente em algum lugar entre esses clusters. A largura média da silhueta fornece uma avaliação da validade do agrupamento. Pode ser usado para selecionar um número apropriado de clusters. A *Silhouette* fornece valores no intervalo de -1 a 1, em que 1 significa que as amostras pertencentes ao cluster estão distantes dos outros clusters, 0 significa que a divisão entre os clusters já está na borda da separação e -1 significa que algumas amostras têm uma chance de serem atribuídas ao cluster errado. Os resultados são analisados pelos pesquisadores. Por exemplo, os tópicos mais frequentes, as questões ou respostas que são relacionadas a estes tópicos e também as séries temporais que podem indicar o comportamento da comunidade a partir da interação dentro do repositório.

Nesse cenário, podemos perceber que a área de MSR se concentra na descoberta de informações úteis sobre software, extraindo e analisando dados de diferentes repositórios de software [Ahasanuzzaman *et al.* 2016]. As abordagens de MSR foram usadas para diferentes objetivos como, por exemplo, análises de contribuição e comportamento do desenvolvedor.

O comportamento do desenvolvedor pode ser analisado a partir de emoções. Ele pode fornecer uma perspectiva diferente para interpretar a satisfação do desenvolvedor sobre o seu envolvimento no processo de desenvolvimento [Murgia *et al.* 2014]. Emoção é um “estado psicológico que surge espontaneamente (não por meio de esforço consciente) e é por vezes acompanhado de alterações fisiológicas” [Novielli *et al.* 2014]. Tipos gerais de emoções são: alegria, tristeza, raiva, desgosto e medo. A alegria está normalmente associada a conquistas positivas. A raiva geralmente está relacionada a um forte sentimento de desprazer e beligerância despertado por um erro. A tristeza é geralmente expressa por infelicidade. Repulsa é uma repugnância causada por algo ofensivo. Finalmente, o medo é uma emoção angustiante despertada pelo perigo iminente, mal, dor etc.

3.6. *Big Data* e Ferramentas

No cenário discutido na seção anterior, verificamos que existe um grande volume de informações geradas a partir das interações dos desenvolvedores dentro de um ecossistema. Por exemplo, somente no SO já são mais de 14 milhões de questões que podem ser mineradas para fornecer *insights* para as organizações que mantêm ecossistemas. Esse grande volume de dados tem sido estudado dentro da área de *Big Data*. Para processar esses dados, há diversos *frameworks*; neste trabalho, vamos utilizar o Spark.

O Spark estende o MapReduce, que é um modelo de programação desenhado para processar grandes volumes de dados em paralelo, dividindo o trabalho em um conjunto de tarefas independentes e evitando mover os dados durante o seu

processamento. Isso é feito por meio de recursos como armazenamento de dados em memória e processamento próximo ao tempo real. Neste contexto, o desempenho pode ser várias vezes mais rápido do que outras tecnologias de Big Data. A seguir, apresentamos o passo-a-passo do funcionamento do MapReduce (Figura 3.8):

1. Divide os dados de entrada em M pedaços. Em seguida, inicia várias cópias do programa em um cluster de computadores. Há uma cópia principal: *master*. As outras cópias são denominadas *workers* e recebem trabalho do *master*. Existem M tarefas de *map* e R tarefas de *reduce* para serem assinaladas. O *master* seleciona *workers* ociosos e assinala a eles uma tarefa de *map* ou de *reduce*;
2. Há um *worker* dedicado à tarefa de mapear o conteúdo correspondente ao pedaço da entrada. Ele interpreta as tuplas de chave/valor a partir dos dados de entrada e passa como parâmetro para a função de *map* do usuário. As tuplas chave/valor intermediárias produzidas pela função de *map* são armazenadas em memória;
3. Frequentemente, as tuplas de dados dos *buffers* são escritas em disco, particionados em R regiões pela função de particionamento. A localização dessas tuplas de dados no disco é informada ao *master*, que irá repassar essa localização para os *workers* com tarefas de *reduce*;
4. A partir do momento que um *worker* de *reduce* é notificado da localização dos dados pelo *master*, este faz uma chamada de procedimento remota para buscar os dados do disco local dos *workers* de *map*. Quando os dados já foram lidos, ele ordena os dados pelas chaves intermediárias, para que todas as ocorrências de uma mesma chave sejam agrupadas em conjunto. É realizada uma ordenação, pois muitas chaves diferentes são mapeadas para a mesma tarefa de *reduce*. Se a quantidade de dados intermediários é muito grande para caber em memória, uma ordenação externa em disco é executada;
5. O *worker* de *reduce* vasculha os dados intermediários já ordenados e, para cada chave encontrada, ele passa como conteúdo para a chave os valores intermediários para a função de *reduce* definida pelo usuário. A saída de cada função de *reduce* é adicionada ao final de um arquivo de saída para aquela partição de *reduce*;
6. Quando todas as tarefas de *map* e *reduce* foram terminadas, o *master* retorna o programa do usuário;
7. Ao final da execução do programa, o resultado está disponível em R arquivos (um para cada operação de *reduce*). É comum que os arquivos de resultados de uma operação de MapReduce sejam entradas para outra chamada de MapReduce.

O Spark armazena resultados intermediários na memória, em vez de escrevê-los no disco, o que é muito útil quando se precisa processar o mesmo conjunto de dados mais de uma vez. Logo, ele é um mecanismo de execução que funciona tanto na memória como em disco. Assim, é possível usá-lo para o processamento de conjuntos de dados maiores que a memória agregada em um cluster. O Spark armazenará a maior quantidade possível de dados na memória e, em seguida, irá persisti-los em disco. Com

esse mecanismo de armazenamento de dados em memória, o uso do Spark traz vantagens de desempenho. Outras características são:

- Suporta mais do que apenas as funções de *map* e *reduce*;
- Otimiza o uso de operadores de grafos arbitrários;
- Avaliação sob demanda de consultas de *Big Data* contribui com a otimização do fluxo global do processamento de dados;
- Fornece APIs concisas e consistentes em Scala, Java e Python;
- Oferece *shell* interativo para Scala e Python.

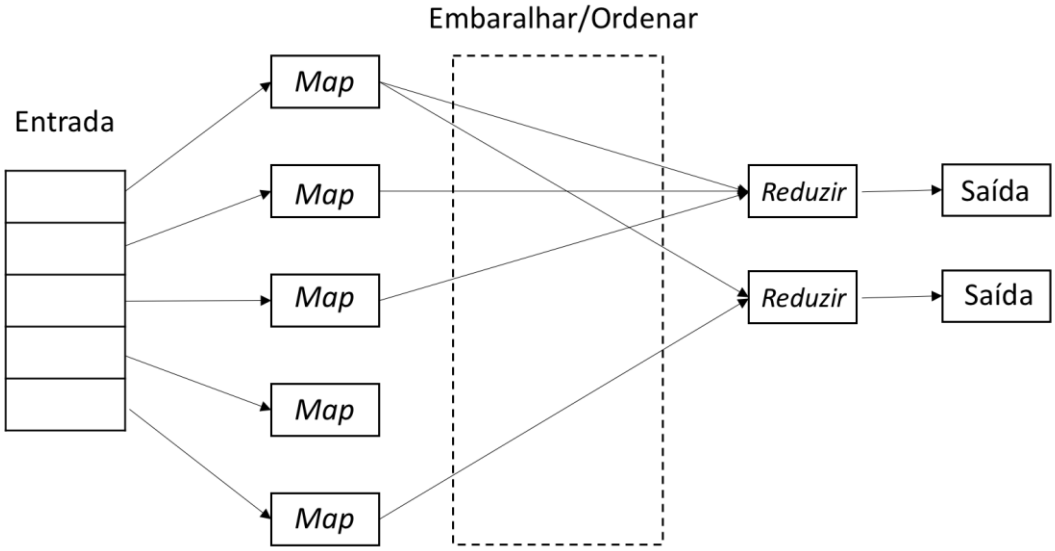

Figura 3.8. Passo-a-passo da execução do MapReduce

O Spark é escrito na linguagem Scala e executa em uma máquina virtual Java. Atualmente, suporta as seguintes linguagens para o desenvolvimento de aplicativos: Scala, Java, Python, Clojure e R.

Outro *framework* que envolve documentação e códigos para mineração de dados é o ScikitLearn[2]. O ScikitLearn é baseado na linguagem de programação Python e é um *framework open-source*. Este *framework* aborda as categorias de: classificação (identificação de que categoria um objeto pertence), regressão (prevê um valor contínuo de um atributo associado um objeto), agrupamento (agrupamento automático de objetos similares em conjuntos), redução de dimensionalidade (redução do número de variáveis randômicas a serem consideradas), seleção de modelos (comparação, validação e escolha de parâmetros e modelos) e pré-processamento (extração de características e normalização). Neste trabalho, abordamos a parte de pré-processamento, redução de dimensionalidade e agrupamentos.

[2] http://scikit-learn.org/stable/

3.7. Considerações Finais

Para o estudo da governança de desenvolvedores em ECOS, é necessário buscar o equilíbrio entre os objetivos da organização que mantém o ECOS e as expectativas dos desenvolvedores. Desta forma, a governança de desenvolvedores considera a disponibilização de processos e de um ambiente que favoreçam o bem-estar econômico e social dos desenvolvedores. Nesse cenário, como forma de obter informações sobre o engajamento e interações dos desenvolvedores, é necessária a utilização de técnicas de mineração de repositórios de software. Repositórios de software guardam informações sobre, por exemplo, Questões e Perguntas técnicas (e.g., Stack Overflow) e projetos de software (e.g., Github). Neste trabalho, compartilhamos conceitos de ECOS, governança e mineração de repositórios de software. Ainda sobre mineração de repositórios de software compartilhamos uma abordagem para o pré-processamento, extração de tópicos e análise de resultados.

Agradecimentos

O autor Rodrigo Pereira dos Santos agradece ao DPq/PROPGPI/UNIRIO pelo apoio para realização desse trabalho.

Referências

A.E., H. (2008). The road ahead for mining software repositories. In Proceedings of the Frontiers of Software Maintenance, FoSM, p. 48-57.

Ahasanuzzaman, M., Asaduzzaman, M., Roy, C. K. and Schneider, K. A. (2016). Mining duplicate questions in stack overflow. Proceedings of the 13th International Workshop on Mining Software Repositories - MSR ’16, p. 402–412.

Alves, C., Oliveira, J. and Jansen, S. (2017). Software Ecosystems Governance - A Systematic Literature Review and Research Agenda. Proceedings of the 19th International Conference on Enterprise Information Systems, n. January, p. 215–226.

Baars, A. and Jansen, S. (2012). A Framework for Software Ecosystem Governance. Proceedings of the International Conference of Software Business. p. 168–180.

Begon, M., Townsend, C. R. and Harper, J. L. (2007). Ecologia - De Indivíduos a Ecossistemas. 4. ed. Porto Alegre: Artmed.

Bhat, V. (2014). Min (e) d Your Tags : Analysis of Question Response Time in StackOverflow. Proceedings of the , Advances in Social Networks Analysis and Mining (ASONAM). p. 328–335.

Bosch, J. (2009). From Software Product Lines to Software Ecosystems. In SPLC ’09 Proceedings of the 13th International Software Product Line Conference. p. 111-119.

Casalnuovo, C., Vasilescu, B., Devanbu, P. and Filkov, V. (2015). Developer Onboarding in GitHub: The Role of Prior Social Links and Language Experience. ESEC/FSE conf., p. 1–12.

De Fontão, A. L., Dos Santos, R. P., Filho, J. F. and Dias-Neto, A. C. (2016). MSECO-DEV: Application development process in mobile software ecosystems. In Proceedings of the International Conference on Software Engineering and Knowledge Engineering. P. 317-322.

Dhungana, D., Groher, I., Schludermann, E. and Biffl, S. (2010). Software Ecosystems vs . Natural Ecosystems : Learning from the Ingenious Mind of Nature. Proceedings of the 4th European Conference on Software Architecture Companion Volume (ECSA '10), p. 96–102.

Dubinsky, Y. and Kruchten, P. (2009). Software development governance (SDG): report on 2nd workshop. ACM SIGSOFT Software Engineering Notes, v. 34, n. 5, p. 46–47.

Fagerholm, F. (2015). Software Developer Experience : Case Studies in Lean-Agile and Open Source Environments.

Fagerholm, F. and Münch, J. (2012). Developer experience: Concept and definition. In 2012 International Conference on Software and System Process, ICSSP 2012 - Proceedings.

Farias, M. A. de F., Novais, R., Colaço, M., *et al.* (2016). A Systematic Mapping Study on Mining Software Repositories. In 31st ACM/SIGAPP Symposium on Applied Computing.

Fontão, A., Bonifácio, B., Dias-Neto, A., Bezerra, A. and Santos, R. (2014). MSECO skill: Construction skills developer ecosystem in mobile software | MSECO skill: Construção de competências de desenvolvedores em ecossistemas de software móvel. In CIBSE 2014: Proceedings of the 17th Ibero-American Conference Software Engineering.

Forman, G. (2008). BNS feature scaling: an improved representation over tf-idf for svm text classification. Proceedings of the 17th ACM conference on Information and knowledge management, p. 263–270.

Genc-Nayebi, N. and Abran, A. (2016). A Systematic Literature Review: Opinion Mining Studies from Mobile App Store User Reviews. Journal of Systems and Software,

Hyrynsalmi, S., Seppänen, M. and Suominen, A. (2014). Sources of value in application ecosystems. Journal of Systems and Software, v. 96, p. 61–72.

Iansiti, M. and Levien, R. (2004). Strategy as ecology. Harvard business review, v. 82, p. 68–78, 126.

Iansiti, M. and Richards, G. L. (2006). The information technology ecosystem: Structure, health, and performance. Antitrust Bulletin, v. 51, p. 77–110.

Jansen, S. and Bloemendal, E. (2013). Defining App Stores: The Role of Curated Marketplaces in Software Ecosystems. Software Business. From Physical Products to Software Services and Solutions. p. 195–206.

Jansen, S., Brinkkemper, S. and Cusumano, M. (2013). Software Ecosystems.

Jansen, S., Brinkkemper, S. and Finkelstein, A. (2009). Business Network Management as a Survival Strategy : A Tale of Two Software Ecosystems. n. 2, p. 34–48.

Jansen, S., Brinkkemper, S., Souer, J. and Luinenburg, L. (2012). Shades of gray: Opening up a software producing organization with the open software enterprise model. Journal of Systems and Software, v. 85, n. 7, p. 1495–1510.

Jansen, S. and Cusumano, M. A. (2012). Defining software ecosystems : A survey of software platforms and business network governance. In Proceedings of the international Workshop on Software Ecosystems 2012.

Jansen, S., Finkelstein, A. and Brinkkemper, S. (2009). A sense of community: A research agenda for software ecosystems. 2009 31st International Conference on Software Engineering - Companion Volume, p. 187–190.

Kim, H., Kim, J., Lee, Y., Chae, M. and Choi, Y. (2002). An Empirical Study of the Use Contexts and Usability Problems in Mobile Internet. In HICSS "02 Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS"02). . IEEE Computer Society Washington.

Krestel, R., Fankhauser, P. and Nejdl, W. (2009). Latent dirichlet allocation for tag recommendation. In Proceedings of the third ACM conference on Recommender systems. . http://portal.acm.org/citation.cfm?doid=1639714.1639726.

Lin, B. and Serebrenik, A. (2016). Recognizing Gender of Stack Overflow Users. MSR conf, p. to appear.

Manikas, K. (2016). Revisiting software ecosystems Research: A longitudinal literature study. Journal of Systems and Software, v. 117, p. 84–103.

Manikas, K. and Hansen, K. M. (2013). Reviewing the Health of Software Ecosystems – A Conceptual Framework Proposal. In Proceedings of the 5th International Workshop on Software Ecosystems.

Miranda, M., Ferreira, R., De Souza, C. R. B., Figueira Filho, F. and Singer, L. (2014). An exploratory study of the adoption of mobile development platforms by software engineers. Proceedings of the 1st International Conference on Mobile Software Engineering and Systems - MOBILESoft 2014, p. 50–53.

Murgia, A., Tourani, P., Adams, B. and Ortu, M. (2014). Do developers feel emotions? an exploratory analysis of emotions in software artifacts. Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014, p. 262–271.

Mustofa, K., Biffl, S., Schatten, A., Tjoa, A. M. and Wahyudin, D. (2007). Monitoring the "health" status of open source web-engineering projects. International Journal of Web Information Systems

Novielli, N., Calefato, F. and Lanubile, F. (2014). Towards discovering the role of emotions in stack overflow. Proceedings of the 6th International Workshop on Social Software Engineering - SSE 2014, p. 33–36.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis.

Santos, R. P. and Werner, C. (2011). Treating business dimension in Software Ecosystems. In Proceedings of the International Conference on Management of Emergent Digital EcoSystems, MEDES'11.

Santos, R., Werner, C., Barbosa, O. and Alves, C. (2012). Software ecosystems: Trends and impacts on software engineering. Proceedings - 2012 Brazilian Symposium on Software Engineering, SBES 2012, p. 206–210.

Santos, R. P. Dos and Werner, C. M. L. (aug 2012). ReuseECOS: An Approach to Support Global Software Development through Software Ecosystems. In 2012 IEEE Seventh International Conference on Global Software Engineering Workshops. . IEEE.

Schaeffer, D. J., Herricks, E. E. and Kerster, H. W. (1988). Ecosystem health: I. Measuring ecosystem health. Environmental Management

Shah, C., Kitzie, V. and Choi, E. (2014). Questioning the question - Addressing the answerability of questions in community question-answering. In Proceedings of the Annual Hawaii International Conference on System Sciences.

Shull, F., Singer, J. and Sjøberg, D. I. K. (2008). Guide to advanced empirical software engineering.

Taylor, R. N. (2013). The role of architectural styles in successful software ecosystems. In ACM International Conference Proceeding Series.

Tiwana, A., Konsynski, B. and Bush, A. A. (dec 2010). Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics. Information Systems Research, v. 21, n. 4, p. 675–687.

Wazlawick, R. S. (2013). Engenharia de softtware: conceitos e práticas. 1. ed. Rio de Janeiro: Elsevier Ltd.

Zagalsky, A., Teshima, C. G., German, D. M., Storey, M. and Poo-caamaño, G. (2016). How the R Community Creates and Curates Knowledge : A Comparative Study of Stack Overflow and Mailing Lists. p. 441–451.

Biografia dos Autores

Awdren Fontão é doutorando do Programa de Pós-Graduação em Informática do Instituto de Computação (ICOMP) na Universidade Federal do Amazonas (UFAM). Suas áreas de pesquisa envolvem: Ecossistemas de Software, Aplicações Móveis, Relações com Desenvolvedores. Tem experiência com comunidades de desenvolvedores, onde atuou por seis anos na área de *Developer Relations* como evangelista de desenvolvedores da Nokia e Microsoft (Brasil/América Latina).

Igor Scaliante Wiese é professor do Departamento de Computação na Universidade Federal Tecnológica do Paraná (UTFPR) – Campo Mourão, onde ele pesquisa técnicas de mineração de repositórios, aspectos humanos na engenharia de software e tópicos relacionados. Wiese fez doutorado em Ciência da Computação na Universidade de São Paulo e foi pesquisador visitante na University of California - Irvine.

Rodrigo Pereira dos Santos é professor do Departamento de Informática Aplicada e membro efetivo do Programa de Pós-Graduação em Informática da Universidade Federal do Estado do Rio de Janeiro (UNIRIO). Atuou como pesquisador visitante na University College London (UCL, 2014-2015) e como consultor em projetos de P&D em engenharia de sistemas na indústria nacional pela Fundação Coppetec (2008-2017). É editor-chefe da iSys: Revista Brasileira de Sistemas de Informação. É membro do Comitê Gestor da Comissão Especial de Sistemas de Informação da SBC. Seus temas de pesquisa envolvem Ecossistemas de Software e Engenharia de Requisitos.

Arilo Claudio Dias Neto é professor do Instituto de Computação (ICOMP) na Universidade Federal do Amazonas (UFAM). Ele lidera o grupo de pesquisa em Experimentação e Teste de Software e participa de pesquisa na indústria e projetos de desenvolvimento. Revisor de periódicos, como: Information and Software Technology, Journal of Information Processing Systems, Journal of The Brazilian Computer Society (Online) e Software Quality Journal. CTO da Méliuz.